1961-1986

25 Years

Leading
Toward
The
Future

DECUS U.S. CHAPTER SIGs NEWSLETTERS

September 1986 Volume 2, Number 1

GENERAL TABLE OF CONTENTS

SECTIONS **PAGE NO.**

ARTIFICIAL INTELLIGENCE SIG

BUSINESS APPLICATIONS SIG

COMMERCIAL LANGUAGES SIG

DATA ACQUISITION ANALYSIS, RESEARCH AND CONTROL SIG

DATA MANAGEMENT SIG

DATATRIEVE/4GL SIG

EDUSIG

GRAPHICS APPLICATIONS SIG

HARDWARE MICRO SIG

IAS SIG

LANGUAGES AND TOOLS SIG

LARGE SYSTEMS SIG

MUMPS SIG

NETWORKS SIG

OFFICE AUTOMATION SIG

PERSONAL COMPUTER SIG

RSTS SIG

RSX SIG

RT SIG

SITE MANAGEMENT AND TRAINING SIG

UNISIG

VAX SYSTEMS SIG

LIBRARY

QUESTIONNAIRE SECTION

ARTIFICIAL INTELLIGENCE

DECUS

ARTIFICIAL INTELLIGENCE SIC

Chairperson
Cheryl Jalbert
JCC
Granville, OH

Assistant Chairman
Don Rosenthal
Space Telescope Science Inst.
Baltimore, MD

Symposium Coordinator
Pam Vavra
KMS Fusion Inc.
Ann Arbor, MI

Session Note Editor
George Humfeld
Naval Sea Systems Command
Washington, DC

Newsletter Editor
Terry Shannon
Digital Review
Boston MA

Newsletter Publisher
Bob Zeek
Pfizer Inc.
Groton, CT

Membership Coordinator
Pam Vavra
KMS Fusion Inc.
Ann Arbor, MI

PSS Scheduling
Tom Viana

Store & Buttons
Sally Townsend
Inst. Defense Analysis
Alexandria, VA

Quality Control Chair
Dick Ciero
Harris Corp.
Palm Bay, FL

Quality Control
Carol Guyermelli

Site Coordinator, Anaheim
Chris Goodard

Volunteer Coordinator, Anaheim
Peter Macdonough
Tractor Inc.
California, MD

Members-at-Large
George Humfeld

Matt Matthews IV

Evaluation Research Corp
King George, VA

Chair of the Newletter Task Force
Becky Wise
Amdahl CSD
Richardson, TX

DEC Counterpart
Art Beane
Digital Equipment Corporation
Hudson, MA

(THE (LINKED LIST))

THE NEWSLETTER OF THE DECUS ARTIFICIAL INTELLIGENCE SPECIAL INTEREST GROUP

...it's the real thing.

Volume 3, Number 1 September 1986

FROM THE EDITOR

After an extended hiatus, THE (LINKED (LIST)) is back! Despite our protracted absence from the pages of the DECUS SIGs NEWSLETTERS, the AI SIG has reached critical mass and continues to grow at a brisk pace.

"Time flies when you're having fun", as the expression goes. The Fall 1986 symposium is only a month away, and I feel like I'm still recovering from the Dallas gathering. If you've recovered from the Spring 86 event and plan to attend the upcoming symposium in San Francisco, you are strongly advised to submit your registration to the DECUS office in Marlboro as soon as possible. The size of the facilities and meeting rooms in the convention center may make it necessary to limit symposium attendance, so make sure you're registered well before the deadline.

As usual, I'm on the lookout for submissions to the newsletter. If you have an article, comment, suggestion or random thought, send it in! There are several public domain AI software packages and a variety of books available for review by SIG members. If you're interested, you can contact me at my (new) work address, listed below. See you in San Francisco!

Terry C. Shannon
Technical Editor
DIGITAL REVIEW
Prudential Tower Suite 1390
Boston, MA 02199
(617) 375-4321

JOIN THE ARTIFICIAL INTELLIGENCE SIG IN SAN FRANCISCO!

By Pam Vavra, AI SIG Symposium Coordinator

For the San Francisco Symposium, the AI SIG offers several excellent Pre-Symposium Seminars, plus a full week (including Friday!) of fast-paced, high-content sessions. Highlights include pre-lunch plenary talks by Earl Sacerdoti, of Teknowledge, Inc., pioneer in expert system tool development for non-technical end-users; and by Richard Gabriel, known for his work in developing LISP, the language most frequently used by AI programmers. We are very pleased to have them share their expertise and experiences with us.

We will offer a wide range of sessions on various AI languages and tools which culminate on Thursday evening with 'The Great Tool Debate'. This discussion promises to be spirited, informative and fun! Our traditional joint reception with LANGUAGES AND TOOLS and UNISIG, where Digital developers promise to entertain us again, is scheduled as an L&T session on Wednesday night.

Following are descriptions of the streams into which the AI sessions have been organized to help give you a more global picture. They are arranged here in chronological order; specific sessions and times are given in the SAAG and the Index. Notice that the roadmap session will be particularly valuaole this time, because we must hop from room to room. Moreover, for the first time, we have AI sessions that overlap in time, so purchasing the AI SIG Session Notes may be vital.

We hope to see you in San Francisco this October as we continue to work toward facilitating information exchange among DECUS AIers. Your participation is needed to help our young group continue to grow and develop in a manner that spells excellence and success for us all.

AI STREAMS

AI SAMPLER -- geared toward the novice AIer; includes survey sessions of elementary and advanced topics.

TOOLS I -- a few product descriptions.

PLENARY TALK -- Earl Sacerdoti from Teknowledge, a pioneer in expert system tool development for non-technical end-users.

OPS5 -- highly technical sessions by developers and advanced users.

LISP -- from elementary to advanced.

PLENARY TALK -- Dr. Richard Gabriel, well-known LISP developer.

EXPERT SYSTEMS EXAMPLES -- informal demos, expert systems of broad interest, and celebrated expert systems featured in literature.

RECEPTION -- Tri-SIG (AI, L&T, and UNISIG) welcoming reception for interested SIG volunteers, newcomers, etc.

KNOWLEDGE ENGINEERING -- numerous sessions from elementary to advanced issues.

TOOLS II -- mostly user-developed AI software tools.

ADVANCED TOPICS -- includes distributed knowledge, language definitions, testing rule-based systems, and more.

THE GREAT TOOL DEBATE -- When to use which tool? When to develop custom tools? Tool makers and users encouraged to express their opinions.

MANAGEMENT FOCUS FOR SOPHISTICATED NEOPHYTES -- selected examples illustrate problems unique to prototyping, operational systems, and mature systems, plus how to manage expert systems programs, how to "think LISP" as a company, etc.

PROLOG -- general introduction plus Digital third party product description.

Sessions Sponsored by the
Artificial Intelligence Special Interest Group (AI SIG)
DECUS, Fall 1986, San Francisco, CA

Monday October 6, 1986

9:00 a.m. - 10:00 a.m. Ballroom H [750] AI044
ARTIFICIAL INTELLIGENCE SIG ROADMAP AND OPENING SESSION
Cheryl Jalbert, J C C, Granville, OH

10:00 a.m. - 11:00 a.m. Ballroom H [750] AI037
INTRODUCTION TO ARTIFICIAL INTELLIGENCE AND ITS APPLICATIONS AT DIGITAL
Art Beane, Digital Equipment Corp., Hudson, MA

11:00 a.m. - 12:00 noon Ballroom H [750] AI062
THE FIFTH GENERATION
Terry Shannon, Digital Review, Boston, MA

12:30 p.m. - 1:15 p.m. 236 & 238 [150] AI070
AI ON THE RAINBOW
Dr. Khin Yin, Kent State University, Warren, OH

1:15 p.m. - 2:00 p.m. 236 & 238 [150] AI097
AI ON PCS
Terry Shannon, Digital Review, Boston, MA

2:00 p.m. - 3:00 p.m. 236 & 238 [150] AI017
ART, AUTOMATED REASONING TOOL
Don Gamon, Inference Corporation, Los Angeles, CA

2:30 p.m. - 3:30 p.m. 208 & 210 [35] AI029
VAX LISP SUBSET FOR LOW-END DIGITAL HARDWARE
Digital Equipment Corp., Merrimack, NH

3:00 p.m. - 3:30 p.m. 236 & 238 [150] AI093
WHY USE AI IN THE NEWSPAPER INDUSTRY
Michael Stock, Composition Systems, Elmsford, NY

3:30 p.m. - 4:00 p.m. 228 & 230 [75] AI060
DIGITAL'S PERSPECTIVE ON FIFTH GENERATION TECHNOLOGY
Art Beane, Digital Equipment Corp., Hudson, MA

4:00 p.m. - 4:30 p.m. 228 & 230 [75] AI056
AI IN THE BANKING AND INSURANCE INDUSTRIES
Roger Jones, Bankers Trust, New York, NY

4:30 p.m. - 5:30 p.m. 228 & 230 [75] AI047
GETTING STARTED IN AI -- NOVICE QUESTION AND ANSWER SESSION
Cheryl Jalbert, J C C, Granville, OH

6:00 p.m. - 7:00 p.m. 228 & 230 [75] AI045
ARTIFICIAL INTELLIGENCE SIG BUSINESS MEETING
Cheryl Jalbert, J C C, Granville, OH

7:00 p.m. - 8:00 p.m. Ballroom F [750] AI094
AI AT MICROELECTRONICS AND COMPUTER TECHNOLOGY CORPORATION (MCC)
Clive Dawson, MCC, Austin, TX

7:00 p.m. - 7:30 p.m. 228 & 230 [75] AI038
SURVEY OF AI LITERATURE
Suzanne McGuire, Nagog Woods, MA

7:30 p.m. - 8:00 p.m. 228 & 230 [75] AI083
AI INFORMATION - A WEALTH OF RESOURCES
Suzanne McGuire, Nagog Woods, MA

8:00 p.m. - 8:30 p.m. 228 & 230 [75] AI051
INTRODUCTION TO ROBOTICS
Sandra Traylor, Target Systems

8:00 p.m. - 9:00 p.m. Ballroom F [750] AI003
INTELLECT/RDB TECHNICAL OVERVIEW
Kelsey Thompson, Artificial Intelligence Corp., Waltham, MA

9:00 p.m. - 9:30 p.m. Ballroom F [750] AI025
SEMANTIC PROBLEMS IN AI AND OTHER AREAS OF COMPUTING
David Slater, Computer Sciences Corporation, Falls Church, VA

9:30 p.m. - 11:00 p.m. Ballroom F [750] AI035
NATURAL LANGUAGE MISPARSING AND AI ESOTERICA
David Slater, Computer Sciences Corporation, Falls Church, VA

Tuesday October 7, 1986

9:00 a.m. - 10:00 a.m. Section E [450] AI030
RS/EXPLORE EXPERT SYSTEM TO RS/1 USERS
Nouna Kettaneh, BBN Software, Cambridge, MA
Jeremy Pool, BBN Software, Cambridge, MA

10:00 a.m. - 11:00 a.m. Section E [450] AI040
VAX OPS5 PRODUCT DESCRIPTION
Art Beane, Digital Equipment Corp., Hudson, MA

11:00 a.m. - 12:00 noon Section E [450] AI011
PLENARY TALK BY EARL SACERDOTI: EXPERT SYSTEM DEVELOPMENT WITH S.1
Earl D. Sacerdoti, Teknowledge, Inc., Palo Alto, CA

2:00 p.m. - 2:30 p.m. Campground B [100] AI042
DIGITAL'S PROCESS FOR SELECTING THIRD PARTY SOFTWARE PRODUCTS
Elinor Burns, Digital Equipment Corp., Merrimack, NH

2:00 p.m. - 3:00 p.m. Section E [450] AI018
DESIGNING AND BUILDING SYSTEMS IN OPS5
Tom Cooper, Digital Equipment Corp., Hudson, MA
John Frost, Digital Equipment Corp., Hudson, MA

3:00 p.m. - 3:45 p.m. Section E [450] AI020
WRITING EFFICIENT OPS5 RULES
Tom Cooper, Digital Equipment Corp., Hudson, MA
John Frost, Digital Equipment Corp., Hudson, MA

3:45 p.m. - 4:30 p.m. Section E [450] AI023
OPS5 CONTROL CONSTRUCTS
Don Rosenthal, Space Telescope Science Inst., Baltimore, MD

4:30 p.m. - 5:00 p.m. Section E [450] AI021
OPS5 PROGRAMMING TECHNIQUES
Tom Cooper, Digital Equipment Corp., Hudson, MA
John Frost, Digital Equipment Corp., Hudson, MA

5:00 p.m. - 6:00 p.m. Section E [450] AI039
OPS5 QUESTION AND ANSWER SESSION
Don Rosenthal, Space Telescope Science Inst., Baltimore, MD

Wednesday October 8, 1986

9:00 a.m. - 9:30 a.m. 220 - 226 [250] AI099
CONVERTING PROCEDURAL LANGUAGE PROGRAMMERS TO LISP
Barbara Eshima, Lucid, Inc., Menlo Park, CA

9:30 a.m. - 10:30 a.m. 220 - 226 [250] AI067
VAX LISP FOR VMS AND ULTRIX, AND THE AI VAXSTATION
Art Beane, Digital Equipment Corp., Hudson, MA

10:30 a.m. - 11:00 a.m. 220 - 226 [250] AI088
VAX LISP FOR ADVANCED USERS
Art Beane, Digital Equipment Corp., Hudson, MA

11:00 a.m. - 11:30 a.m. 220 - 226 [250] AI027
A VAX-BASED INTEGRATED LISP ENVIRONMENT
James C. Sanborn, Martin Marietta Labs, Baltimore, MD

11:30 a.m. - 12:30 p.m. 220 - 226 [250] AI078
PLENARY TALK BY RICHARD GABRIEL: PERFORMANCE IN THE EVOLUTION OF LISP AS A GENERAL PURPOSE LANGUAGE
Dr. Richard Gabriel, Lucid, Inc., Menlo Park, CA

1:30 p.m. - 2:30 p.m. Campground B [100] AI080
DEVELOPMENT OF A VAX TUNER USING OPS5
Robert A. Small, Vitro Corp., New London, CT

2:00 p.m. - 3:00 p.m. 202 - 206 [100] AI041
LISP QUESTION AND ANSWER SESSION
David Slater, Computer Sciences Corporation, Falls Church, VA

2:30 p.m. - 3:30 p.m. Campground B [100] AI022
A RULE-BASED BIDDING SYSTEM FOR CONTRACT BRIDGE
Don Rosenthal, Space Telescope Science Inst., Baltimore, MD

3:30 p.m. - 4:00 p.m. 232 & 234 [150] AI032
HEURISTIC LISP EXPERT SYSTEM FOR VMS SECURITY LOG OVERVIEW
Sarah Townsend, IDA, Alexandria, VA

4:00 p.m. - 5:0u p.m. 232 & 234 [150] AI013
DISTRIBUTED EXPERT SYSTEMS EXAMPLE

5:00 p.m. - 6:00 p.m. 232 & 234 [150] AI010
ARTIFICIAL INTELLIGENCE IN A DISTRIBUTED NETWORK, COOPERATING EXPERT SYSTEMS
Michael Stock, Composition Systems, Elmsford, NY

Thursday October 9, 1986

9:00 a.m. - 10:00 a.m. 250 & 262 [125] AI085
KNOWLEDGE ENGINEERING AND AI TECHNOLOGY TRANSFER AT DIGITAL
Doug Fulrath, Digital Equipment Corp., Hudson, MA

10:00 a.m. - 10:30 a.m. 250 & 262 [125] AI084
KNOWLEDGE REPRESENTATION ISSUES
Rebecca Wise, Thompson Components Mostek, Carrollton, TX

10:30 a.m. - 11:00 a.m. 250 & 262 [125] AI081
AN IMPLEMENTATION OF A RULE EDITOR - RULE * CALC TM
John Thorp, Martin Marietta Labs, Baltimore, MD

11:00 a.m. - 11:45 a.m. 250 & 262 [125] AI001
ON SOLVING VERY LARGE KNOWLEDGE ENGINEERING PROBLEMS
Philip Y. Li, JPL, Pasadena, CA

11:45 a.m. - 12:30 p.m. 250 & 262 [125] AI006
KNOWLEDGE ACQUISITION TOOLS
Eric J. Bean, Weyerharuser Sci. Computing, Tacoma, WA

1:30 p.m. - 2:30 p.m. 250 & 262 [125] AI005
EXPERT SYSTEMS CONCEPTS CRASH COURSE
David Frydenlund, U. S. Coast Guard R&D, Washington, DC

2:30 p.m. - 3:00 p.m. 250 & 262 [125] AI015
SO YOU THINK YOU'RE READY TO BUILD AN EXPERT SYSTEM?
Elinor Burns, Digital Equipment Corp., Merrimack, NH

2:30 p.m. - 3:30 p.m. Campground B [100] AI016
SURVEY OF EXPERT SYSTEM TOOLS POSITIONING FROM A PRODUCER'S VIEW
Elinor Burns, Digital Equipment Corp., Merrimack, NH

3:00 p.m. - 4:00 p.m. 250 & 262 [125] AI100
A KNOWLEDGE-BASED SOFTWARE ENGINEERING ENVIRONMENT WITH EXPERT SYSTEM TOOLS
Gary B. Lamoant, Wright State Univ, Dept of CS, Dayton, OH

4:00 p.m. - 5:00 p.m. 250 & 262 [125] AI007
AN EXPERT SYSTEM TOOL THAT STARTED OUT ON PERSONAL COMPUTERS
Elinor Burns, Digital Equipment Corp., Merrimack, NH

5:00 p.m. - 6:00 p.m. 250 & 262 [125] AI082
A DISTRIBUTED KNOWLEDGE NETWORK MODEL
J. Steven Hughes, Jet Propulsion Lab, Pasadena, CA

6:00 p.m. - 7:00 p.m. 228 & 230 [75] AI002
EXPERT SYSTEMS IN A MANUFACTURING ENVIRONMENT
Elinor Burns, Digital Equipment Corp., Merrimack, NH

6:00 p.m. - 7:00 p.m. 250 & 262 [125] AI026
PANEL ON TESTING RULE-BASED EXPERT SYSTEMS
Tom Cooper, Digital Equipment Corp., Hudson, MA
John Frost, Digital Equipment Corp., Hudson, MA

8:00 p.m. - 9:00 p.m. 250 & 262 [125] AI096
AN EXPERT SYSTEM SHELL FOR USE IN A DISTRIBUTED OBJECT-ORIENTED ENVIRONMENT
Randall W. Hill, Jr., Jet Propulsion Laboratory, Pasadena, CA

9:00 p.m. - 9:30 p.m. 250 & 262 [125] AI075
DEFINING AN EXPERT SYSTEM LANGUAGE
Stephen Pacheco, Ship Analytics, North Stonington, CT

9:30 p.m. - 10:00 p.m. 250 & 262 [125] AI076
IMPLEMENTING AN EXPERT SYSTEM LANGUAGE COMPILER
Stephen Pacheco, Ship Analytics, North Stonington, CT

10:00 p.m. - 11:00 p.m. 250 & 262 [125] AI091
THE GREAT TOOL DEBATE -- DISCUSSION WITH THE DEVELOPERS OF VARIED LANGUAGES AND TOOLS USED IN AI
Cheryl Jalbert, J C C, Granville, OH

Friday October 10, 1986

9:00 a.m. - 9:45 a.m. 232 & 234 [150] AI064
FLARES--AN EXPERT SYSTEM FOR FAULT LOCALIZATION, ASSESSMENT, AND REPAIR OF COMPLEX ELECTRONIC EQUIPMENT
John Williamson, NUSC, Newport, RI

9:45 a.m. - 10:30 a.m. 232 & 234 [150] AI077
AN EXPERT SYSTEM PROTOTYPE FOR POWER PLANT DESIGN EVALUATION
Stephen L. Arnold, Bechtel Power Corp., Norwalk, CA

10:30 a.m. - 11:00 a.m. 232 & 234 [150] AI057
CARE AND FEEDING OF MATURE EXPERT SYSTEMS
Doug Fulrath, Digital Equipment Corp., Hudson, MA

11:00 a.m. - 11:30 a.m. 232 & 234 [150] AI068
MANAGING EXPERT SYSTEMS PROGRAM DEVELOPMENT
Doug Fulrath, Digital Equip. Corp., Intelligent Systems Technology, Hudson, MA

11:30 a.m. - 12:30 p.m. 252 - 256 [100] AI079
LEARNING LISP AS A COMPANY
Tom Hempel, Lucid, Inc., Menlo Park, CA

12:30 p.m. - 1:30 p.m. 252 - 256 [100] AI031
ORGANIZATIONAL AND OTHER MANAGEMENT ISSUES FOR ARTIFICIAL INTELLIGENCE
Art Beane, Digital Equipment Corp., Hudson, MA

2:30 p.m. - 3:00 p.m. Section D [450] AI008
AN INTRODUCTION TO PROLOG
Shoba Murali, Digital Equipment Corp., Mérrimack, NH

3:00 p.m. - 4:00 p.m. Section D [450] AI014
QUINTUS PROLOG FOR VAX VMS AND ULTRIX
Shoba Murali, Digital Equipment Corp., Merrimack, NH

4:00 p.m. - 5:00 p.m. Section D [450] AI046
ARTIFICIAL INTELLIGENCE SIG WRAPUP SESSION
Cheryl Jalbert, J C C, Granville, OH

DECUS AI SIG MENTIONED IN WALL STREET ANALYSIS

The following excerpts are from a D.H. Brown Associates report titled "DEC CHALLENGES SYMBOLICS LEADERSHIP". (Reprinted courtesy of Don Brown.)

"In our view, however, Symbolics will be frustrated with its hopes to capture the mainstream end-user base. General purpose systems will dominate the market with Common LISP, related languages and associated development software becoming another set of tools in the data processing shop. Artificial intelligence will become a natural part of the computing environment. Systems such as the IBM PC/AT or DEC VAXstar, and their cousins, will become the delivery vehicles of choice at the low-end.

Depending upon the criteria employed to judge market penetration, DEC either has already, or will soon, pass Symbolics as the market leader...."

"DEC's embryonic AI marketing effort has exploded over the latest year. Reliable trade sources suggest that DEC has signed around 1,500 licenses for its Common LISP. Possibly half of the total licenses went to production oriented/commercial research shops, and the other half may have gone to universities at steep discounts. Over two thirds of the total licenses have been written in the latest year. Each license may serve 2 to 3 users, representing 3.000 to 4,000 individuals in total. In the DECUS user group, no other chapter has grown as fast as the AI group has over the latest year - ever.

DALLAS AFTER ACTION REPORT - THE NATURAL LANGUAGE MISPARSINGS PANEL

By George Humfeld

The AI SIG expanded its operation at the Dallas symposium to include its first "magic" session in the form of a Natural Language Misparsings Panel. At 41 attendees this session was reasonably well attended, considering that it started at 8:30 PM and was scheduled opposite several other magic sessions, including the ever-popular VAX magic.

Panelists were Joseph Winograd of (JW) Artificial Intelligence Corp., developers and distributors of INTELLECT, a natural language front end for data bases; Herbert Stahlke (HS), a linguist and professor of English at Ball State University; and David Slater (DS) of Computer Sciences Corporation, a professor at University of Maryland, Baltimore Campus.

The following is a summary of the transcript of the session. Allow me to apologize in advance if you are improperly identified or if your name or organizational affiliation is misspelled. My only defense is that I am working from the tape of the session.

DS: 9 parsings available for sign on the boardwalk: WARNING SPLINTERS WEAR SHOES. E.g., (1) Splinters don't dress, thus don't wear shoes; (2) People don't normally drive on the boardwalk, so this is not a warning regarding potential wear on break shoes.

DS: REGAN'S CHINA CRISIS. This one requires knowledge of a particular historical event to properly parse. Without it, every semantic clue leads to the wrong interpretation.

HS: SYLVIA ROLLED UP THE CARPET. Illustrates a structural ambiguity rather than a semantic ambiguity. E.g., (1) Sylvia may work for Bekins; (2) or she laid down at one of the carpet and rolled to the other. Ambiguity arises in how you parse it. If "up" is taken as a preposition, then Sylvia laid down at one end of the carpet and rolled over and over until she got to the other end. [DS: I thought she filled it with rolls.] That is a semantic ambiguity based on the other parsing. She rolled it up into a cylindar, or you can verb a noun and she filled it up with rolls. Both have the same parsing.

HS: Many delightful puns result from misparsings. As a linguist I sometimes forget to look at spelling and instead look at sound. Consider the one about THE TWO MAGGOTS MAKING LOVE IN DEAD EARNEST.

JW: One of my favorites is about the fellow who went swimming at the beach in Florida where there was a sign which said PRIVATE BEACH NO SWIMMING ALLOWED. When confronted by the police, he said that he had interpreted the sign to say PRIVATE BEACH? NO! SWIMMING ALLOWED.

JW: You need the structure of the real world to properly understand the next one. THE CITY COUNCIL DENIED THE DEMONSTRATORS A PERMIT BECAUSE THEY FEARED VIOLENCE vs. THE CITY COUNCIL DENIED THE DEMONSTRATORS A PERMIT BECAUSE THEY ADVOCATED VIOLENCE. The syntax is identical for these two but it is clear that the word "they" refers to different groups of people in each case.

JW: BLIND VENICIANS vs. VENICIAN BLINDS. This one was raised in connection with the INTELLECT product. INTELLECT will not handle both, but how many data bases have both? The side benefit is that INTELLECT will properly interpret a query for BLACK FEMALES or FEMALE BLACKS as the same. In the real world the confusion between blind Venicians and Veincian blinds just doesn't come up.

Chris Goddar of CINPAC: When he sees a STOP AHEAD sign he pictures himself coming to a screeching halt in front of a skull. [DS: Oh, no. They are just anti-marijuana ads.]

David Campbell of LTD Aerospace: BLACKBERRIES ARE RED WHEN THEY ARE GREEN OR BOILED WHEN THEY ARE RIPE. Difficult for both humans and computers. The key is that when blackberries are green, they are red in color. They turn red again if you pick them when they are ripe and boil them.

DS: From Naomi Sagar's book, Because of our common sense knowledge, we don't see it as ambiguous. THIS MORNING MOTHER HAD A PAIN. The computer doesn't see it as ambiguous either, but comes up with a different interpretation. [Audience comment: If you see kids as a pain in the neck, it may mean that she gave birth.] Syntactically, "morning" is an adjective which describes "mother" as in "night watchman," unless you realize that "mother" is a full-time occupation. [HS: Disagree. The way you say it forces an interpretation.] I'm talking of a written sentence here. [HS: If properly written, it will have a comma in it. Written English is a different code from spoken English. Written text is not really natural language.] Neither is spoken English. When I was in college I worked with a group of psychologists. They pointed out that the way people interpreted the two meanings of"light-house-keeping" is not by anything spoken but by where they do their eye-blinks. -- As the arguement ensued, the audience requested the whistle, but as chair I closed it with the suggestion that a computerized vision system be developed to sort it out.

Lisa Eisenberg, New York City Transit Authority: TIME FLIES LIKE AN ARROW AND FRUIT FLIES LIKE A BANANA. [DS: That was published in Scientific American as an eight verse song with three parses of each of four sentences.] Newspaper headline: BEGIN MARS PEACE TALKS. She wasn't previously aware of ongoing peace talks with Mars. Reading further she found that the reference was to Menachem Begin marring peace talks with someone else.

HS: There is a class of sentences called garden path sentences since they lead you where you don't expect to go. For example: THE RED PENCIL MARKS EASILY ERASED DISAPPEARED. You have to get to "disappeared before you know what anything before it is. "The" is OK, but you can't tell if "red" is a noun or an adjective. "Pencil" explains "red," but it may be a noun or an adjective. That can't be decided until you know whether "marks" is a verb or a noun. And when you get to "easily" it doesn't tell you, since you don't know if it modifies "marks" or "erased." You still don't know when you get to "erased;" it could be a participal or a past tense. Then when you hit "disappeared," it all makes sense. But you've got to store all that until you get to "disappeared."

David Campbell of LTD Aerospace: Have guards at the gates at work. When guards aren't available, we have TV gates to go through with turnstyles. Situation arose where two signs were put up which individually are not ambiguous, but put together... The first sign said: USE TV GATE WHEN GUARD NOT ON DUTY. Had littering problems so they put up another sign above this one which said: PLEASE USE TRASH RECEPTACLES.

Kelsey Thompson, Artificial Intelligence Corp.: Sentences which are syntactically the same but quite different semantically. For example: I HAD CHICKEN FOR DINNER TONIGHT vs. I HAD GRANDMA FOR DINNER TONIGHT. I SAW THE CONCORDE FLYING OVER NEW YORK vs. I SAW THE STATUE OF LIBERTY FLYING OVER NEW YORK.

Dan Harmon: Consider the road sign: SLOW CHILDREN.

DS: Poem by Ogden Nash ... ends with "the wisest of these proverbs is CROSS CHILDREN WALK."

Cheryl Jalbert, JCC: A member of DECUS Europe told of a session in doubt at a symposium. The sign which was meant to resolve the problem read SESSION FLIES. Did that mean that it flew out the window or that it was off and roaring? [DS: I didn't know we had "doubt" as one of our session topics.] [Unidentified: I didn't know it was a city.]

DS: Movie ad about A SMALL COUNTRY BOY AND A LARGE CITY DETECTIVE. Syntax is same in two phrases, but normal parse is different. [HS: You have two adjectives modifying "boy" but "large" modifies "city."] Actually the words are semantically and syntactically similar, but the syntactic structure differs.

Kelsey Thompson, Artificial Intelligence Corp.: Fun with natural language products which interface with data base. They generally try to strip the plurals to obtain the singular sense of the word. Consider: WHO ARE THE OLD TIMERS IN ACCOUNTING? It understood "old," "accounting," "who," and all of that, but it took "timers" gave me the singular sense of "Tim" and listed all the old Tims in the accounting department. Another: WHAT WERE THE SALES FOR DEC IN 1985? It gave the sales for the company for December 1985 rather than the 1985 sales for DEC.

George Humfeld, Naval Sea Systems Command (session chair): Book of poetry by Silverstien, a favorite of my youngest, has poem something like: The pet store owner said this was a very fine ANT EATER; it turned out to be an AUNT EATER and now my uncle is mad.

Lisa Eisenberg, New York City Transit Authority: Reading a magazine. Saw a letter to the editor requesting CROCHET INSTRUCTIONS FOR A FROG. It was a crafts magazine, of course.

Pam Vavra, KMS Fusion: Suggestion that the panelists judge the entries for this session and that we announce the results in the newsletter and maybe award some sort of a prize.

DS: Comment on Matt Matthews: His real name is Herbert J. Matthews, IV (the fourth or intravenous, depending on how you read it). His wife is pregnant. If they have a boy, they are going to name him after a movie (V). His father's called Herb. His grandfather is called Jerome, which is his middle name. By the way, his father hasn't had to buy lunch in the last 6 months. People have been dumping burgers on his desk every day.

HS: George, I misparsed your last comment. We still haven't heard anything from the floor.

DS: Yes, but we did hear something from the chair.

Elinor Burns, DEC: Serious problem with our own natural language query system. I'd like systems to be able to handle I LIKE RED AND GREEN APPLES. Also, I'd like it to handle I LIKE RED <pause> AND GREEN APPLES without having to say "I like red apples and green apples." [DS: I presume one of the parsings you have in mind is "I like the color red and apples which are green"?] No, "I like apples which are red and green" vs. "I like both red apples and green apples." But in the latter case I don't want to repeat "apples." When I say "or" it's not a problem. But when we talk, we say "and." [DS: One approach is a knowledge base which realizes that people generally aren't interest in red and green apples. There is a concept called "level of discourse." Only a person in natural language and an English professor would use "red and green apples" to refer to apples which are red and green. After about 5 data base queries you can find these people because they have already done some weird things and you have flagged them.]

JW: The three [tu;s] (to, too, two) in the English language poses an interesting problem in voice input.

And the session continued in a similar vein with further references to the three [tu;s], red and green apples, a nervous wreck (what lies on the floor of the ocean and shakes?--thanks, Chris), Slow Children Playing who grow up to become Slow Men Working, and so on.

As you can see, we had a good time and learned something, too. Perhaps our next magic session will have subsessions for LISP magic, OPS5 magic, and so forth. Save your ideas and bring them to San Francisco!

AI, L & T, UNISIG SUITE REPORT - SPRING 1986 DECUS SYMPOSIUM

By Cheryl Jalbert

The suite shared by the AI, L&T, and UNISIGs at the Dallas symposium provided valuable function despite its distance from the sessions. The Convention Center was such a body-destroying environment that people needed to retreat when sessions ended. Without the suite the SIG leadership, Digital developers, speakers, attendees and newcomers would most likely have lost contact during non-session time periods.

The suite was open Saturday, Sunday, Monday after the sessions, Tuesday after the reception, Wednesday after dinner, Thursday after the sessions, and Friday begining about three. Being open meant being OPEN to everyone. The availability of the suite was widely advertised in our sessions and campground.

In addition to the usual advantages of being able to continue in a more relaxed atmosphere conversations on SIG-specific topics or SIG business begun earlier in sessions or the campground, there were some particularly interesting activities.

Wednesday evening was "Newcomer's Nite" in the suite. It was a time to explain what it means to be a DECUS volunteer and help individuals determine what role might be appropriate for them. AI, for example, signed up twelve much-needed volunteers that evening. It was also a time to make people feel welcome as members. L&T (whose idea it was) had a special pin for anyone who attended; about 100 people did. The majority just stopped by to say hello; but L&T did get to know several people who look like they will turn out to be excellent additions to the L&T volunteer list, filling some sorely-needed positions.

We each held meetings there Sunday following the PSSes. AI uses the Sunday meeting as an OPEN meeting to do a last minute review of sessions and session chairs. This permits the inclusion of some newcomers, discovered mostly in our PSSes, as session chairs. The AI Membership Coordinator also provides some training and instructions. For L&T and UNISIG this is also a very important opportunity to re-group before the week begins.

Saturday, the suite served as an important point of contact for all the SIGs. In addition, AI held a very special meeting there. Because AI'S status was in jeopardy when the newsletter deadline for April rolled around and because we had wanted to try meeting Saturday before symposia as a way of expanding leadership involvement beyond the group which we may have at woods meetings, we published a notice in the April newsletter announcing an open meeting beginning in the suite at 3 to discuss SIGnificant issues. We anticipated that most people's reservations were made and this time, with our late notice, not many could make it. Indeed, some people who had been at the woods meeting in March were late arriving. However, we did draw one totally new member and several Steering Committee members who were not able to attend the woods meeting. We were able to calmly discuss our tenuous SIG status and make some major new plans. Our task force approach to solving problems and incorporating new leadership began here.

There was a difficulty with the suite this time. Tuesday after the reception, we must have been the "only game in town." We were inundated. With the press of people, the noise level, and the tenuous nature of some people's involvement with our SIGs or topics, the value of the event receded sharply. It is the only time we've had doubts about our open door policy. We are attempting to analyze why we got mobbed and how we can make a similar situation more valuable in the future.

Tuesday was not the only night with an overflow. Wednesday with the effort to talk in depth with potential steering committee members, Sunday with three meetings to hold, and several lesser times we expanded into the adjoining bedrooms. Jim Livingston, outgoing UNISIG Chair, and Chris Goddard, AI Membership Coordinator/Suite Coordinator deserve a big vote of thanks for their patience in absorbing the overflow into their (non-DECUS-funded) bedrooms.

A CONCISE GLOSSARY OF AI TERMS, DEFINITIONS AND BUZZWORDS

By Terry C. Shannon

ACTIVE VALUE

A procedure invoked when data values are changed. The current state of a system variable.

ADMISSABLE

A term used to describe a HEURISTIC SEARCH ALGORITHM that will always terminate in an optimal path to a GOAL.

AGENDA

An ordered list of actions by which some KBS store information and reason about actions.

ALGORITHM

A predefined, systematic procedure that, if followed, guarantees a correct outcome. This strategy is used in conventional computer programming.

ART

An acronym and trade name for Inference Corporation's Automated Reasoning Tool, a powerful KBS shell that runs on DEC VAX processors and Symbolics LISP machines.

ARTIFICIAL INTELLIGENCE

The subfield of computer science concerned with making computers behave intelligently. A set of programming techniques that imitate human decision making processes or produce results similar to these processes.

ATOM

The basic building block of LISP programs.

ATTRIBUTE

A property of an object. Attributes are stored in SLOTS in FRAME-BASED knowledge representation systems.

AUGUMENTED TRANSITION NETWORK

A grammar representation used by natural language processing systems to parse input. Input text is compared with ATNs, or strings that represent legal parts of speech, to determine the validity of the text. Actions to be performed on or with the input text may be associated with an ATN.

AUTOMATIC PROGRAMMING

Programs that write programs. A specialized computer program that uses AI techniques to convert an algorithm or a nonalgorithmic program specification, sometimes expressed in natural language, into programming language source statements.

BACKTRACKING

A search technique in which nodes are examined in a depth-first manner until until nodes at lower levels no longer can be searched. At this point, the search algorithm backs up the data tree to the last successful fork and resumes its downward search along another line. This search strategy is used by the PROLOG programming language and is best suited to diagnostic, or GOAL-DRIVEN problems.

BACKWARD CHAINING

Solving a problem by stating a goal and looking in the database for conditions that would cause the goal to be realized. Each time a satisfactory condition is found, it becomes the new goal and the search is reinitiated to find that goal's preconditions.

BAYESIAN PROBABILITIES

Probabilities based on the theory of conditional probability invented by Thomas Bayes, a British probabilist. Bayesian probabilities can are used to reason about uncertainty in KBSs. See PROBABILITY.

BEST-FIRST SEARCH

A search strategy in which the search works from the goal to be proven back through the conditions that would result in proving that goal. See BACKWARD CHAINING.

BLACKBOARD ARCHITECTURE

A KBS technique in which several independent knowledge bases each examine a common working memory or data structure called a "blackboard." An agenda-based control system continually examines all the possible pending actions on the blackboard and selects the one to try next.

BREADTH-FIRST SEARCH

A search strategy in which each node at a given level of a tree is examined before the search mechanism proceeds to deeper levels.

BRANCHING FACTOR

The average number of branches per node or fork in a tree search.

CERTAINTY

The degree of confidence one has in a fact or relationship. Contrast with PROBABILITY, or the likelihood that an event will occur.

CERTAINTY FACTOR

A number or percentage (numerical weight) that indicates the degree of certainty that a rule or fact is valid. Similar to the "70% probability of rain" technique used by weather forecasters. These numerical weights work differently and less formally than probablity coefficients. Most KBSes use certainty factors rather than probabilities.

CHUNK

A collection of facts which are stored and retrieved as a single unit. The limitations of working memory are usually expressed as the number of chunks that can be handled simultaneously.

COMMON LISP

A standardized LISP dialect that is meant to be a starting point or base language definition. Like ADA, COMMON LISP is a government inspired language standard.

COMBINATORIAL EXPLOSION

A situation that can arise when a problem is characterized by numerous possible outcomes. The combinatorial explosion is one of the stumbling blocks faced by developers of large KBSes. The likelihood of combinatorial explosion can often be reduced by efficient SEARCH STRATEGIES or PRUNING. See COMBINATORICS, below.

COMBINATORICS

A branch of mathematics dealing with combinatorial problems that arise from explosive growth of trees due to large numbers of combinations of events and outcomes. For example, a chess game has over 10 (120) possible outcomes when all possible moves are considered.

COMPETENT SYSTEM

A more appropriate name for an expert system. A class of KBS whose knowledge base is derived from sources who are not necessarily experts.

COMPILED KNOWLEDGE

Knowledge which people arrange and store as chunks and networks. Knowledge compiled into more and more abstract and theoretical patterns is referred to as DEEP KNOWLEDGE. Knowledge acquired by practical experience is called SURFACE knowledge. Most people begin by acquiring DEEP KNOWLEDGE which they later associate with the SURFACE KNOWLEDGE they obtain through experience. This information is subsequently recompiled into practical knowledge or HEURISTICS.

CONDITION

The part of a rule that is tested to determine whether the rule should be executed. Also called the left hand side (LHS) of a rule or production.

CONFIDENCE FACTOR

See CERTAINTY FACTOR.

CONFLICT RESOLUTION

Resolving the differences among rules in situations where more than one rule can be applied. The second phase of the OPS5 recognize-act cycle.

CONNECTED-SPEECH SYSTEM

A system that can understand a continuous stream of speech in which the speaker does not pause between words to emphasize their beginnings and endings. Contrast with ISOLATED-WORD SYSTEM.

CONSTRAINT

A limit imposed on search space.

CONSULTATION PARADIGM

A generic problem-solving scenario used by a KBS building tool or SHELL. Most such tools employ a DIAGNOSTIC/PRESCRIPTIVE paradigm.

CONTROL

In a KBS, the method used by the INFERENCE ENGINE to determine the order in which reasoning takes place. BACKWARD CHAINING, FORWARD CHAINING and BLACKBOARD AGENDAS are all examples of KBS control strategies.

DARPA

The Defense Advanced Research Projects Agency, a Department of Defense agency that has provided much of the research funding to the AI community.

DATA-DRIVEN

Searching a data tree forward from the observed data to the conclusion. Also referred to as INDUCTIVE SEARCH or FORWARD CHAINING.

DEDUCTION

The process of reaching a conclusion by logical means.

DEEP KNOWLEDGE

Knowledge of theories, axioms and underlying facts about a domain. Contrast with SURFACE KNOWLEDGE.

DEPTH-FIRST SEARCH

A search technique in which all nodes along a single path at all levels in a tree are examined before the next path is searched. When the lowest node on a path is reached, the search mechanism uses BACKTRACKING to return to the last successful fork in the data tree.

DEMON

A procedure activated by changing or accessing the values in a database.

DIAGNOSTIC/PRESCRIPTIVE CONSULTATION PARADIGM

A KBS paradigm that requires the user to identify the symptoms or characteristics of a problem in order to determine which of several alternative solutions is most appropriate.

DOCUMENT UNDERSTANDING SYSTEM

A system that accepts text input, stores and classifies it, then uses it to fulfill the requirements of a task. Some systems paraphrase input text, some answer questions, some draw inferences and others perform translation.

DOMAIN

A topical area or region of knowledge. The area of expertise practiced by an expert. For example, a doctor's domain is the field of medicine. Today's KBSes can provide competent advice only within very narrowly defined domains.

DOMAIN EXPERT

The person from whom the knowledge is extracted or derived when building the knowledge base of a KBS.

ELLIPSIS

The omission of words so that a sentence is not grammatically complete but still can be understood by its audience. The way people talk as opposed to the way write.

EXHAUSTIVE SEARCH

A search strategy in which the entire set of possible outcomes is examined for a solution. This technique usually leads to COMBINATORIAL EXPLOSION.

EXPERIMENTAL KNOWLEDGE

Knowledge gained from "hands-on" experience, typically specific facts and rules of thumb or heuristics. Contrast with DEEP KNOWLEDGE.

EXPERT SYSTEM

A computer program that solves problems or achieves goals by manipulating knowledge derived from expert sources rather than by following a predefined algorithm. Also referred to as a COMPETENT SYSTEM and, more appropriately, a KNOWLEDGE BASED SYSTEM (KBS). Referred to in the popular press as almost anything. When used by mendacious software vendors, refers to any computer program that costs much more than it should.

EXPERTISE

The skill and knowledge possessed by some people that permits performance far above the norm. Often consists of massive amounts of raw information combined with heuristics, simplifications, rare facts and proven procedures.

EXPLANATION FACILITY

The part of a KBS that explains how conclusions were reached by citing the information processed during problem evaluation. A simple explanation facility traces the rules used to arrive at a solution, more complex facilities supply encoded reasons and references for the reason one solution was chosen over its alternatives.

EXTENSIBILITY

A programming language's ability to be enhanced or tailored by a user. If a procedure is not provided by default in an extensible language, the user can write the procedure using statements available in the language, then integrate the extension into the language itself. Extensibility is one of the most valuable features of the LISP programming language.

FACT

A statement whose validity is generally accepted. In a KBS, a fact is an ATTRIBUTE and a specific associated value.

FIFTH-GENERATION COMPUTER

An evolving class of computers characterized by great speed, parallelism, symbolic data manipulation and a natural language interface. A catchall term used to describe the high-performance non-Von Neumann computers currently being developed here and abroad.

FIRE

The execution or instantiation of a RULE in a PRODUCTION SYSTEM.

FLAVORS

An OBJECT-ORIENTED programming feature found on AI development processors such as the LISP MACHINE from LMI and Symbolics. VAX FLAVORS is available as an extension to VAX LISP.

FORWARD CHAINING

The type of activity done in a system that applies operators to a current state in order to produce a new state, and so on until a solution is reached.

FORTH

A recursive, linked-list programming language invented by Charles Moore. FORTH features rapid execution, extensibility and portability, and is has seen use as a KBS development language.

FRAME

A knowledge representation method used to organize the knowledge base and to provide inheritance features to other frames. OBJECTS and their ATTRIBUTES fit into SLOTS within a frame. Useful for reducing duplication of objects and attributes.

FUZZY LOGIC

An approach to representing uncertainty and approximate reasoning in a KBS. Based on a theory invented by Lofti Zadeh, fuzzy logic lets programmers assign numerical values or weights to "gray areas," such as the height of an individual, that cannot be defined by yes-or-no terms like "TALL" and "NOT TALL."

FUZZY SET

Used in FUZZY LOGIC to represent sets of outcomes whose precise values are only vaguely known.

GOAL

The solution a system attempts to reach. Sometimes, in order to reach the goal, subgoals must first be achieved.

GOAL DRIVEN

Same as BACKWARD CHAINING.

GRANULARITY

The level of detail in a frame or rule. Fine granularity implies more detail than coarse granularity.

GRAPH

A data structuring and modeling device consisting of NODES and BRANCHES. A TREE is a special type of graph.

HEURISTIC

A process, sometimes called an educated guess or a rule of thumb, that may help in the solution of a problem. A heuristic does not guarantee an optimal solution or, for that matter, any solution at all.

HIERARCHY

A ordered network of concepts, events, objects or facts in which some are subordinate to others. Hierarchies occur in databases, societies, organizational charts and KBSes. In general, hierarchies imply INHERITANCE, so objects higher in the organizational network contain the attributes associated with their subordinate events or objects.

HIMIKO

The name chosen by Japanese researchers for their parallel PROLOG programming language.

HORN CLAUSE

In logic programming, an expression connected with an "OR" that has at most one positive proposition. Logic programming is made more efficient by restricting assertions to Horn clauses, much as production systems reqeire rules to be stated as IF-THEN constructs.

ICON

A graphic symbol to which a computer user can point an interface device, such as a mouse, to perform a system function. Icons were first used in AI applications and now are found in computers like the Apple Macintosh.

IF-THEN RULE

A statement of a relationship between a set of facts. IF-THEN rules may be definitional or heuristic and serve as the key elements, or PRODUCTIONS, in PRODUCTION SYSTEMS.

IMAGE PROCESSING

The examination by computer of a digitized data about a scene and the features in it in order to extract information.

INDUCTION SYSTEM

A KBS whose knowledge base consists of examples. An induction algorithm builds a decision tree from the examples and the KBS uses this tree to provide advice.

INFERENCE

The process by which new facts are derived from known facts. A rule, combined with a new rule and a known fact, results in a new fact.

INFERENCE CHAIN

The sequence of rules applied by a KBS.

INFERENCE ENGINE

The interpreter in a KBS. The portion of a KBS that contains the inference and control strategies. Inference engines are characterized by the inference and control strategies they employ, such as FORWARD CHAINING and BACKWARD CHAINING.

INFERENCE NET

The network of all possible inference chains that might be searched during a KBS consultation session.

INHERITANCE

A process by which characteristics of one object are assumed to be characteristics of another. For example, if we determine that an animal is a mammal, we automatically can assume that it has all of the characteristics of mammals.

IPL II

Information processing language, a precursor to LISP developed in 1955 by Newell, Shaw and Simon.

INSTANTIATION

The firing of a rule in a production system. The specification of particular values. A specific person with a particular ailment is an instantiation of the generic object "patient."

INTELLIGENT SYSTEM

A system equipped with a knowledge base that can be manipulated in order to make inferences.

INTERFACE

The auxiliary computer programs through which the user interacts with a program or KBS.

INTERLISP

A LISP dialect characterized by a programming environment that facilitates program development. INTERLISP includes such features as automatic parenthesis matching and a "DWIM" (Do What I Mean) editor that attempts to correct spelling errors.

ISOLATED-WORD SYSTEM

A speech understanding system whose input must consist of words enunciated separately, instead of run together, to make word identification easier for the system. Contrast with CONNECTED-SPEECH SYSTEM

KBS

Knowledge based system. A computer program characterized by symbolic, rather than numeric, processing. Also known as an expert system.

KNOWLEDGE

An integrated collection of facts and relationships which, when appropriately exercised, produce competent performance. The quantity and quality of the knowledge possessed by a person or a computer program can be judged by the number and variety of situations in which the person or program can obtain successful results.

KNOWLEDGE ACQUISITION

The process of locating, extracting and refining knowledge from a variety of sources such as human experts and documented research. The material gleaned during knowledge acquisition must be converted into a form that can be used by a computer program.

KNOWLEDGE BASE

The portion of a KBS that contains the rules, facts and heuristics about a PROBLEM DOMAIN.

KNOWLEDGE ENGINEER

A person whose specialty is assessing problems, acquiring knowledge and building KBSes. Knowledge engineers must be skilled in various aspects of cognitive psychology, computer science and AI techology.

KNOWLEDGE STRUCTURING

Reduction of a set of skills from a problem solving domain to a representation that may be programmed on a computer.

KNOWLEDGE REPRESENTATION

The method used to encode and store rules, facts and relationships in a knowledge base. Popular knowledge representation paradigms include FRAMES, SEMANTIC NETWORKS, OBJECT-ATTRIBUTE-VALUE TRIPLETS and PRODUCTION RULES.

LIPS

Logical Inferences Per Second. A measure of the inferencing speed of a computer in symbol processing applications. One LIP is equal to one PROLOG procedure call or approximately 50-100 traditional machine language instructions.

LISP

An acronym for LISt Processing language. A list-based, user-extensible computer programming language invented in 1957 by MIT AI reseacher John McCarthy. LISP is the most popular AI development language in the United States.

LISP MACHINE

A specialized computer whose architecture and large addresses are designed for rapid LISP program execution. Most LISP machines provide a sophisticated graphics interface and other facilities that are useful in AI program development. Likewise, most LISP machines are standalone, single-user systems that are not directly connected to conventional computers and their databases. This limits their usefulness as application delivery systems.

LIST

A data structure in the LISP programming language.

LOGIC

A system that describes rules for manipulating symbols. Common logic systems include propositional and PREDICATE CALCULUS.

LOGICAL CONNECTIVES

Special symbols from propositional logic and the predicate calculus to indicate logical relationships among statements.

MACHINE LEARNING

The ability of a computer program to learn from experience and deduce outcomes from information that it does not explicitly possess.

MACHINE TRANSLATION

The translation by computer of text in one native language into text in another native language.

MACLISP

A LISP dialect that is optimized for efficiency rather than development. Contrast with INTERLISP.

MCC

An acronym for the Microelectronics and Computer Technology Corporation, a nonprofit collaborative venture formed in 1982 by major American computer and electronics firms.

METAKNOWLEDGE

Background information or knowledge about knowledge. Information that tells a system what it knows, what it can do with what it knows and what the limits of its knowledge are.

MODUS PONENS

A basic rule of logic that asserts IF a condition is true, THEN an action is true. For example, IF we know that A implies B and we know that A is in fact true, THEN we can assume B. This premise/conclusion technique, which can be modeled by tree structures, is the most common inferencing logic used in KBS.

MONOTONIC REASONING

A reasoning system based on the assumption that once a fact becomes true, it remains unalterably true throughout the remainder of the reasoning process. Contrast with NONMONOTONIC reasoning.

NATURAL LANGUAGE

A person's native tongue.

NONMONOTONIC REASONING

A reasoning system in which facts that become true can again become false if some value changes during a session. For example, the truth of the statement "the sky is blue" depends on the time of day and the prevailing weather. Nonmonotonic reasoning is more difficult to implement than MONOTONIC reasoning, but is more appropriate for situations in which values change widely over a short period of time.

NSA

Acronym for the National Security Agency, a federal intelligence agency concerned with signal processing, natural language translation and advanced computer hardware.

OBJECT

In a FRAME-BASED knowledge representation strategy, a physical or conceptual entity that has many ATTRIBUTES. When attributes or rules are divided into groups, each group is arranged around an OBJECT. See OBJECT-ATTRIBUTE-VALUE TRIPLET, below.

OBJECT-ATTRIBUTE-VALUE TRIPLET

A method of representing factual knowledge in which an OBJECT is a physical or conceptual entity. ATTRIBUTES are properties (such as weight or cost) associated with OBJECTS, and VALUES are the numeric quantities (such as 100 pounds or $50.00) associated with ATTRIBUTES.

OBJECT-ORIENTED PROGRAMMING

Programming that focuses on objects rather than procedures, a key technique for knowledge manipulation.

PARALLEL PROCESSING

The ability of a computer to carry out more than one instruction at the same time instead of carrying them out sequentially. Contrast with SERIAL PROCESSING.

PARSING

Breaking down and identifying the components of language statements as various parts of speech.

PENETRANCE

In tree searching, the length in number of nodes of the minimal path from the beginning state to the solution state divided by the number of nodes generated to get the solution. The higher the penetrance value, the more direct and efficient the search is.

PIPELINED PROCESSING

A technique in which computer instructions are overlapped so more work can be done per unit of time or machine cycle. Processors like the VAX 8800 use pipelined processing to perform operations on specific portions of four different instructions during the same machine cycle. This "assembly line" technique helps wring additional performance from conventional Von Neumann computers.

PREDICATE LOGIC

A type of logic, based on predicate calculus, that can be used in knowledge based programs. Each basic unit in predicate calculus is called an OBJECT. Statements about OBJECTS are called PREDICATES. The PROLOG programming language is based on predicate logic.

PROBABILITY

A variety of approaches to statistical inference that can be used to determine and quantify the likelihood of a particular event or relationship. Most KBSes use CONFIDENCE FACTORS instead of probability, but some use a modified version of BAYESIAN PROBABILITY to calculate the likelihood of various outcomes.

PROBLEM SOLVING

The process in which one starts from an initial state and searches through a problem space to identify the sequence of operations that will lead to a desired goal. Successful problem solving depends on a variety of factors, including knowing the initial state, knowing what constitutes an acceptable outcome and knowing the elements and operators that define the boundaries of the problem space. If the elements and operators are large in number or poorly defined, a huge or unbounded problem space results, causing a COMBINATORIAL EXPLOSION that makes EXHAUSTIVE SEARCH impossible.

PRODUCTION RULE

A condition-action or situation-response rule that is usually stated in an IF-THEN format. Each such rule, when executed, results in new conditions and new actions, making it possible for subsequent, different rules to be executed.

PRODUCTION SYSTEM

A computer program composed of IF-THEN statements or production rules. The basis of the OPS5 and OPS83 programming languages. In a production system, rules are selected for execution by matching their patterns against the contents of WORKING MEMORY. When a match is found, the rule is FIRED or INSTANTIATED and its consequence is invoked.

PROLOG

A logic-based programming language based on PREDICATE CALCULUS. PROLOG features a built-in inference strategy, backward chaining and a highly readable English-like syntax. Invented by Alain Colmerauer in 1972, PROLOG enjoys widespread use by AI programmers in Europe and Japan.

PROTOCOL

A set of rules for accomplishing an activity.

PROTOTYPING

An evolutionary technique, used in the development of a KBS, that involves continuous interaction with and frequent review by the domain expert as a KBS evolves from concept to finished program. This differs from conventional programming where detailed plans are made before coding actually begins.

PRUNING

A term used to describe the result of constraining a tree search by eliminating irrelevant branches of the tree from the search pattern.

PSI

An acronynm for Personal Sequential Inference machine, an AI workstation prototype designed by the Japanese during the first phase of their FGCS program.

QUERY

In PROLOG, a pattern or template which is used to find possible values for a given variable, or to determine the truth of a statement.

RECURSION

The process of a routine or function calling itself. LISP is a recursive programming language.

REPRESENTATION

The way in which knowledge is stored in a KBS.

RESOLUTION

A logical inferencing form from the predicate calculus that allows reduction or "resolution" of two statements into one.

RESOLUTION REFUTATION

A method for getting an answer from resolution based inferencing as used in PROLOG.

RULE

An IF-THEN statement in a KBS. Same as a PRODUCTION.

SATISFICING

The concept of finding a satisfactory solution to a problem as opposed to an optimal solution. Satisficing is frequently used by AI programs--and people--to limit searches and reduce the likelihood of COMBINATORIAL EXPLOSION.

SCHEMA

A data representation technique similar to FRAMES.

SCRIPT

A knowledge representation strategy based on predefined, stereotyped situations. When a specific activity fits into a script, a script-based system can predict other likely events by analogy. For example, once a person has dined at a McDonalds, he develops a "fast-food" script. If the person later eats at a Burger King, he does so almost automatically by following the protocol defined in the fast-food script.

SEARCH

The process of trying different actions until a goal is achieved. The process of elimination inherent in most AI and knowledge-based applications.

SEARCH SPACE

The set of possible outcomes for the problem presented. A large search space can result in COMBINATORIAL EXPLOSION.

SEMANTIC NET

A tool borrowed from cognitive psychology. A knowledge representation paridigm based on nodes that describe objects and links that describe relationships between nodes. Nodes are usually physical objects, but may be physical abstractions. Links may categorize (IS-A) or imply ownership (HAS-A). Once an attribute has been associated with a node, that attribute can be passed to subsequent, downstream nodes through a process called inheritance.

SEMANTICS

The meaning of words within context. The principles through which we know the differences between "Mickey Mantle threw a ball" and "Princess Diana threw a ball," or "time flies like an arrow" and "fruit flies like a banana."

SHELL

A KBS building tool that contains an INFERENCE ENGINE and an empty KNOWLEDGE base. Many sophisticated shells provide editors, explanation facilities and advanced graphics systems. A user fills the shell with the appropriate rules and facts for the selected problem domain. The same shell can be used as the processor for many different KBSes.

SLOT

The storage area in a FRAME associated with an object's ATTRIBUTES.

SPEAKER-DEPENDENT SYSTEM

A speech recognition system that can be trained to recognize words spoken by a specific individual.

SPEECH GENERATION SYSTEM

A system that transforms text into audible speech with correct pronunciation. DECtalk is a simple speech generation system.

SPEECH UNDERSTANDING SYSTEM

A system that converts the digitized signals of audible speech into printed text.

SYNTAX

The structure into which words fit.

TURING TEST

An early test for machine intelligence devised by British mathematician Alan Turing. In this test, a person is placed in a room with a computer terminal that interacts either with a computer or another person. If the person is unable to determine whether he is interacting with the person or the computer, the computer is regarded as intelligent.

UNIFICATION

A term used in PROLOG to indicate the matching of patterns.

VISION SYSTEM

A system that digitizes and interprets input about the size, shape, color and location of an object to determine what it represents or what its important features are.

VLSI

An acronym for Very Large Scale Integration, the development of extremely dense, complex electronic circuits that can be placed on small silicon chips.

WORKING MEMORY

The dynamic, changing portion of a production system. Working memory contains the changing database of a production system as rules fire.

READER INPUT:

I would like to hear from anyone with information about AI applications that run on the Rainbow, PRO-350 or other DEC personals.

Ev Batey
Santa Barbara PC LUG Chair
364 Fairview Dr.
Ventura, CA 93001
(BATEY) on DCS, or via FIDO 102/701
Voice days: (805) 982-5881
VOICE pm's: (805) 656-1504.

DECUS

BA

BUSINESS APPLICATIONS

BUSINESS APPLICATIONS SIC STEERING COMMITTEE

Chairman
Stuart Lewis
Douglas Furniture
Bedford Park, IL

Symposium Coordinator
Steve Simek
IRT Corporation
San Diego, CA

Asst. Symposium Coordinator
Bobbie Wiley
CEI Perry Nuclear Power Plant
Euclid, OH

LRP and Marketing Coordinator
Arnold I. Epstein
D-M Computer Consultants
Rolling Meadows, IL

Marketing Asst.
George Dyer
Gallaudet College
Washington, DC

Communications Representative
Steven Lacativa
Price Waterhouse
New York City, NY

Newsletter Editor
Thomas Byrne
L. Karp and Sons
Elk Grove Village, IL

Session Notes Editor
Raymond Swartz
Goodyear Tire and Rubber Co.
Akron, OH

Library Representative
David Hittner
Projects Unlimited
Dayton, OH

CL SIG Liaison
Becky Burkes
Financial Insurance Consultants
Covington, LA

DMS SIG Liaison
Joe Sciuto
Army Research Institute
Alexandria, VA

Members-at-Large
Robert D. Lazenby
Dixie Beer Dist., Inc.
Louisville, KY

Robert Kayne
Gallaudet College
Washington, DC

Ray Evanson
Paragon Data Systems
Winona, MN

Digital Counterparts
Sue Yarger
Digital Equipment Corporation
Merrimack, NH

Ray Arsenault
Digital Equipment Corporation
Merrimack, NH

SIC Mentor
Bill Brindley
Networks SIG Chair

SIC Review Committee
Larry Jasmann
Leslie Maltz
Ted Bear
Jeff Killeen

BUSINESS APPLICATION SIG

FALL 1986 DECUS SYMPOSIUM SESSIONS

In order to help DECUS members better prepare for attendance at the San Francisco Symposium, the Business Applications SIG is providing a complete list of scheduled Business Applications sessions. We refer you to the symposium Sessions at a Glance in your Preliminary Program and Registration Kit for the date, time and location of each session.

A. SIG Organization Activities

1. BA005 - Business Applications Roadmap

   What, where, when, and why of BASIG symposium activities

2. BA012 - Business Applications Clinic and Get-Together

   Experts provide and informal problem-solving clinic - cash bar and hors d'ouvres

3. BA013 - Business Applications SIG Business Meeting

   Chance to get involved - planning for Nashville symposium - review and critique of the week's activities in San Francisco

4. SIG Hospitality Suite

   *The BASIG Suite will be open every night of the week. For exact times and suite location attend the Roadmap session and watch each issue of UPDATE*DAILY*

B. Computer Integrated Manufacturing

1. BA016 - Should You Jump on the MRP and JIT Bandwagon

   Theory vs practice - management commitment, personnel training, typical shop floor changes, common pitfalls, applicability to differing manufacturing environments

2. BA017 - Addressing the True Software Needs of the Small- to Medium-sized Manufacturer

   Looks at small manufacturers and job shops

3. BA040 - Implementing Advanced Manufacturing Systems in Businesses with Complex Product Engineering Requirements

   How to establish personnel responsibility; using MRP II in a make-to-order environment; advantages of using outside consultants

4. BA027 - Managing Product Engineering Data Flow with EDCS

   Managing and communicating company-wide engineering data with the Engineering Data Control System

C. Application Development and Integration

1. BA004 - Applications Software Design for the Multilingual Environment

   Designing business applications software that runs the same executables in different languages at the same time

2. BA007 - Reduce Development Time and Make Users Happy with Prototype Programming Methods

   What it is

3. BA011 - Management Concerns in Integrating VAX Applications

   Structuring organization management information via integration of PC's, mini's, mainframes, and office and business applications

4. BA022 - VMS Software Productivity Tools Workshop

   Overview of DEC's programmer productivity tools, ways to measure programmer productivity, small group workshops to identify and develop solutions - a unique opportunity for one on one consulting with experts

5. BA024 - Satisfying Business Applications Using 4th Generation Languages

   A how-to-do-it workshop featuring experienced users

6. BA034 - Using the SMG Facility on VAX/VMS

   How to write terminal independent BASIC programs using SMG, general menu system example, do's and don'ts

7. BA043 - Benefits of an Integrated System

   What is application integration, how others have done it with A-to-Z, DEC product demo

8. BA051 - Virtual Applications Engines - Added Value to Multi-User Systems

   How to transport applications across hardware and operating systems without a headache

9. BA045 - Application Development Clinic Using 4th Generation Language Tools

   Participate in developing a working system using the A-to-Z Applications Generator and Database Manager on RMS files

10. V33 - Vax Performance Monitoring and Capacity Planning Tools

   A workshop designed for the MIS manager which provide another unique opportunity for one on one consulting with experts - sponsored by the VAX SIG and recommended by BASIG

D. **Communications**

1. BA008 - Digital's Solutions for Telcom Managers

   DEC product overview of PBX/Facilities Management software for the VAX

2. BA050 - Asynchronous Communications - Methods and Applications

   Asynchronous vs WAN and LAN - sometime asynch is better

E. **Solutions**

1. BA001 - Digital's Small Business Accounting

Product Strategy

Overview of DEC and third party products

2. BA002 - Update on the Digital Accounting System

   Where it is plus new features and services

3. BA036 - Digital's Solutions for Finance and Business Operations

   What is the current state of the art in spread sheet and financial modeling technology? How important are operational consistency and accuracy? Attend and find out all about it.

4. BA020 - Considerations for the Selection and Implementation of Financial Management and Accounting Software Packages

   An experienced user tells all

5. BA003 - Announcing the Digital Construction Package

   Product announcement for general and subcontractors and anyone else who needs to manage construction jobs

6. BA006 - Management Considerations in Word Processor Selection

   Save yourself some grief

7. BA009 - Application Software for Digital Systems

   A look at everything DEC sells - both theirs and third party

8. BA018 - Laser Printer Applications for Business

   If you bought one, learn what to do with it

9. BA023 - Creating Videotex Business Solutions

   VAX VTX and VAX VALU in the real world of business

10. BA025 - Future Trends in Printing

    What do I do with my printing equipment when the paperless office is knocking on my door?

11. BA030 - Overview of SPICE - a Poor Man's FMS

    Works on RSX and VMS, resembles FMS and Datatrieve, and interfaces terminals and programs with RMS indexed files.

12. BA031 - The VAX Solution for Economic Surveys

    The Bureau of the Census tells us how they did it on a VAX from start to finish

13. BA032 - Digital's Solutions for Human Resource or Personnel Departments

    DEC shows how users are meeting their human resource management needs in innovative and useful ways

14. BA035 - Digital's Solutions for Sales and Marketing Departments

    DEC discusses sales and marketing department needs for information access, organizational communication and operational tools and shows how these needs are being met today and may be met in the future

15. BA049 - PC's vs Terminals - Making the Decision

    Recognizing and dealing with the symptoms and problems of PC's that won't do the job anymore

16. BA048 - A to Z - State of the Product

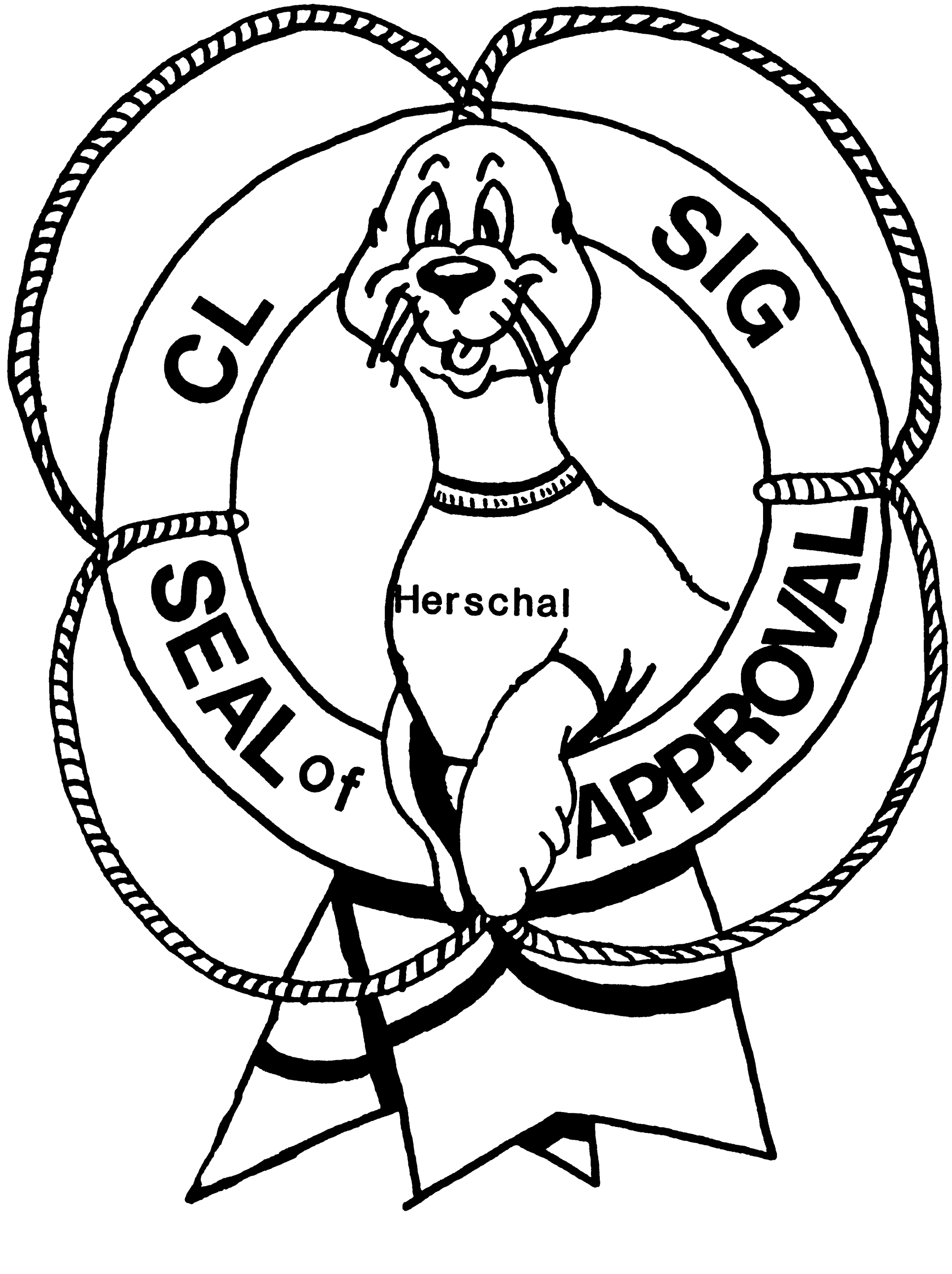
    What is this A-to-Z thing that controls the user environment, integrates software, and does application generation and database management on RMS files? Come and see.

17. BA047 - Bridging the PDP to VAX Gap with A to Z

    Product panel featuring DEC and third party developers who will discuss using A-to-Z to migrate from PDP to VAX without grief or tears

18. BA019 - A Generic User Interface

    A user shows how to set up a controlled user environment on the VAX and go home on time every night



COBOL BASIC DIBOL RPG

COMMERCIAL LANGUAGES SIG

Chairperson
Dena Shelton
Systems Industries
Milpitas, CA

Symposium Coordinator
Ray Strackbein
Palm Desert, CA

Library Coordinator
Philip Hunt
System Industries
Milpitas, CA

Session Note Editor
Bob Van Keuren
Userware International, Inc.
Escondido, CA

Newsletter Editor
Ted A. Bear
Ramtek
Santa Clara, CA

Ass't Newsletter Editors
Beverly Welborne
LaPorte, IN

Kevin Cullen
VITA-Mix Corp.
Holmstead Falls, OH

Daniel Cook
Userware International, Inc.
Escondido, CA

BASIC Working Group Members
Mark Hartman
Jadtec Computer Group
Orange, CA

Rocky Hayden
UserWare International Inc.
Escondido, CA

Bill Tabor
Computer Productss
Pompano Beach, FL

Ted A. Bear
Ramtek
Santa Clara, CA

COBOL Working Group Members
Keith Batzel
Crowe, Chizek & Co.
South Bend, IN

Mary Anne Feerick
RDBS Inc.
Kernersville, NC

Bill Leroy
The Software House, Inc.
Atlanta, GA

Herbert J. Matthews IV
ManTech International Corp.
Alexandria, VA

Jim Welborne
Crowe Chizek & Co.
South Bend, IN
Jim Wilson
Pizer Inc. QC Div.
Terre Haute, IN

DIBOL Working Group Members
Neil Baldridge
CompuShare
Lubbock, TX

Becky Burkes-Ham
Financial Insurance Consultant
Covington, LA

Colin Chambers
Software Ireland Rep. Inc.
Portola Valley, CA

Mark Derrick
WAAY-TV
Huntsville, AL

Gary A.P. Kohls
Milwaukee, WI

Ken Lidster
Disc
Sacramento, CA

Kenneth M. Schilling
MCBA
Montrose, CA

Marty Schultz
Omtool Inc.
Tewksbury, MA

Marty Zergiebel
The Software Gallery
Brookfield, CT

RPG Working Group Members
Keith Batzel
Crowe Chizek & Co.
South Bend, IN

Digital Counterparts
Tom Harris
Nashua, NH

Jim Totten
Nashua, NH

Joe Mulvey
Nashua, NH

Shirley Ann Stern
Nashua, NH

Standards Representatives
BASIC
Dan Esbensen
Touch Technologies, Inc.
Escondido, CA

COBOL
Bruce Gaarder
Macalester College
St. Paul, MN

DIBOL
Eli Szklanka
TEC
Newton, MA

A NEWLY ENHANCED DEBUGGER FOR THE PDP-11S

PDP-11 SYMBOLIC DEBUGGER V2.0
(formerly FORTRAN-77 DEBUG)
by
Marilyn Finch
PDP-11 Languages
Software Product Manager

PDP-11 Symbolic Debugger V2.0 is a major release providing I & D space support and an interface to COBOL-81. The name of this product has been changed to emphasize the additional language support. Now not only can the debugger assist programmers in debugging their FORTRAN-77 and MACRO-11 programs, but COBOL-81 programmers can also utilize this timesaving utility. A SET LANGUAGE command is available to allow debugging of multi-language programs.

This version of the debugger has been designed to occupy less than 4k bytes of user task space. Unlike the debugger which is bundled in the COBOL-81 distribution kit, the PDP-11 Symbolic Debugger is much smaller and faster. The PDP-11 Symbolic Debugger allows the user to refer to program locations by symbols or line numbers rather than by addresses, thus, saving valuable programmer time. This also makes the PDP-11 Symbolic Debugger an execellent replacement for ODT. Breakpoints and tracepoints may be set in overlay segments that are not currently resident. A programmer may step through a program by source line, which facilitates source debugging, or by PDP-11 instructions.

The documentation has been totally rewritten for this release. One installation guide contains the information needed to install the PDP-11 Symbolic Debugger on any of the supported operating systems. A language specific manual has also been added to assist the user to debug each specific language. A quick reference guide is provided for experienced users of the debugger highlighting proper syntax for this release. Release notes have been placed on the distribution medium to allow us the opportunity to provide you the customers with the latest possible information. The following lists the contents of the documentation set.

PDP-11 Symbolic Debugger User's Guide
PDP-11 Symbolic Debugger Quick Reference Guide
PDP-11 Symbolic Debugger Installation Guide
PDP-11 Symbolic Debugger Information for COBOL-81 Users
PDP-11 Symbolic Debugger Information for FORTRAN-77 Users
PDP-11 Symbolic Debugger Information for MACRO-11 Users

The PDP-11 Symbolic Debugger is available on RSX-11M, RSX-11M-PLUS, Micro/RSX, RSTS/E, Micro/RSTS, VMS via VAX-11 RSX, and P/OS via the Pro/Tool Kit or the Professional Host Tool Kit.

For further information on this product, contact your Digital Equipment Corporation local sales office.

COBOL-81 V2.4
by
Marilyn Finch
PDP-11 Languages
Software Product Manager

COBOL-81 V2.4 is a release that provides bug fixes, some user requested enhancements, and an interface to the PDP-11 Symbolic Debugger.

Enhancements to this version include:

FEATURE	BENEFIT/DESCRIPTION
o An UNLOCK statement	Eases development of file sharing applications allowing the programmer to unlock the last record read
o NO OTHERS clause on the OPEN statement	Allows the programmer to specify exclusive access to files in a multi-user environment
o OMITTED as a CALL statement parameter	Enables direct calling of RSX system services
o RSTS Support routines	Provides an interface to RSTS system services without using MACRO subprograms
o Conditional compilation	Individual program source code lines can be included conditionally by placing a conditional compilation select character in the program indicator area. Alternatively, all conditional compilation lines may be included by using the WITH DEBUGGING MODE clause.
o Five new extensions to the ACCEPT STATEMENT	
- PROTECTED SIZE id	Allows a numeric identifier to be used to specify the size of a protected field in an ACCEPT statement. You are now able to write independent generalized screen handling sections.
- FILLER	Enables a single character literal to fill an ACCEPT field. The ACCEPT field character positions can now be clearly marked.
- AUTOTERMINATE	Input automatically terminates when the number of characters specified have been entered for the ACCEPT field. The overhead of typing keys is reduced for data entry screens with large numbers of small ACCEPT fields.

- DEFAULT CURRENT VALUE

  Retains the value of the ACCEPT field variable if data is not entered when the ACCEPT statement executes. The programmer avoids using buffers to hold temporary data.

- NO BLANK

  The ACCEPT field will not be space filled until the user has typed at least one character or a terminator key. This allows the programmer to fill the ACCEPT field with sample/prompt information.

o PDP-11 Symbolic Debugger interface

This version of COBOL-81 also provides an interface to the PDP-11 Symbolic Debugger. This debugger has been designed to occupy less than 4k bytes of user task space and is much smaller and faster than the debugger bundled in the COBOL-81 kit. The PDP-11 Symbolic Debugger allows the user to refer to program locations by symbols or line numbers rather than by addresses, thus, saving valuable programmer time. Breakpoints and tracepoints may be set in overlay segments that are not currently resident. A programmer may step through a program by source line or by PDP-11 instructions which facilitates source debugging. Programs composed of modules written in COBOL-81 and MACRO-11 can be debugged by using the SET LANGUAGE command. The new "PDP-11 Symbolic Debugger Information for COBOL-81 Users" manual provides the debugger information specific to COBOL-81.

A specially priced upgrade license option is being offered to allow current PDP-11 COBOL V4.4 customers to upgrade their systems to COBOL-81. A Translator Utility and Translator Manual are provided in the documentation and media kits to aid in source code conversion. The pricing for the COBOL-81 Upgrade Option has been designed to protect the customer's original investment in PDP-11 COBOL.

For further information on COBOL-81 V2.4, PDP-11 Symbolic Debugger, or the upgrade package, please contact your Digital Equipment Corporation local sales office.

DECUS

DAR

DAARC

DAARC

Chairman
James Deck
Inland Steel Research Lab
East Chicago, IN

Symposium Coordinator
Mack Overton
FDA
Chicago, IL

Newsletter Editor
Ellen Reilly
William H. Rorer
Ft. Washington, PA

DEC Counterpart
Nancy Kilty
Digital Equipment Corporation
Marlboro, MA

Hardware & Interfacing
Peter Clout
Los Alamos National Lab
Los Alamos, NM

Math Statistics & Analysis
Herbert J. Gould
C.C.F.A. University of Illinois Med Center
Chicago, IL

Process Control - Industrial Automation
Bill Tippie
Kinetic Systems Corp.
Lockport, IL

RS-1
George Winkler
CPC International
Argo IL

EDITOR'S CORNER

The following article is an application note on VMS realtime programming techniques written by William Forbes of the Laboratory Data Products Group of DEC. I would like to thank him for submitting the article in addition to the two other authors listied this month, George Winkler and John Davies. In order to communicate information to DECUS members, I need you to relay your knowledge. If you have anything to submit I would appreciate it. Deadlines for submission are the 15th of every month.

If you would like to contact Bill Forbes with any questions, his address is :

William B. Forbes
Laboratory Data Products
Digital Equipment Corporation
One Iron Way MR02-3/M91 Box 1002
Marlboro, Mass. 01752-9102
(617) 467-5753

Don't forget to register for the upcoming symposium in San Francisco!

SEPTEMBER 1986
TABLE OF CONTENTS

| d | i | g | i | t | a | l |

VAX/VMS REALTIME PROGRAMMING APPLICATION NOTES

May 29, 1986

The realtime programmer who faces a VAX/VMS system for the first time may feel intimidated by the seeming complexity of the system in comparison to, say, PDP-11 systems running RT-11 SJ. In fact, the program development environment under VAX/VMS is very rich, comprising many tools, utilities and system services for developing and running realtime application programs.

The material contained in this application note is intended to help you get started using some basic VAX/VMS services for realtime programming. A more informal title might have been "How to Do Programmed I/O under VAX/VMS without Writing a Device Driver." Specifically, information is presented on two relevant techniques: accessing the VAX I/O page for direct device control, and using the connect-to-interrupt driver to do interrupt programming. An emphasis has been placed on explaining the principles underlying the use of these services rather than the details of their operation. A more detailed description may be found in Appendix C of the VAX/VMS Release notes for Version 4.0. For additional copies of these notes or for technical assistance, call the Laboratory Data Products Hotline at (617) 467-5441.

1 DIRECT PROGRAM CONTROL OF I/O MODULES UNDER MICROVMS

The most basic level at which realtime devices can be controlled is by direct manipulation of the contents of device registers. Direct control of device registers is relatively simple in PDP-11 systems and, in fact, is a common programming technique for certain kinds of realtime applications. For example, the following MACRO-11 code fragment illustrates the acquisition of a single data value from an A/D device.

```
        ADCSR   = 770400        ;Address of device CSR
        ADBUF   = ADCSR+2       ;Address of device buffer

        MTPS    #340            ;Set processor priority level to 7
        MOV     #1,ADCSR        ;Set the GO bit to start a conversion
LOOP:   BIT     #200,ADCSR      ;Test the A/D DONE bit
        BEQ     LOOP            ;If clear, test again
        MOV     ADBUF,R0        ;If set, move the data value
                                ;  into register 0
        MTPS    #0              ;Drop priority level back to 0
```

In this example, the processor priority level is first raised to 7. This insures that the processor will execute the subsequent instructions without interruption. An A/D conversion is then initiated by setting bit 0 of the device control/status register (CSR). When the conversion is complete, bit 7 of the CSR is set by the device. The program repeatedly tests this bit until it finds it to be set, then moves the data value from the device's buffer register into a general-purpose processor register (or into a storage location in main memory). This technique is called "polled I/O".

Since many polled I/O operations require that the processor be totally dedicated to the polling loop, this technique has not been widely used on VAX/VMS systems, which typically support multiple tasks or users at a time. However, the increasing use of MicroVAX systems for dedicated realtime processing has brought about a demand for polled I/O routines implemented under MicroVMS.

As with RT-11 based PDP-11 systems, improper manipulation of device registers in the MicroVAX I/O page can lead to undesirable consequences. Generally speaking, direct device control techniques should be used to manipulate only process-dedicated realtime option module registers. Inadvertently changing other device register contents may cause a system crash or the corruption of a mass storage volume. You should be sure to have a secure backup copy of the system whenever debugging direct device control routines.

To develop polled I/O (or other direct control routines) under MicroVMS, two special techniques are called for:

1. A means of addressing device registers in the I/O page, and
2. A means of raising and lowering the processor priority level.

This section illustrates the techniques used to accomplish these operations in the context of programs for performing single-channel clocked analog input using an AXV11-C analog module and a KWV11-C clock module.

1.1 Accessing Device Registers In The VAX I/O Page

On PDP-11 systems running the RT-11 Single-Job monitor, accessing addresses in the I/O page involves nothing more than supplying a 16-bit address in the range 160000 - 177776 (octal). Those addresses correspond to the portion of the physical address space within which device registers are defined - the I/O page. As with PDP-11 systems, the VAX architecture reserves a region of the physical address space for device registers. For MicroVAXes (I and II) the I/O page starts at address 20000000 (hex), and is 2000 (hex) bytes in length.

Under MicroVMS, the 32-bit virtual address space of the system is mapped into physical memory, so that a user-supplied address does not ordinarily correspond to any particular physical address. The problem, then, is to establish a means of addressing a particular range of physical addresses - that is, the MicroVAX I/O page.

This is accomplished by mapping the physical addresses which constitute the I/O page into a portion of the process's virtual address space. The VMS $CRMPSC system service is used. The following parameters are passed to the $CRMPSC service:

- o The starting and ending virtual addresses into which the section is to be mapped; this virtual address block should be 8K bytes long and page-aligned
- o The starting page frame number of the section to be mapped (a page frame is a 512-byte block of physical memory); computed as <I/O page starting physical address>/512
- o The number of pages in the section (16 in this case)

In addition, the $CRMPSC system service accepts a flag mask specifying the type of section to be created, as well as its characteristics. In the present context, the essential flags are SEC$M_PFNMAP and SEC$M_WRT, which define the section as a page frame section with the read/write attribute.

Finally; the call to the $CRMPSC system service must specify storage locations into which the starting and ending virtual pages actually mapped and the channel number assigned by the system to the section may be written. Ordinarily, these data are not used by the programmer and need not be elaborated on here. For a fuller discussion of the $CRMPSC system service, see the VAX/VMS System Services Reference Manual.

Once the mapped section has been created, instructions which address locations within the virtual address space into which the section has been mapped will effectively address the corresponding locations in the I/O page. To address a particular device register in the I/O page, the absolute offset in the I/O page of the register is added to the base address of the virtual address block, yielding a virtual address which corresponds to the desired physical address.

1.2 Raising And Lowering Processor Priority Level

VAX architecture defines 32 interrupt priority levels (IPL 0 - IPL 31). At any point in time, the processor will be operating at some IPL, usually 0 during execution of user code. Device interrupts occur at IPL 20 - 23, corresponding to bus request levels 4 - 7. On MicroVAXes, driver code executes at IPL 23, regardless of the level at which the device interrupted. For example, a device interrupt on BR 4 would not be granted until the processor priority dropped below 20. However, when the request was granted, the processor priority would be set to IPL 23, thus blocking ALL device interrupts until the driver lowered the IPL. The system timer interrupts at IPL 22 and is granted at the same level.

When the processor is executing code at a low IPL, device, timer, or software interrupts at a higher IPL can pre-empt the system. When this happens, the processor saves the context of the currently executing image (program counter, processor status longword) and transfers control to the interrupt service routine (ISR). When the ISR completes the context of the pre-empted image is restored and it continues executing where it left off (assuming there are no new interrupts pending). Thus, an image executing at a low IPL may be suspended for various periods of time due to the occurrence of interrupts.

When performing polled I/O it may be desirable for the processor to respond as quickly as possible to the availability of data from the device being polled. To prevent interrupts from distracting the processor from the polling operation, the polling code must be executed at an IPL above that of any device which might generate a pre-emptive interrupt. To accomplish this, the code raises the IPL to 30 using the DSBINT system macro. The value of 30 is chosen to enable a power-fail interrupt (IPL 31) to be processed, should one occur. This is advisable since allowing the power-fail ISR to execute may prevent corruption of the system device. Besides, blocking the power-fail ISR will probably not salvage the application.

VAX processor design imposes the restriction that IPL cannot be raised except while the processor is operating in kernel mode. The $CHMKNL system service must be invoked to change mode to kernel before executing the DSBINT macro. The polling code then executes in kernel mode. Whenever the processor is executing above IPL 2, page faults are fatal (the system will crash). Thus, before entering kernel mode, the $LCKPAG system service must be called to lock any data areas that are addressed in the polling routine into physical memory. When the polling routine completes, the program should restore the original IPL (usually 0), return to user mode, and, optionally, unlock the pages addressed in the polling routine.

In summary, the sequence of program steps in performing polled I/O at elevated IPL is as follows:

1. The I/O page is mapped into process virtual address space using the $CRMPSC system service.

2. Any required device initialization is performed at IPL 0, user mode.

3. Any pages addressed in the polling routine are locked into physical memory using the $LCKPAG system service.

4. The processor mode is changed to kernel using the $CHMKNL system service.

5. In kernel mode, the IPL is raised to 30.

6. The polling code executes and data is moved into a process buffer which has been locked into memory (step 3, above).

7. When data transfer is complete the IPL is returned to its prior value (usually 0).

8. The processor mode is changed back to user by exiting the kernel mode routine.

9. The device is reset, if necessary.

10. Optionally, the pages addressed in kernel mode are unlocked using the $ULKPAG system service.

11. Post-processing of the acquired data is performed in user mode at IPL 0.

The following FORTRAN and MACRO-32 modules illustrate this sequence for single-channel clocked analog input using AXV11-C and KWV11-C modules.

To execute:

```
        $ FORTRAN POLLED_AD
        $ MACRO FASTAD32
        $ LINK POLLED_AD,FASTAD32
        $ SET PROCESS/PRIV=(CMKRNL,PSWAPM,PFNMAP) !Required privileges
        $ RUN POLLED_AD

=========================================================================

        PROGRAM POLLED_AD

        INCLUDE '($SYSSRVNAM)'

        INTEGER*2 IOFF, IVAL, IBUF(60000)
        INTEGER*4 ISTATUS, MAPIOP, NPNTS, LOCKS(2)

c  Use the $LCKPAG system service to lock the input buffer into physical
c  memory.

        LOCKS(1)=%LOC(IBUF(1))
        LOCKS(2)=%LOC(IBUF(60000))
        ISTATUS=SYS$LCKPAG(LOCKS,,)
        IF(.NOT.ISTATUS) CALL EXIT(ISTATUS)

c  Call routine MAPIOP to map the I/O page in virtual address space.

        ISTATUS=MAPIOP()
        IF(.NOT.ISTATUS) CALL EXIT(ISTATUS)

c  Input/initialize data acquisition parameters

        TYPE 9060
9060    FORMAT('$Base clock rate, clock preset, number of samples? ')
        ACCEPT *, IRATE, KOUNT, NPNTS
        ICHAN = 0
        MODE = 0

c  Call the sampling routine. Control will return to the main program
c  only after I/O is complete.

        CALL FASTAD32(ICHAN,KOUNT,IRATE;IBUF,NPNTS,MODE,ISTATUS)

c  Output data, return status (residual AXV CSR)

        TYPE 9130, (J, IBUF(J),J=1,NPNTS)
9130    FORMAT(1X,2I10)
        TYPE 9140,ISTATUS
9140    FORMAT(/' ISTATUS =',O10)

        END
```

```
=========================================================================

        .TITLE FASTAD32

        .LIBRARY /SYS$LIBRARY:LIB.MLB/

        .PSECT  MAPIOP_MAIN     RD,WRT,PAGE

; NOTE: all subsequent MACRO-32 code in this example is contained in this
;       .PSECT.

VIOP:           .BLKB   8192            ; The virtual pages to which the
VIOP_END:                               ; I/O page will be mapped

PIOPAGE:        .LONG   VIOP            ; Starting and ending virtual page
                .LONG   VIOP_END        ;   addresses
RETPAGE:        .BLKL   2               ; Starting and ending page addrs
                                        ;   used (should be the same)
SECNAME:        .ASCID  /IOPAG_GLSEC/   ; Name of the section to be created
IOPAGECHAN:     .LONG                   ; Channel # to be associated with
                                        ;   the created section
PFNUM:          .LONG   ^X100000        ; Page frame number of the I/O page
                                        ;   (20000000 hex) / (200 hex)

;++
; Routine to map the I/O page into process virtual address space
;--

        .ENTRY  MAPIOP,^M<>

        $CRMPSC_S-
                INADR=PIOPAGE,-
                RETADR=RETPAGE,-
                FLAGS=#SEC$M_PFNMAP+SEC$M_WRT,-
                GSDNAM=SECNAME,-
                CHAN=IOPAGECHAN,-
                PAGCNT=#16.,-
                VBN=PFNUM

        RET
```

```
=========================================================================

;++
;       FASTAD32 - Performs single channel, polled I/O using AXV11-C and
;                  KWV11-C modules.
;
;       The FORTRAN calling interface is:
;
;       CALL FASTAD32(ichan,kount,irate,ibuf,npnts,mode,istatus)
;
;       where:
;               ichan   =       AXV11-C A/D channel number
;               kount   =       KWV11-C preset value
;               irate   =       clock rate:
;                                       1 = 1 MHz
;                                       2 = 100 KHz
;                                       3 = 10 KHz
;                                       4 = 1 KHz
;                                       5 = 100 Hz
;               ibuf    =       array to store data
;               npnts   =       number of elements in ibuf
;               mode    =       if 0 then start immediately;
;                               if<>0 then start on ST2
;               istatus =       return status; if<0 then error
;
; This routine should be in the same .PSECT as the MAPIOP routine.
;
; The CLK OVFL pin on the KWV is strapped to the RTC IN pin on the AXV.
;
;--

        ADCSR   = VIOP+^O10400  ;address of AXV11-C A/D CSR
        ADBUF   = ADCSR+2       ;address of AXV11-C A/D BUFFER
        KWCSR   = VIOP+^O10420  ;address of KWV11-C CSR
        KWPRE   = KWCSR+2       ;address of KWV11-C BUFFER/PRESET

        ; Argument pointer offsets

        ICHAN   =  4
        KOUNT   =  8
        IRATE   = 12
        IBUF    = 16
        NPNTS   = 20
        MODE    = 24
        ISTATUS = 28


        .ENTRY  FASTAD32,^M<R6,R7,R8,R9,R10>

; Note that all instructions which address the I/O page are WORD MODE.

        TSTW    @#ADBUF         ;clear A/D DONE flag
        MOVZWL  @ICHAN(AP),R6   ;channel # to sample
        ASHL    #8,R6,R6        ;move channel # to byte 2 of R6
        BISB    #^O40,R6        ;enable clock driven
```

```
        MOVW    R6,@#ADCSR      ;load A/D CSR

        MOVZWL  @IRATE(AP),R6   ;clock rate in R6
        BICL    #^O177770,R6    ;clear excess bits
        ASHL    #3,R6,R6        ;shift rate to bits 3 - 5
        MOVW    R6,@#KWCSR      ;load KW CSR

        MNEGW   @KOUNT(AP), -
                @#KWPRE         ;load KW preset register

        MOVL    IBUF(AP),R6     ;load address of IBUF into R6
        MOVZWL  @NPNTS(AP),R7   ;load NPNTS into R7
        MOVL    #ADCSR,R8       ;R8 points to A/D CSR
        MOVL    #ADBUF,R9       ;use R9 as pointer to A/D buffer reg
        MOVZWL  @MODE(AP),R10   ;pass MODE arg to POLL in R10

        $LCKPAG_S -             ;lock polling code into memory
                INADR=LCKPAG, -
                RETADR=LCKRET

        $CMKRNL_S POLL          ;execute routine POLL in kernel mode

        CLRW    @#KWCSR         ;zero KWCSR - turns clock off
        MOVZWL  (R8), -         ;return status is residual AXV CSR
                @ISTATUS(AP)
        CLRW    (R8)            ;clear A/D CSR

        RET                     ;return to fortran


LCKPAG: .LONG   POLL
        .LONG   POLL_END
LCKRET: .BLKL   2

        .ENTRY  POLL,^M<R6,R7,R8,R9,R10>

        DSBINT  #30             ;disable all interrupts (except powerfail)
        TSTW    R10             ;test mode
        BEQL    1$              ;if 0 then trigger immediately
        BISW    #20002,@#KWCSR  ;set clock to wait for ST2
        BRB     2$              ;and skip over next instruction
1$:     BISW    #3,@#KWCSR      ;trigger immediately

2$:     BBC     #7,(R8),2$      ;is conversion done?
        MOVW    (R9),(R6)+      ;store A/D value
        SOBGTR  R7,2$           ;decrement NPNTS; if not zero, loop again

        ENBINT                  ;restore IPL to prior value

POLL_END:
        RET

        .END
```

```
===========================================================================

;++
;  These routines are not used in the present example. They are provided
;  only for general reference.
;
;  FORTRAN calling interface:
;
;     CALL IPEEK32 ( OFFSET, VALUE )
;
;     CALL IPOKE32 ( OFFSET, VALUE )
;
;        where
;
;           OFFSET      = integer*2 offset in the I/O page
;
;           VALUE       = integer*2 value from/for the register addressed
;
; These routines should be in the same .PSECT as the MAPIOP routine.
;
;--

        .ENTRY  IPEEK32,^M<R6>

        MOVZWL  @4(AP),R6               ;put OFFSET in R6
        MOVZWL  L^VIOP(R6),@8(AP)       ;put value from iopage in VALUE

        RET


        .ENTRY  IPOKE32,^M<R6>

        MOVZWL  @4(AP),R6               ;put OFFSET in R6
        MOVW    @8(AP),L^VIOP(R6)       ;put VALUE into iopage

        RET
```

2 INTERRUPT PROGRAMMING UNDER VMS USING CONNECT-TO-INTERRUPT

Interrupt service routines on unmapped, single-tasking systems such as PDP11s running RT11-SJ are fairly straightforward. An interrupt occurs and control is transfered directly to the interrupt service routine (ISR). This involves saving the current program counter (PC) and processor status word (PSW) and replacing them with the PC and PSW stored at the vector address for the interrupting device. When I/O is complete, a completion flag may be set to notify the mainline program that the data have been stored (or output). The interrupt service routine returns control to the mainline program by executing an RTI instruction which restores the saved PC and PSW registers. The mainline program may test the completion flag to detect I/O completion.

The virtual, multi-tasking, multi-user nature of VAX/VMS makes interrupt handling more complicated. Since the process which requested the I/O might not be current at the time an interrupt occurred, the ISR code and data must reside in system virtual address space and must execute in system context. If this were not the case, a context switch to that of the requesting process might be needed before the interrupt could be serviced and this would impose too great a penalty in response time. Similarly, notification of I/O completion must be handled asynchronously. Finally, the sharing of I/O resources among multiple, possibly non-cooperating processes imposes additional architectural demands.

VMS resolves these issues through the implementation of an elegant, though complex, I/O subsystem. All VMS device drivers must interface to the I/O subsystem to insure the orderly flow of information between the various users' processes and shareable I/O devices. However, most realtime application programmers would prefer to avoid writing a full VMS device driver for controlling realtime I/O modules. This is particularly true in light of the fact that realtime devices generally need not be shareable among non-cooperating processes, whereas device shareability is one of the sources of complexity in the I/O subsystem. The connect-to-interrupt driver enables users to program interrupt service routines and asynchronous I/O completion routines without having to write a full VMS device driver.

2.1 The Connect-to-Interrupt User Interface

The connect-to-interrupt driver (CONINTERR or CIN) is a template VMS device driver into which blocks of user-supplied code and data may be linked. The user-supplied code and data are contained in a single .PSECT hereafter referred to as the "CIN buffer". The CIN buffer is compiled and linked as part of the application program. The linkages between the CIN driver and the user-supplied CIN buffer are formed at run-time by the $QIO system service. As a part of the process of building these links, the CIN buffer is mapped into system virtual address space (S0) while preserving the mapping in process virtual address space (P0). The result is that the CIN buffer is doubly-mapped in S0 and P0 virtual address space. Thus, code and data in the CIN buffer are accessible in both system and process context.

The CIN buffer comprises 5 sections:

1. A data area containing all data structures to be addressed during the execution of user-supplied CIN code.

2. A device initialization routine which is executed during recovery from a power failure.

3. A start I/O routine which is executed at the time the $QIO is issued.

4. An interrupt service routine which is executed in response to a device interrupt.

5. A cancel I/O routine which is executed when the user process issues a cancel I/O request.

In addition, the user may specify an AST routine to be executed in process context on I/O completion (or partial completion). It is important to note that any user-supplied code in the CIN buffer may only address data and code contained within the CIN buffer. Code which executes in process context, including the user-specified AST routine, may also address data and code in the CIN buffer and, of course, in any other portion of process virtual address space. The sections of the user-supplied CIN buffer are illustrated diagramatically below.

```
        .PSECT  CIN_USER        PIC,USR,CON,REL,LCL,NOSHR,EXE,RD,WRT

        |________________________________|
        |                                |
        |          Data buffers          |
        |                                |
        |________________________________|
        |                                |
        |          INIT routine          |
        |     Executed after powerfail   |
        |                                |
        |________________________________|
        |                                |
        |          START routine         |
        |         Executed by $QIO       |
        |                                |
        |________________________________|
        |                                |
        |           INT routine          |
        |  Executed on device interrupt  |
        |                                |
        |________________________________|
        |                                |
        |         CANCEL routine         |
        |       Executed by $CANCEL      |
        |                                |
        |________________________________|
```

This application note is intended to provide an overview of CONINTERR concepts for intermediate to advanced programmers who want to get started using this facility. Many of the details of CONINTERR functionality and internals have been omitted, though what is presented is sufficient for many applications. For a more detailed description of CONINTERR, see the VAX/VMS Release Notes, Appendix C.

2.2 Example Code Internals

The example program performs continuous interrupt-driven analog input to a process buffer using an AXV11-C analog module and a KWV11-C clock module for timebase generation. Data are sampled into a 4096-word ring buffer with 2 sub-buffers. The ring buffer is contained in the CIN buffer and is therefore doubly mapped in S0 and P0. When a sub-buffer is filled, an AST is delivered to the calling process. The AST routine moves the data from the sub-buffer to a process buffer (singly mapped in P0 virtual address space). It then checks to determine whether I/O is complete; that is, whether the process buffer has received all the data requested. If it has, the AST issues a $CANCEL system service call to terminate I/O and sets a completion flag to notify the calling program of I/O completion.

SYSTEM SETUP FOR CONNECT-TO-INTERRUPT

1. Log in to SYSTEM account.

2. Insert the following line in the file SYS$SYSTEM:MODPARAMS.DAT

```
REALTIME_SPTS = 20
```

3. Enter:

```
$ @SYS$UPDATE:AUTOGEN SAVPARAMS REBOOT
      .
      .
(system reboots)
```

4. Log in to SYSTEM account or other privileged account and enter:

```
$ MCR SYSGEN
SYSGEN> CONNECT AXA0:/ADA=0/CSR=%O770400/VEC=%O400/DRIVER=CONINTERR
SYSGEN> EXIT

$ MACRO AXV
$ FORTRAN CALL_AXV
$ LINK CALL_AXV,AXV
$ SET PROCESS/PRIV=(PSWAPM,CMKRNL)   !Required privileges
$ SET PROCESS/PRIORITY=17
$ RUN CALL_AXV
```

===

System page table entries for double-mapping of CONINTERR buffers are drawn from a pre-allocated pool. In steps 1 - 3, above, the size of this pool is set by modifying the SYSGEN parameter REALTIME_SPTS. The number of page table entries allocated must be sufficient to map the user buffers (data and code) of all CONINTERR-driven devices which are connected at any given time. In the present example, 20 page table entries are allocated, sufficient to map 5120 words of data and code. This step needs to be taken only when additional REALTIME_SPTS are required for mapping CONINTERR buffers.

In step 4, above, the device is connected to the system. In this case, the device was given the name "AXA0"; this is the name used in the $ASSIGN system service call. Switches are used to designate the adapter number (always 0 on MicroVAXes), the CSR and vector addresses of the module, and the driver name (CONINTERR). This step needs to be taken each time the system is booted. The commands may be inserted into the file SYS$MANAGER:SYCONFIG.COM, if desired.

```
        PROGRAM CALL_AXV        !Calling program for AXV_SAMPLE

        INCLUDE '($SYSSRVNAM)'

        BYTE ICMPF
        INTEGER*2 PBUFF(30000)
        INTEGER*4 ISTATUS, ICHAN, IPRESET, ICOUNT
        INTEGER*4 LOCKS(2)
        INTEGER*4 AXV_SAMPLE

        COMMON /PROCESS_DATA/ PBUFF, ICMPF

c  Lock the process buffer into the working set. This is not essential,
c  but it helps to avoid page faulting in the AST routine.

        LOCKS(1) = %LOC(PBUFF(1))
        LOCKS(2) = %LOC(PBUFF(30000))
        ISTATUS = SYS$LCKPAG (LOCKS,,)
        IF(.NOT.ISTATUS) CALL EXIT (ISTATUS)

c  Assign a channel number for the AXV

        ISTATUS = SYS$ASSIGN ('_AXA0:' ,ICHAN,,)
        IF(.NOT.ISTATUS) CALL EXIT (ISTATUS)

c  Input/initialize arguments to be passed to the calling routine

        TYPE *     / preset, sample count?'
        ACCEPT  , 'PRESET, ICOUNT

c  Routine AXV_SAMPLE handles all of the details of setting up and
c  executing the I/O operations. When the process buffer is filled,
c  the completion flag is set.

        ICMPF = 0       ! Clear the completion flag
        ISTATUS = AXV_SAMPLE (ICHAN, IPRESET, ICOUNT)
        IF(.NOT.ISTATUS) CALL EXIT (ISTATUS)

c  By looping on the completion flag, the process is insured of remaining
c  computable throughout the I/O operation.

100     IF(ICMPF.EQ.0) GO TO 100

c  For the sake of simplicity in this example, only a few of the acquired
c  data values are output. In a real application, the following step would
c  be replaced with file/graphic output.

        TYPE 9000, (J,PBUFF(J),J=1,1000,100)
9000    FORMAT(2I10)

        END
        .TITLE  AXV

        .LIBRARY /SYS$LIBRARY:LIB.MLB/

        $IDBDEF                 ; Definition for I/O drivers
        $UCBDEF                 ; Data structurs
        $IODEF                  ; I/O function codes
        $CINDEF                 ; Connect-to-interrupt
        $CRBDEF                 ; CRB stuff
        $VECDEF                 ; more


        .PSECT PROCESS_ROUTINES     PIC,USR,CON,REL,LCL,NOSHR,EXE,RD,WRT

;++
; This example routine issues the QIO to connect to the AXV interrupts.
; It takes care of the internals associated with the connect-to-interrupt
; QIO.
;
; FORTRAN calling sequence:
;
;   STATUS = AXV_SAMPLE ( CHAN, PRESET, COUNT )
;
;   where
;
;      CHAN     = longword channel # for device _AXA0
;
;      PRESET   = longword preset value for KWV
;                 (positive value <= 32K)
;
;      COUNT    = longword # of samples
;
;      STATUS   = longword return status of $QIO call
;
; In this example, the AXV and KWV CSRs are "firm-wired" for the following
; characteristics:
;
;       AXV - gain=1, RTC ENABLE, DONE INT ENABLE, channel=0
;
;       KWV - GO, mode=1, rate=1 (1 MHz)
;
; Ordinarily, they would be configured according to arguments passed
; to this routine from the calling program.
;
; The CLK OVFL pin on the KWV is strapped to the RTC IN pin on the AXV.
;
;--

        .ENTRY  AXV_SAMPLE,^M<>

; These values are stored in process address space for use in the AST
; routine

        MOVL    @4(AP),AXV_CHAN         ; Channel # for $CANCEL
        MOVL    @12(AP),POINT_COUNT     ; Point count
```

```
; These values are stored in system address space for use in the CIN user
; start routine.

        MOVW    #^O13,KWV_CSR_VALUE      ; GO, MODE=1, 1MHz
        MOVW    #^O140,AXV_CSR_VALUE     ; RTC ENABLE, DONE INT ENABLE
        MNEGW   @8(AP),KWV_PRESET_VALUE ; Negated KWV preset value

        $QIO_S  CHAN=@4(AP),-            ;Channel
                FUNC=#IO$_CONINTWRITE,- ;Allow writing to the data buffer
                IOSB=AXV_CIN_IOSB,-      ;I/O status Block
                P1=AXV_CIN_BUF_DESC,-    ;Buffer descriptor
                P2=#AXV_CIN_ENTRY,-      ;Entry list
                P3=#AXV_CIN_MASK,-       ;Status bits,etc
                P4=#USER_AST,-           ;AST service routine
                P6=#10                   ;preallocate some AST control
                                         ; blocks
        RET                              ;Return to calling routine

AXV_CIN_BUF_DESC:                        ;Buffer descriptor for CIN
        .LONG   USER_END - RINGBUF
        .LONG   RINGBUF

AXV_CIN_ENTRY:
        .LONG   USER_INIT - RINGBUF      ;Init code
        .LONG   USER_START - RINGBUF     ;Start code
        .LONG   USER_INT - RINGBUF       ;Interrupt service routine
        .LONG   USER_CNCL - RINGBUF      ;I/O cancel routine

AXV_CIN_IOSB:
        .LONG   0                        ; I/O Status Block
        .LONG   0

; Control mask - see VMS Release Notes, Appendix C for explication

AXV_CIN_MASK = CIN$M_REPEAT!CIN$M_START!CIN$M_ISR!CIN$M_CANCEL


        .SBTTL  USER_AST, User AST routine

;++
; This routine is invoked when I/O completion is signaled by the USER_INT
; routine. It is queued to the user's process in the access mode of the
; $QIO which initiated the I/O. It executes in process context at
; IPL$_ASTDEL (IPL 2).
;
; I/O completion is signalled by loading SS$_NORMAL (1) into R0 on exiting
; the USER_INT routine. In the present example, this does not signal full
; completion of the I/O, since the CIN$M_REPEAT flag was set in the $QIO
; call.
;
;--

        .ENTRY  USER_AST,^M<R2,R3,R4,R5>
        MOVAL   RINGBUF,R2              ; Get CIN buffer address
        MOVZWL  RINGBUF_INDEX,R3        ; Get buffer index
        BBS     #11,R3,10$              ; Is bit 11 set?
        ADDL    #4096,R2                ; No, xfer the top half
10$:    MOVAL   PBUFF,R4                ; Get process buffer address
        MOVC3   #4096,(R2),(R4)         ; Move data (trashes R0-R5)
        ADDL    #4096,PBUFF_INDEX       ; Update process buffer "index"
        SUBL2   #2048,POINT_COUNT       ; Update point count
        BGTR    20$                     ; Have we moved all the data?
        JMP     ALL_DONE                ; Yes - Finish up
20$:    MOVZWL  #SS$_NORMAL,R0          ; Set return status
        RET                             ; And return

ALL_DONE:
        $CANCEL_S -                     ; Cancel the I/O request
                CHAN=AXV_CHAN
        CLRL    PBUFF_INDEX             ; Reset index into PBUFF
        MOVB    #1,ICMPF                ; Set the completion flag
        MOVZWL  #SS$_NORMAL,R0          ; Set return status
        RET                             ; And return

PBUFF_INDEX:
        .LONG   0

POINT_COUNT:
        .LONG   0

AXV_CHAN:
        .LONG   0


        .PSECT PROCESS_DATA     PIC,OVR,REL,GBL,SHR,NOEXE,RD,WRT,LONG

;++
; This is the FORTRAN common block /PROCESS_DATA/
;--

PBUFF:  .BLKW   30000

ICMPF:  .BYTE   0
```

```
        .PSECT CIN_USER         PIC,USR,CON,REL,LCL,NOSHR,EXE,RD,WRT

;++
; This is the CIN buffer. In the present example, it is contained in
; the same source code module as the process routines shown above.
;--

        .SBTTL  DATA STRUCTURES

DATA_BUF_SIZE  =        4096

RINGBUF:                .BLKW   DATA_BUF_SIZE
RINGBUF_END:
RINGBUF_INDEX:          .WORD

AXV_CSR_VALUE:          .WORD
KWV_CSR_VALUE:          .WORD
KWV_PRESET_VALUE:       .WORD

; In the CIN user routines, the system virtual address of the CSR of the
;   device is supplied. Other addresses in the I/O page used by the code
;   must be handled as offsets from the device CSR.

AXV_ADDRESS     = ^O170400              ;Address of AXV
KWV_ADDRESS     = ^O170420              ;Address of KWV
KWV_OFFSET      = KWV_ADDRESS-AXV_ADDRESS ;Offset for KWV CSR
PRESET_OFFSET   = KWV_OFFSET+2          ;Offset for KWV BPR
DBR_OFFSET      =2                      ;Offset for AXV DBR


        .SBTTL  USER_INIT       ; Dummy dev intialization routine
;++
; This routine is invoked after power recovery. It executes in system
; context at IPL$_POWER (IPL 31).
;
; This routine is not implemented in the present example. However, the
; entry point must be defined and included in the entry list of the
; $QIO (P2).
;
; See VAX/VMS Release Notes, Appendix C for inputs.
;
;--

USER_INIT::
        RSB


        .SBTTL  USER_START, Start I/O routine

;++
; This routine is invoked by the $QIO system service. It executes in
; system context at IPL$_QUEUEAST (IPL 6).
;
;   On entry:
```

```
;
;       0(R2)  - arg count of 4
;       4(R2)  - Address of the process buffer (system mapped)
;       12(R2) - Address of the device's CSR
;
; See VAX/VMS Release Notes, Appendix C for other inputs.
;
; The routine must preserve all registers except R0-R4.
;
;--

CSR_ADD = 12                    ; Argument list offset of CSR addr
                                ; (only valid in USER_START and USER_INT)

USER_START::
        CLRW    RINGBUF_INDEX           ; Clear ring buffer index
        MOVL    CSR_ADD(R2),R0          ; Get address of the AXV CSR
        TSTW    DBR_OFFSET(R0)          ; Clear AD DONE bit, if set,
                                        ;   by reading AXV DBR
        MOVW    AXV_CSR_VALUE,(R0)      ; Set up the AXV
        MOVW    KWV_PRESET_VALUE, -     ; Set KWV preset
                PRESET_OFFSET(R0)
        MOVW    KWV_CSR_VALUE, -        ; Set KWV CSR
                KWV_OFFSET(R0)
        MOVZWL  #SS$_NORMAL,R0          ; Load a success code into R0.
        RSB                             ; Return


        .SBTTL  USER_INTERRUPT, Interrupt service routine

;++
; This routine is invoked on device interrupt. It executes in system
; context at IPL$_DIPL (IPL 23 in MicroVMS).
;
; On entry:
;
;       0(R2)  - arg count of 5
;       4(R2)  - Address of the process buffer
;       8(R2)  - Address of the AST parameter
;       12(R2) - Address of the device's CSR
;
; See VAX/VMS Release Notes, Appendix C for other inputs.
;
; The routine must preserve all registers except R0-R4
;
;--

USER_INT::
        MOVL    CSR_ADD(R2),R0          ; Get AXV CSR address
        MOVAL   RINGBUF,R1              ; Get CIN buffer address
        MOVZWL  RINGBUF_INDEX,R3        ; Get buffer index
        MOVW    DBR_OFFSET(R0),(R1)[R3] ; Read data (assume no error)
        INCW    R3                      ; Increment buffer index
        BICW    #^O170000,R3            ; Clear all but bottom 12 bits of
                                        ;   (ring) buffer index
```

```
        MOVW    R3,RINGBUF_INDEX        ; Put updated index in storage
        CLRL    R0                      ; Clear return status
        BICW    #^O174000,R3            ; Now test bottom 11 bits of
                                        ;   buffer index
        BNEQ    10$                     ; Skip AST if not zero

; The user AST will be queued if the LSB of R0 is set on return

5$:     MOVZWL  #SS$_NORMAL,R0          ; Queue the AST
10$:    RSB


        .SBTTL  USER_CANCEL, Cancel I/O routine

;++
; This routine is invoked by the $CANCEL system service. It executes in
; system context at IPL$_QUEUEAST (IPL 6).
;
; On entry:
;
;   JSB interface:
;
;       R3      -  Addr of current IRP
;       R4      -  Addr of PCB of cancelling process
;       R5      -  Addr of the UCB
;
;   CALL interface:
;
;       0(AP)   Arg count 4
;       8(AP)   Addr of IRP
;       12(AP)  Addr of PCB
;       16(AP)  Addr of UCB
;
; See VAX/VMS Release Notes, Appendix C for other inputs.
;
; The routine must preserve all registers except R0 and R3.
;
;--

USER_CNCL::
        MOVL    UCB$L_CRB(R5),R0         ; Get Address of the CRB
        MOVL    CRB$L_INTD+VEC$L_IDB(R0),R0 ; Address of the IDB
        MOVL    IDB$L_CSR(R0),R0         ; Get addr of AXV
        CLRW    KWV_OFFSET(R0)           ; Stop clock
        TSTW    DBR_OFFSET(R0)           ; Clear AD DONE bit
        CLRW    (R0)                     ; Clear AXV CSR
        MOVZWL  #SS$_NORMAL,R0           ; Load return status
        RSB                              ; And return

; Label that marks the end of the module

USER_END:                               ; Last location in module
        .LONG

        .END
```

ATTENTION RS/1 USERS

DECUS SYMPOSIUM IN NASHVILLE

George A. Winkler
RS/1 Working Group

The time for session submissions is past for San Francisco, however, it is not too early for session submissions for the Nashville DECUS symposium. If you will be going and can give a paper on how you are using RS/1, please submit an abstract to me or to Mack Overton. The addresses are

Mack Overton
Food & Drug Administration
10 W. 35th Street
Chicago, IL 60607

George Winkler
CPC International
P.O. Box 345
Argo, IL 60501

The more we hear about your actual applications, the more we can make use of our resources.

I'm also interested in receiving any favorite RS/1 procedures you have so they can be compiled on a tape for distribution. I would like to hear any suggestions you might have with regard to distributing these procedures.

Hope to see you in San Francisco and Nashville.

MICRO POWER PASCAL/RT

John T. Davies III

There has been a problem with MicroPower Pascal/RT crashing after exiting large programs, like MPP (see section 1.12.1 of System Release Notes of Version 2.1). I have found that this problem seems to occur only when spool is running as a foreground or system job. This may occur with any foreground or system job, I'm not sure yet.

To work around this problem however, edit the MPBLD generated command file to inlcude the following lines at the beginning:

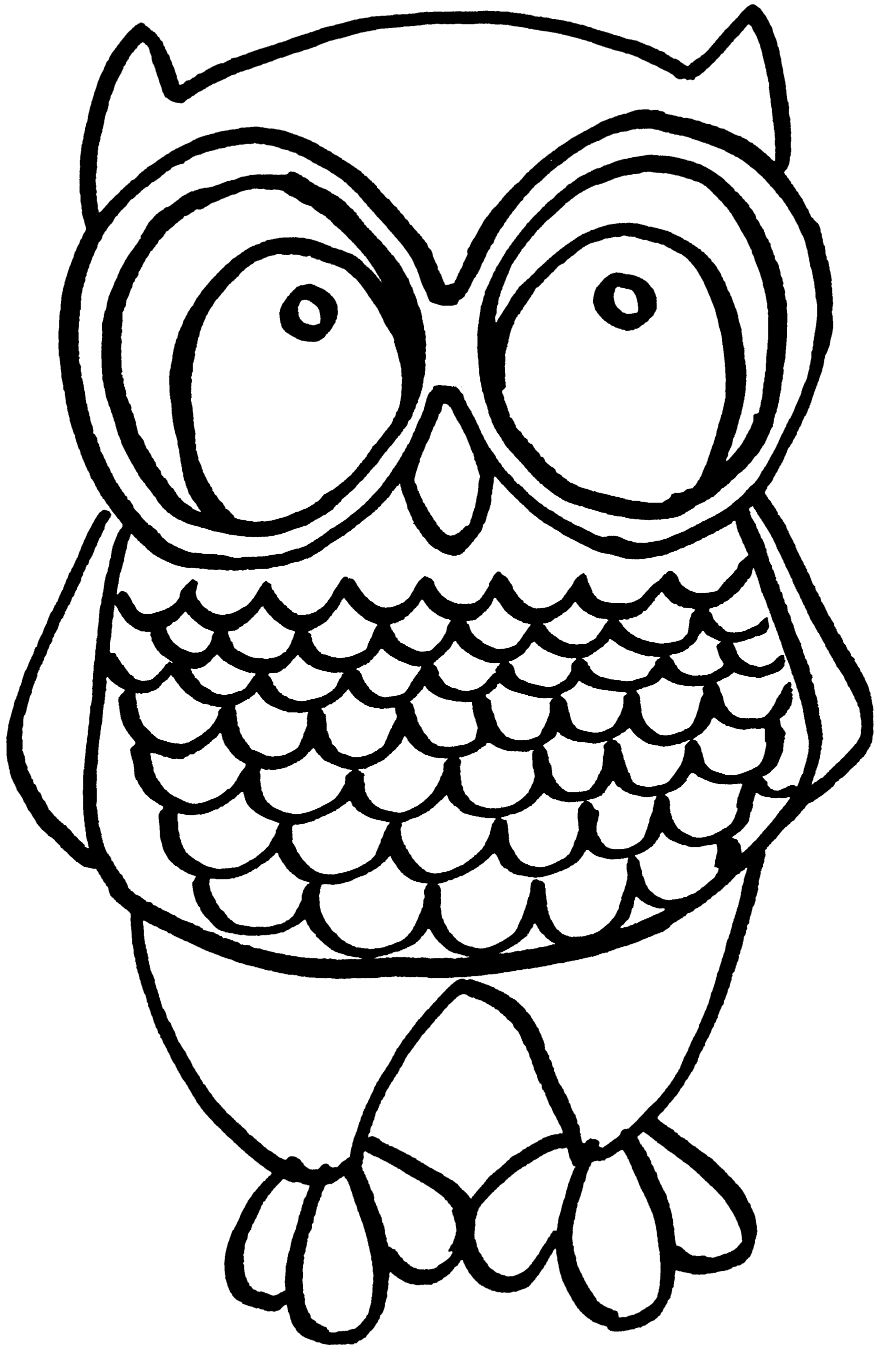
```
SET ERROR NONE
ABORT SPOOL
UNLOAD SPOOL
SET ERROR WARNING
```

The SET ERROR commands are used to keep the command file from aborting if SPOOL isn't running. At the end of the command file, either FRUN SY:SPOOL or SRUN SPOOL to put it back in place.

Any questions or comments please contact me at :

John T. Davies III
Thermo Electron Instruments
524 Alpha Drive
Pittsburgh, PA 15238-2912
(412) 963-0903

DMS SIG



JOIN THE WISE

Data Management Systems SIG Steering Committee
September 1, 1986

SIG Chairman:
Joseph F. Sciuto
Army Research Institute
Alexandria, VA
(202) 274-9420

Comptroller:
Alan Schultz
Land Bank National DP Center
Omaha, NE
(402) 397-5040

Symposium Coordinator:
Keith Hare
JCC

Granville, OH
(614) 587-0157

Symposium Coordinator:
Barbara Mann
TRW
Redondo Beach, CA
(213) 532-2211

Communications Committee Representative/
Newsletter Editor:
J. G. Russell Poisson
SEED Software Corporation
Alexandria, VA
(800) 428-9400

Session Note Editor:
Mark Morgan
Farm Credit Banks
Springfield, MA
(413) 732-9721

Membership Coordinator:
VACANT

MIS Working Group Coordinator (Past SIG Chairman):
Steve Pacheco
Ship Analytics
North Stonington, CT
(203) 535-3092

MIS Working Group (Past SIG Chairman):
Sandy Krueger
Key Financial Systems, Inc.
Pine Brook, NJ
(201) 299-6600

Working Group Coordinator/
Database Working Group:
Jim Perkins
PSC, Inc.
Shelburne, VT
(802) 863-8825

Forms Working Group:
Debbie Kennedy
Land Bank National DP Center
Omaha, NE
(402) 397-5040

Non-Digital Working Group:
Doug Dickey
GTE Government Systems
Rockville, MD
(301) 294-8400

RMS Working Group Coordinator:
Allen Jay Bennett
Lear Siegler Apistan
Grand Rapids, MI
(616) 451-6429

Pre-Symposium Seminar Coordinator/
Black Book:
David B. Turner
Korn/Kerry International
Los Angeles, CA
(408) 945-9600

ANSI Standards Coordinator:
Herman "Spence" Spencer
Army Research Institute
Alexandria, VA
(202) 294-9420

Member-At-Large:
Larry W. Hicks
Relational Database Services
Kernersville, NC
(919) 996-4882

Member-At-Large:
Richard Arndt
Cognos Corporation
Houston, TX
(713) 690-1105

AI SIG Liaison:
David Slater
Institute for Defense Anal
Alexandria, VA
(703) 845-2200

Datatrieve Liaison:
John Schutt
J. R. Simplot Company
Boise, ID
(208) 336-2110

DEC Counterpart:
Wendy Herman
Digital Equipment Corp
Nashua, NH
(603) 881-2494

Contributions

Contributions to the newsletter can be sent to either of the following addresses:

Editor, DMS SIG Newsletter c/o DECUS U.S. Chapter 219 Boston Post Road, BP02 Marlboro, MA 01752	Russ Poisson DMS SIG Newsletter Editor SEED Software Corporation 2121 Eisenhower Avenue Alexandria, VA 22314

Letters and articles for publication are requested from members of the SIG. They may include helpful hints, inquiries to other users, reports on SIG business, summaries of SPR's submitted to DEC, etc. Machine readable input is highly desirable.

Submitters should keep in mind the DECUS policy on commercialism.

A Proposal for Additional View Security for Rdb/VMS

Keith W. Hare
JCC
P.O. Box 381
Granville, Ohio 43023
614-587-0157

July 8, 1986

1 Introduction

In Rdb/VMS, views can be created to present subsets of data to the user. These views contain some sort of record selection expression based on values defined at view definition time and on values stored in the database. These views present a specific subset of the data to those users who have access to the views.

The security mechanism in Rdb, in conjunction with the VMS security, allows the database designer to limit a particular user, or class of users, in the access of views and relations. This mechanism works well and is fairly elegant.

However, in the current Rdb implementation, the database designer must create a separate view for each class of users that is to access a particular subset of the data. It is not possible to define a general purpose view that will present different subsets of the data to different users.

2 External values in Record Selection Expressions

The general purpose view described in the introduction could be created if a little more information was available to the RSE at runtime. The information needed at runtime is some parameter that is external to the database, such as a logical name, the current time, or a parameter passed from a program.

Time Time could be implemented with a function such as Rdb$CURRENT_TIME. This would allow a record selection expression to compare the current time with some table of times to determine which data should be presented. One use of this mechanism would be to restrict access to all data to a particular time range.

This function would also be useful for storing the current time when modifying or adding data to the database.

Logical Names Logical names could be implemented using the function Rdb$_LOGICAL_NAME(*log-name-table*) where *log-name-table* is the logical name table to be search for the translation. The logical name table could be any one of the standard tables or an application specific table. *Log-name-table* could also be a logical name, thereby allowing the application definer to create a search list of logical name tables.

While this mechanism would only provide a limited amount of security, depending on the logical name table used, it would provide a great deal of flexibility by deferring binding of certain values until runtime.

Usernames Usernames could be implemented using the function Rdb$USERNAME. This function would return the username of the current user. The query could then match this value against a value stored in the database to determine which information to present to the user.

This would also be useful for implementing some form of audit trail since Rdb does not currently support audit trails.

A number of other functions might also be useful. These include functions to retrieve process parameters, the ability to use parameters passed from a programming language or even user-written functions. The user-written functions and program parameters would provide a great deal of capability and flexibility but would probably be difficult to implement.

3 View Security Using External Values

The focus of this article is to argue that external values would be extremely useful in view security, in addition to their utility in other instances.

The primary utility of external values in views is to allow general purpose views to be created where part of the criteria is defined at runtime. This would allow a single view to be created to service multiple users rather than creating a separate view for each user. This would allow applications to be developed with greater generality .

The function Rdb$USERNAME would allow a view to be developed that included a specification of exactly which records a particular user could see. The username of the current user could be compared to a username stored in a record to give a unique definition of what records could be accessed or in some combination of relations.

The account budget view (figure 1) of the account budget relation (figure 2) provides an example of how the Rdb$USERNAME function might be used to present the user with a limited view of the data. This view implements a one-to-many relationship between the user and the account budget records.

A many-to-many relationship between users and budget records could be implemented using a user-account relation that consists of the account number and the username shown in figure 3.

```
define view account_budget_view
description is
{*
        This view shows a user's accounting budget information
*}
of ab in account_budget
        with ab.username eq Rdb$USERNAME.
        ab.account.
        ab.budget.
        ab.description.
end account_budget view.
```

Figure 1: Account Budget View

This feature would also allow for dynamic changes in protection schemes. A user could be added to the distribution list for some piece of data simply by making an entry in some sort of access relation.

The time and logical-name functions could be used in a similar manner to to determine what information is to be accessed. A time-access or logical-access relation could be changed as needed to reflect changes in policy or needs transparently to the database definitions, the applications and the users.

4 Function Evaluation Time

There are three possible times for the evaluation of these functions, database invoke, first access within a transaction and when the the rse is evaluated. Each of these would implement differing levels of dynamics, which might be equally appropriate depending on the application.

This argues that the function evaluation time should be specifiable in the view definition.

5 Errors

The functions described leave the possibility of errors at function evaluation time. Some consideration must be given to what should happen in the case of such errors.

The most consistent solution would be to return a null value in the case of an error and provide information for the application's exception handling mechanism. This would entail some additional exception messages but could easily be implemented in the context of the current on-error end-error construct.

Account	Budget	Description	...	Username
10110	500.00	Office Supplies	...	SMITH
10120	200.00	Miscellaneous	...	SMITH
10130	7500.00	Travel	...	SMITH
10140	5000.00	Slush Fund	...	SMITH
11110	400.00	Office Supplies	...	JONES
11120	100.00	Miscellaneous	...	JONES
11130	1000.00	Travel	...	JONES
12110	400.00	Office Supplies	...	BROWN
12120	100.00	Miscellaneous	...	BROWN
12130	1000.00	Travel	...	BROWN
...	...	...	...	...

Figure 2: Budget Relation

Account	Username
10110	SMITH
10110	JONES
10110	BROWN
10120	SMITH
10130	SMITH
10140	SMITH
10140	GREEN
...	...

Figure 3: Budget Relation

6 Conclusions

The ability to include an external value in a record selection expression provides the database designer with a great deal of flexibility. Among other possibilities, it allows for views to be defined so as to use a username as part of the criteria for accessing the data. This means that a general purpose view can be created that restricts a particular user to a particular subset of the data rather than having to create a view for each user.

EXPERIENCES IN SELECTING A RELATIONAL DATABASE MANAGEMENT SYSTEM FOR A VAX

Jessica Bondy and Sheryl Shipton
Medical Computing Center
University of Colorado Health Sciences Center
4200 East Ninth Avenue, Box B-119
Denver, CO 80262

In September 1985, we began the search for a database management system for our VAX/11-750, which runs VMS. The event that prompted this search was our decision to move database applications which had been developed on a Tandem NonStop II onto the VAX. Since the Tandem supports a truly relational (although unintegrated) database management system, we were restricted in our search to relational products. The only other firm requirement we had when starting our search was that the system be appropriate for end-users; we have a small staff and cannot offer programming services for either database application development or conversion.

Before starting our evaluations, we developed an elaborate set of weighted criteria, including sixteen required features, for rating the packages (see below). The criteria did not concern basic relational functionality, which we assumed would exist in a product which called itself relational, but with extra features. We expected to find many good products which not only met our required criteria but were more powerful and integrated than we were used to. After all, our main experience was with an older DBMS (circa 1977), created before relational databases were common, for use on a less popular computer.

ORIGINAL (NAIVE) VAX DBMS RATING SCALE

* Indicates the feature is required; any DB package lacking this feature is disqualified.

USER INTERFACE

*|__|..Access to all functions from a dumb terminal.

*|__|..Entry screen creation via painting.

|__|..Report screen creation via painting.

*|__|..Common language for creation, query, and reporting (package integration).

|__|..Record scrolling on a screen.

|__|..Screen query from multiple relations.

|__|..Updating multiple relations from one screen.

|__|..Multiple screen entry for one relation.

|__|..Update of multiple records by one command.

|__|..Deletion of multiple records by one command.

|__|..Batch mode with parameter passing to the database language.

|__|..Runs on PC as well as VAX.

DATA TYPES & DATABASE INTEGRITY

*|__|..Date and time data types and date/time arithmetic.
*|__|..Decimal data: fixed & floating point types.
|__|..Variable-length, large free text fields (graphics, text, etc.)
|__|..Non-contiguous keys.
|__|..Auto-sequenced fields.
|__|..Table translation of coded values.
*|__|..Data validation across fields.
*|__|..Data validation across relations.
*|__|..Handle missing data usefully.
|__|..Overriding of data validation.
|__|..Field-level security on input, output.
|__|..Record locking for multiple users.
|__|..Journaling & rollback of batch updates.
|__|..Journaling & rollback of logical transactions.

EXTERNAL INTERFACE

*|__|..Data importing.
*|__|..Data exporting.
|__|..Batch update of data dictionary.
*|__|..Export of dictionary data.
*|__|..Interface to in-house billing system via ^{2}MS accounting.

MODIFIABILITY OF DATABASE STRUCTURE

*|__|..Changing record definitions without requiring a reload of data.
|__|..Adding alternate keys, access methods.
|__|..Creation of new tables by extracting data from existing tables.

REPORTING FROM DATABASE

|__|..Automatic generation of simple reports.
|__|..Control over report format.
|__|..No restrictions on fields used in joining.
|__|..Support of external and inner joins.
|__|..Aggregation & other simple statistics.

ADVANCED PROGRAMMING

|__|..Using DB language within application language program.
|__|..Using application language routines within DB language programs.
|__|..Macros in the DB language.
|__|..Informative diagnostic messages.
|__|..Direct inspection of data files.

CAPACITY

*|__|..Large databases. (At least 100,000 records)
*|__|..Large records. (At least 2K bytes/record, 1500 attributes/record)

SUPPORT
*|__|..Expected survival of vendor.
|__|..Access to vendor help.

Although the original criteria we developed were useful in guiding our in-house evaluations, we soon discovered that they were the wrong criteria. It became apparent that finding a system which performed adequately on even the most central relational functions would be difficult. Accordingly, we developed a second set of criteria to determine quickly whether a package which purports to be relational in fact is. All of the criteria included in this second set got there because some package we had believed was relational could not handle them. The criteria and the packages we evaluated with them (See Table 1) will be discussed one by one, explaining the kinds of problems we encountered.

Note in Table 1 that the various DEC data management products were not evaluated, because we wanted an integrated system suitable for relatively unskilled users.

TABLE 1: DBMSs Evaluated

Product	Version	Company
ADT	2.4B	Interactive Systems
EMPRESS	1.4	Rhodnius
FOCUS	1.5(PC)	Information Builders
INFOCEN	11	3CI
INGRES	3.0/24A	Relational Technology
RIM	6.0	Boeing
SIR	2.1.3	SIR, Inc.
SYSTEM 1032	4.63 & 5.0	Software House

BARE ESSENTIAL DBMS CRITERIA

Criterion 1: Relations must have unique records based on a unique key index value.

For data to be in the normal form, they must be structured into a table with no repeating groups, homogeneous columns, and distinct rows. What makes the rows distinct is their key values. All of the products we evaluated allowed keying, but some could not conveniently enforce uniqueness. System 1032, for instance, required modifying the code to a default application generated to allow data entry through forms. The modified code would do a lookup on a record to be added to determine whether the attribute(s) named as the key already existed in a record in the table. An error message could be issued if a duplicate record was found. There was no way to force uniqueness when adding records via command mode.

Criterion 2: Key index field can consist of multiple fields anywhere in a record.

In order for a key to be unique, it usually will consist of several attributes each of which is meaningful in itself. For instance, a patient ID number might be unique with a single attribute (Social Security number, for example), but to guarantee uniqueness in a key for an office visit, the key would have to consist of several attributes: patient ID, date of visit, and physician ID.

Some systems, such as INGRES, limited the number of attributes that could comprise a key (in this case, to six). Others, such as System 1032, did not support keys consisting of multiple attributes at all. To create such an index would require storing the data twice -- once in combined form for the index and once in its parts.

The position of the attribute in a table determined whether it could be keyed in FOCUS. Keys could consist of multiple fields but had to be first N contiguous attributes in a record.

Criterion 3: Multiple relations can be joined without requiring one relation to be central, i.e., can join A--B--C--D.

Part of the appeal of relational databases is their great flexibility in extracting information. Unlike hierarchical databases, which are based on a central relation or "case", relational databases should allow relations to be joined in any fashion (sensible or not), so long as each pair of the tables to be joined has at least one attribute with datatypes in common. The packages we evaluated had several different kinds of problems with this criterion.

Some packages had trouble joining multiple relations. INFOCEN and RIM, for instance, simply could not handle non-procedural joins between more than two tables, except in a stepwise fashion. Users were expected to execute a series of joins between pairs of tables. Each join would create a new table, which could then be joined to the next, etc. Alternatively, users could write procedural code to join multiple tables directly, without the intermediate steps.

Other packages could join multiple relations, but only in certain configurations. These packages typically had been developed as hierarchical systems and were in transition to a relational implementation. This heritage showed itself sometimes in the inability to handle what we termed "chain" (vs. "star") joins. FOCUS, for instance, could join several relations only if one relation was central (join A, B, and C to D). If a chain of joins was required (join A to B, B to C, and C to D), a series of joins like those described for RIM and INFOCEN had to be performed.

SIR also had problems joining relations in certain configurations. In this case, the distinction was not between stars and chains, but between system-generated and user-defined links. SIR automatically creates hierarchical links between tables, based on the names and order of indexed attributes (see criterion 4). User-defined links do not operate the same way as do system-generated ones -- user-defined links are, for example, unidirectional, while system-generated links are bidirectional -- and had to be mentioned explicitly in a query in order to guarantee they would be used. In effect, all joins under SIR were not created equal; joining arbitrary tables outside the system-generated hierarchy was quite complex and messy.

Criterion 4: A join of any two relations can involve any fields in either relation, regardless of
- o whether the fields are part of a key
- o the position of the fields in the relation
- o the names given to the fields
- o the number of fields used in the join

As was mentioned above, relational databases should be very flexible with regard to how relations are joined. This flexibility pertains not just to the configuration of the relations, but to the fields as well. We were quite surprised by the kinds of constraints we encountered regarding which fields could be joined.

Several packages required that the fields in at least one of the relations being joined be keys. System 1032 and ADT were two such packages. One would expect a join between two non-keyed fields to be slow, since the tables would have to be searched sequentially for matches, but it should be possible.

The positions and names of attributes turned out to be important to joins in the SIR package. If attributes in different relations had the same names, were indexes, and appeared in the same N order, SIR automatically created static joins between the relations (another remnant from its hierarchical past). An unwary user could easily create unwanted joins based simply on his choice of attribute names and order, and superceding these automatic joins with an arbitrary, user-defined join was not easy.

Another permutation of the attribute name problem was perpetrated by RIM, which requires attributes used in a join to have the same name. RIM allows attributes to be renamed on the fly, but this should not be necessary.

Finally, and most importantly, some packages did not allow relations to be joined on more than one attribute. Given what we have already said about how keys often consist of multiple attributes, it is obvious that this lack is devastating to the utility of a package as a relational database manager. FOCUS was one product which could not use indexes to join on more than one field (in other words, such joins were implemented using sequential searches). Another, System 1032, could not handle joins on multiple attributes except by user-written procedures.

Criterion 5: All joins should use indexes to help locate records, whenever indexes are available.

It would seem obvious that using indexes whenever possible would improve a system's performance. Nevertheless, we encountered packages which did not. FOCUS, as mentioned above, had this problem. FOCUS provides two mechanisms for combining tables -- 'JOIN' which uses indexes but cannot apply to more than one attribute per table and 'MATCH" which does not use indexes, even when indexed fields are referenced.

.

Criterion 6: Joins of any two relations can be either "internal" (a record is included in the result when specified field values match in the two relations), or "external" (all records in each relation are included in the result, whether or not they match).

External joins are important for tracking down missing data. If the missing datum is an element of an existing record, listing cases with a specified value ("missing") for that attribute will be sufficient to document the state of the data. If, however, an entire record is missing, it cannot be listed. In this case only an external join will point out that data are missing.

Virtually all of the evaluated packages supported internal joins; fewer could perform external joins. ADT, EMPRESS, INFOCEN, and RIM could all handle external joins, although the EMPRESS syntax was somewhat awkward.

A variant of the external join, which we termed a "half external join", lists all of the records in one relation, along with the matching records in the second relation. Packages in which joins are unidirectional (joining A to B is not the same as joining B to A, and sometimes you can do only one at a time) necessarily cannot perform a full external join.

FOCUS, which always produced unidirectional joins, could manage, at best, a half external join when using indexes. It did, however,offer the option of perfórming a full outer join using 'MATCH', i.e., without benefit of indexes. SIR's user-defined paths were unidirectional, while its system-generated paths were bidirectional. As a result, one could produce a full external join on system-generated paths, a half external join on user-defined paths involving keyed fields, and only internal joins on non-keyed fields. Very confusing. Neither INGRES nor System 1032 could perform external joins without writing procedural code.

Criterion 7: All joins must be done without procedural programming code, such as IF-THEN tests on fields or records, incremental accumulation of summary records, loop and branching constructs, explicit calls to sort routines, etc.

This criterion may be somewhat more important to us than it would be to another site. As mentioned above, our users are generally not programmers and we do not have the resources to program for them. Nevertheless, a relational database manager should not require users to write code in order to perform a generalized functions like joins. That is why you are buying a package. Several packages -- notably INFOCEN, SIR, and System 1032 -- required substantial programming to accomplish what we considered basic relational functions.

What is the moral of this story? If you have determined that you really need a relational database manager, concentrate first on evaluating the relational functionality of the package. If it meets your needs in that respect, only then is it worthwhile to evaluate other features such as handling of missing data, data validation, package integration or other features important to your site's applications.

We found sales literature and even the answers of sales (and sometimes technical) personnel to be of little use in distinguishing packages. More than once, a technical person admitted to us that a product was hierarchical, after we had been told to the contrary by sales personnel, had asked specifically about the product's performance on the seven criteria described above, and had spent a week or so discovering this for ourselves. So, inevitably, you must work with the products in-house.

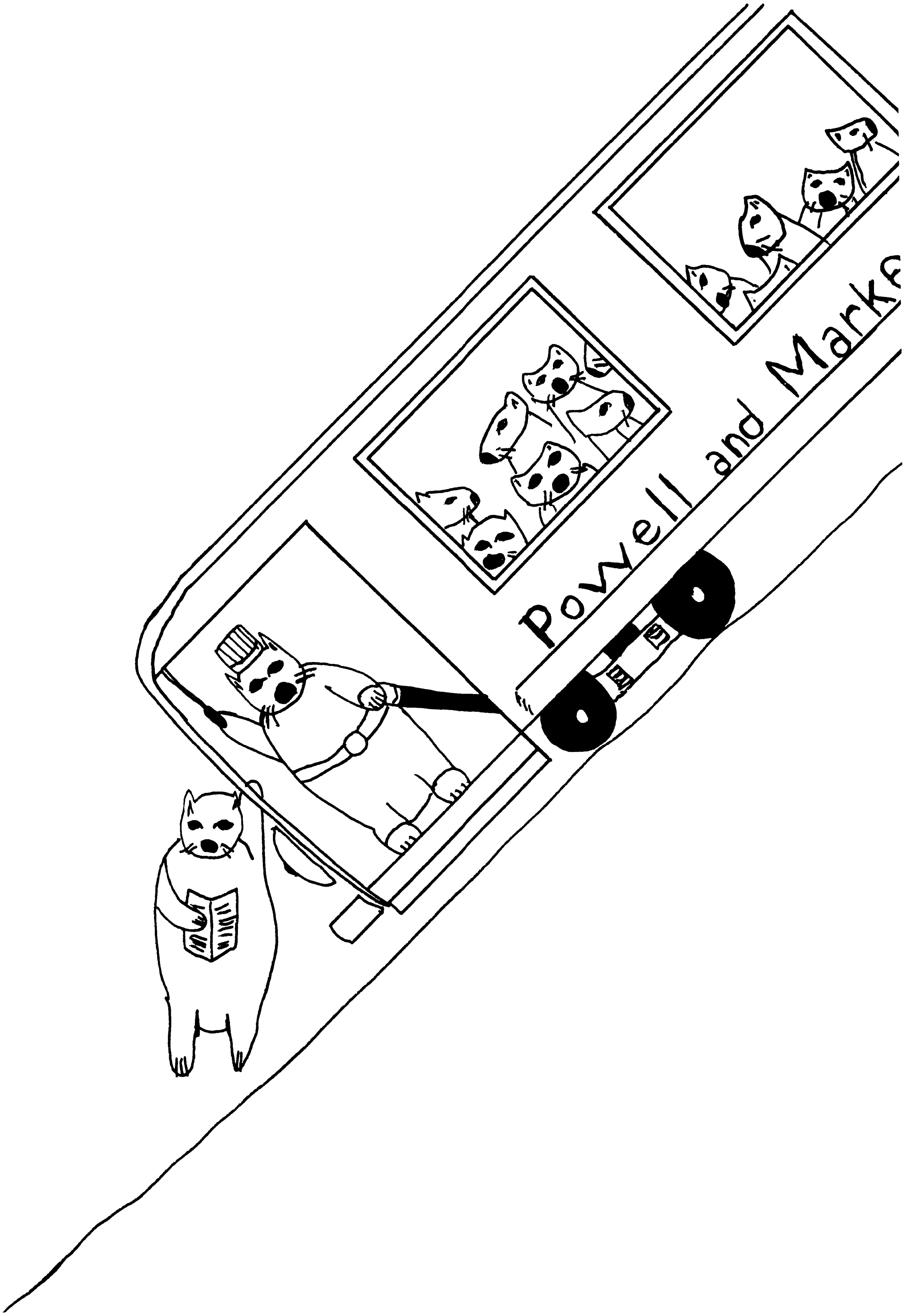
When you get a product in-house, you will find that predefined criteria are very useful in forcing evaluators to be even-handed and thorough. By all means, implement your own database on the systems you are evaluating. Make sure that it includes all of the features you want to test (multiple-attribute fields, external joins, or whatever). If you simply follow the tutorials provided by vendors, you will not discover the limits of the package, since tutorials are designed to show what the package can do, not what it can't.

And what package did we buy? Well, I guess it would be unfair to leave you in suspense. But remember that this field is in great flux (some of the packages were undergoing major rewrites as we evaluated them) and our needs were shaped in no small degree by the features of the Tandem system from which we were converting. Ultimately, we found EMPRESS to be the package that best suited our needs.
$

The Wombat EXAMINER & 4GL Dispatch

"Increases the Circulation of Anyone in America"

Volume 8 Number 1



DTR

DTR/4GL Special Interest Group - Officers

SIG Chairman
Joe H. Gallagher
Research Medical Center
2316 East Meyer Blvd
Kansas City, MO 64132
816-276-4235

Past SIG Chairman - Vol. Coord.
Larry Jasmann
U. S. Coast Guard
10067 Marshall Pond Rd.
Burke, VA 22015
202-426-2344

Asst. Volunt. Coord.
Susan Krentz
NKF Engineering, Inc.
12200 Sunrise Valey Dr.
Reston, VA 22091
703-620-0900

Campground
Bert Roseberry
c/o U. S. Coast Guard
500 Camp Street
New Orleans, LA 70130
504-589-4934

DEC Counterpart - Datatrieve
Andy Schneider
Digital Equipment Corp.
110 SpitBrook ZK02-2/N59
Naushua, NH 03062

Editor & Comm. Rep.
Donald E. Stern, Jr.
Warner Lambert Company
10 Webster Road
Milford, CT. 06460
203-783-0238

Production Editor
Steve Cordiviola
Kentucky Geological Survey
311 Breckinridge Hall
Lexington, KY 40506
606-257-5863

Library Comm. Rep & Artist
Bart Z. Lederman
Consulting Engineer
2572 E. 22nd St.
Brooklyn, NY 11235
718-743-9593

WW Editor & PIR Coordinator
Philip A. Naecker
Consulting Engineer
3011 N. Mount Curve Ave
Altadena, CA 91001
818-791-0945

DEC Counterpart - RALLY, TEAMDATA
Basil Harris, Jr.
Digital Equipment Corp.
110 Spitbrook ZK02-2K29
Nashua, NH 03062-2698

Symposia Comm. Rep.
Chris Wool
E.I. duPont Eng. Dept.
Louviers Bldg.
Wilmington, DE 19898
302-366-4610

Symposia Asst. & New Prod. Coord.
Diane Pinney
Computer Sciences Corp.
443 Inyokern Road
Ridgecrest, CA 93555
619-446-6585x284

Pre-Symposia Seminars
Dana Schwartz

15719 Millbrook Lane
Laurel, MD 20707
301-859-6277

Session Notes
Sonia Anderson
SRI International MS:PN341
333 Ravenswood Avenue
Menlo Park, CA 94025
415-859-2577

Special Effects
Alex L. Lamb
U.S. Army
32 Pearce Ave.
Eatontown, NJ 07724
201-532-3318x1843

DTR/4GL Masters List

2-26-86

VAX-11 Datatrieve -

Joe H. Gallagher	4GL Solutions	10308 Metcalf Suite 109	Overland Park, KS 66212	913-894-9550
Bart Z. Lederman		2572 E. 22nd Street	Brooklyn, NY 11235	718-743-9593
Joe Kelly	WymAn Gordon Co.	Worchester Road	North Grafton, MA 01536	617-839-4411x5480
Larry Jasmann	U. S. Coast Guard	10067 Marshall Pond Rd.	Burke, VA 20015	202-426-2344
James Swanger	G. D. Searle & Co.	4901 Searle Parkway	Skokie, IL 60077	312-982-7430
Chris Wool	E. I. duPont, Engineering Dept.	Louviers Bldg.	Wilmington, DE 19898	302-366-4610
Dick Azzi	Motorola SPS	Box 2953 MS: P116	Phoenix, AZ 85062	602-244-4316
Chris Heinz	American Board of Family Practice	2228 Younge Drive	Lexington, KY 40505	606-269-5626
Bert Roseberry	U. S. Coast Guard	500 Camp Street	New Orleans, LA 70130	504-589-4934
Donald E. Stern	Warner Lambert Company	10 Webster Road	Milford, CT 06460	203-783-0238

Pro-Datatrieve -

Bart Z. Lederman		2572 E. 22nd Street	Brooklyn, NY 11235	718-743-9593
Joe H. Gallagher	4GL Solutions	10308 Metcalf Suite 109	Overland Park, KS 66212	913-894-9550

Datatrieve-11 (RSX) -

Bart Z. Lederman		2572 E. 22nd Street	Brooklyn, NY 11235	718-743-9593

Contributions

Submissions to this newsletter are constantly sought. A submission can be an article, a letter to the Wombat Wizard, a technical tip, or anything of interest to people using or considering the use of Datatrieve or any 4GL product. Submissions on magnetic media are preferred but almost any type will be considered.

Contributions for the newsletter can be sent to either of the following addresses:

Editor, DATATRIEVE Newsletter
c/o DECUS U.S. Chapter
219 Boston Post Road, BP02
Marlboro, MA 01752

Donald E. Stern, Jr
Warner Lambert Company
10 Webster Road
Milford, CT 06460

Table of Contents

DECUS U.S. Chapter SIG Newsletters, Volume 2, No. 1, Sept. 1986
Wombat Examiner and 4GL Dispatch, Volume 8, No. 1

About the Cover: "Wombats Enjoying the San Francisco Sights"

Chairman's Corner

Joe H. Gallagher, 4GL Solutions, Overland Park, KS

When your boss says, "Why should I send you to San Francisco to the DECUS meeting? You just want to go to have a good time." Here's a list of some of the things you can say in reply.

* I will probably learn enough new things about DATATRIEVE (or substitute one of your favorite languages) that the trip will probably pay for itself.

* I will be able to solve many of the problems that I have had at the DATATRIEVE clinic by asking developers and experienced users questions.

* If I listen carefully, I will be able to know what Digital is going to do in the future. Thus, we will be in a better position over our competitors to take advantage of new technology.

* By visiting the exhibit hall, I will be able to see in one place all the new hardware and get demos on the new software that we are considering. This will save time and money since I won't have to run around to several places to see it all AND get my question answered by experts. I can also get a whole lot of free publications on hardware, software, architecture, and more.

* I will get to attend more than forty hours of sessions on topics that I need training on. The cost of the Symposia is less than the corresponding number of days of training classes.

* I'll also be able to stop by DEXPO and see the third party hardware and software products that you wanted me to evaluate.

If all that doesn't convince him (or her), tell your boss that you will bring back a set of buttons and T-shirts for him or her. If you don't attend the Fall Symposium in San Francisco you will miss a great chance to learn a whole lot about DATATRIEVE, RALLY/TEAMDATA, and other DEC and non-DEC 4GL products. There will be several sets of sessions on DATATRIEVE which will be more advanced than have been previously presented at a symposium. So don't miss out. See you in San Francisco.

Wombat Wizard
Edited by - Philip A. Naecker, Consulting Software Engineer

Dear Readers:

I want to start this month's columns by delivering on some promises I've made over the previous few articles. Here are a couple of Waves of the Wiz's Wand:

Passing Parameters to DCL from DTR

In the July Wombat Examiner I discussed using the FN$TRANS_LOG user defined function to pass information from DCL to DTR. I also mentioned that a similar technique might be used in the reverse direction. The FN$CREATE_LOG function is provided as one of the standard functions in DTR and provides a means of creating a logical from Datatrieve. But the logicals created with FN$CREATE_LOG are user-mode logicals, and as such they are lost when the DTR image runs down (exits.) So to create logicals that exist beyond the DTR image, you need to be able to create a logical in supervisor mode.

The VMS RTL call LIB$SET_LOGICAL does just that. It is defined in the following fashion.

```
; FN$DEFINE_LOG - Define a logical name
;
; no output
; Input is a string descriptor of the logical name

$DTR$FUN_DEF FN$DEFINE_LOG, LIB$SET_LOGICAL, 2
    $DTR$FUN_OUT_ARG  TYPE = FUN$K_STATUS
    $DTR$FUN_NOVALUE
    $DTR$FUN_NOOPTIMIZE
    $DTR$FUN_IN_ARG  TYPE = FUN$K_DESC, DTYPE = DSC$K_DTYPE_T,
       ORDER = 1
    $DTR$FUN_IN_ARG  TYPE = FUN$K_DESC, DTYPE = DSC$K_DTYPE_T,
       ORDER = 2
$DTR$FUN_END_DEF
```

Inside of DTR, you might use it like this:

```
$ DTR EXECUTE procedure
DTR> EXECUTE procedure
...
DTR> SET ABORT
DTR> FN$DEFINE_LOG("DTR_STATUS","FAILURE")
...
DTR> IF 'foo' NE 'bar' then FN$DEFINE_LOG("DTR_STATUS",
       "SUCCESS") ELSE ABORT
...
DTR> EXIT
$ IF F$TRNLNM("DTR_STATUS") .NES. "SUCCESS" THEN -
       GOTO ERROR_IN_DTR
...
```

The exact usage is not important, but this function overcomes a serious limitation of DTR without it - all exits from DTR are "successful." That is, if you check the DCL variable $STATUS upon exiting DTR, you will find that it contains a success status, even if you executed an ABORT statement or encountered some other error - such as "File not found" - in the procedure you ran. This function allows you to explicitly pass information to DCL about the status of your DTR procedures. For example, you might want to periodically update the value of a logical as you reach various steps in your procedure, much like the DCL restart function.

For those of you who are especially adventurous, you might want to include the argument to LIB$SET_LOGICAL that allows you to specify the logical name table. By omitting this argument, we've forced LIB$SET_LOGICAL to put our logical names in LNM$PROCESS, but if we had that argument we could put them in any table we have write access to. (Of course, that would be yet another argument to type in, which is why I left it out in this simple example.)

Translating EBCDIC Data in DTR

In that same July issue, I discussed using STR$TRANSLATE to convert between alphabets. A much easier way is to use the RTL functions provided. Besides, the method I gave before doesn't work very well, and only with a great deal of hassle.

```
; FN$STR_EBCDIC_TO_ASCII - Translate EBCDIC to ASCII
;
; Output is a string
; Input is an output string descriptor,
; an input string descriptor,

$DTR$FUN_DEF FN$STR_EBCDIC_TO_ASCII, LIB$TRA_EBC_ASC, 2
    $DTR$FUN_OUT_ARG  TYPE = FUN$K_STATUS
    $DTR$FUN_HEADER  HDR = <"ASCII"/"String">
    $DTR$FUN_IN_ARG  TYPE = FUN$K_DESC, DTYPE = DSC$K_DTYPE_T,
       ORDER = 1
    $DTR$FUN_IN_ARG  TYPE = FUN$K_TEXT , OUT_PUT = TRUE
$DTR$FUN_END_DEF

; FN$STR_ASCII_TO_EBCDIC - Translate ASCII to EBCDIC
;
; Output is a string
; Input is an output string descriptor,
; an input string descriptor,

$DTR$FUN_DEF FN$STR_ASCII_TO_EBCDIC, LIB$TRA_ASC_EBC, 2
    $DTR$FUN_OUT_ARG  TYPE = FUN$K_STATUS
    $DTR$FUN_HEADER  HDR = <"EBCDIC"/"String">
    $DTR$FUN_IN_ARG  TYPE = FUN$K_DESC, DTYPE = DSC$K_DTYPE_T,
       ORDER = 1
    $DTR$FUN_IN_ARG  TYPE = FUN$K_TEXT , OUT_PUT = TRUE
$DTR$FUN_END_DEF

DTR> PRINT FN$STR_EBCDIC_TO_ASCII (EBCDIC_STRING)

            ASCII
           String

This string was originally in EBCDIC.
```

Printing Page Breaks at Convenient Locations

In the June issue I discussed using a prompt to cause the screen to hold at the end of a page, rather than scrolling by. The basic trick is this:

```
 AT BOTTOM OF PAGE -
    PRINT " "|*."any character and RETURN to continue" USING X
```

It works nice, but, as Richard J. Trilling and Debra Mangum of Environmental Monitoring and Services, Inc. points out, it has the nasty side effect of preventing you from using the same report in both batch and interactive mode. They proposed a fix that involves examining the contents of the output file logical for the string "TT:", but that seems to me to have some difficulties. Instead, we can use the parameter passing trick from the same issue and the DCL function F$MODE to tell DTR our execution mode. So, with an enhancement or three by the Wiz, here is Richard and Debra's solution.

```
$ IF F$MODE() .EQS. "BATCH" THEN -
       DEFINE/USER/NOLOG DTR_EXECUTION_MODE BATCH
$ DTR EXECUTE REPORT_PROCEDURE
...
DECLARE NUMBER_OF_LINES USAGE WORD.
DECLARE INTERACTIVE PIC X.
!
IF FN$TRANS_LOG("DTR_EXECUTION_MODE") EQ "BATCH" THEN
       BEGIN
               INTERACTIVE = "N"
               NUMBER_OF_LINES = 43
       END ELSE BEGIN
               INTERACTIVE = "Y"
               NUMBER_OF_LINES = 8
       END
! 8 and 43 allow room for headers, etc., and correspond to
! interactive and batch mode, respectively
!
DECLARE PAUSE COMPUTED BY CHOICE
       INTERACTIVE EQ "Y" THEN
         " "|*.any character and CR to continue"
       ELSE " "
END_CHOICE.

DECLARE PAGE_CONTROL_GROUP COMPUTED BY
       FN$NINT(RUNNING COUNT/NUMBER_OF_LINES).

REPORT whatever ON
       *."report destination (e.g., file, TT: for term., etc.)"
...report statements...
AT BOTTOM OF PAGE_CONTROL_GROUP PRINT PAUSE(-) USING X, NEW_PAGE
END_REPORT
```

Note that although a user need not define the logical DTR_EXECUTION_MODE if they are running interactively, it must be defined in any procedures which execute this procedure from a batch command file. However, if the same command file is used to execute the procedure interactively and in batch, it will work correctly in both cases.

Dear WW:

The default Date Format that DEC uses is RIDICULOUS. The contortions that one has to go through to use an alternative date like NN/DD/YY (with the slash marks already embedded on a form) are unsatisfactory. I.e., read the date into a text variable, feed it into a usage date field, then display it using a computed by format statement.

Signed,
Ticked at Dates

Dear Ticked:

I don't think it's really quite as difficult as you seem to think, but I grant it could be easier. (There are only two steps, instead of the three you mention, but who's counting?) There is one solution that I can offer, however. DTR is completely customizable in virtually everything you see except the syntax. You can change the help files, the names of keywords, the error messages, and - YES! - the output formats. (If you stop to think about it, it only makes sense to be able to change the date output format - after all, they use different formats in different languages, and DTR is completely customizable from a language point of view.)

See the Guide to Programming and Customizing for more information, but the basic idea is this:

1. Locate the file DTR$LIBRARY:DTRTEXT.MAR. Editing it and committing the edits to the DTR shareable image is described in Chapter 10 of the Guide.

2. I believe the variable you want to change is DTR$$K_TXT_DAT_PIC. Change it from:

```
       $DTR$TEXT_ENTRY  DTR$$K_TXT_DAT_PIC,-
               ^\DD-MMM-YYYY?"  "\
```

to:

```
       $DTR$TEXT_ENTRY  DTR$$K_TXT_DAT_PIC,-
               ^\NN/DD/YY?" "\
```

By the way, you can also use this same technique to uppercase the names of the weekdays if this has been a source of difficulty for you. (See the Wombat Magic in the July issue of the Examiner.)

3. Build a new version of DTR. If you want, you can make a separate, different version of DTR with these changes in it, so it won't affect other users. Just be aware that if you have two shareable images, even if both are installed, paging performance may suffer. (If you don't know what I mean by this, come hear my talk on DTR performance at the San Francisco Symposium next month!)

Dear WW:

The examples in the DTR documentation only show how to print the elements of a DTR hierarchical element. These are not adequate to help understand how to do other things such as modify. For example, how do I modify KIDS without doing a FIND FAMILIES then a FIND KIDS;SELECT;. It took me quite a while to figure this out and I often need to refer back.

Signed,
Hurt by Hierarchies

Dear Hurt:

Let's see if I can help a little. First, there are three ways to deal with hierarchies, and none of them involve FINDs, except in quite unusual circumstances. The first way (and easiest except for very large domains) is to "flatten the hierarchy." This is done with a special case of the CROSS clause, and allows you to deal with the list elements as if they were primary elements of the record. See the example below.

The second way, and a better performer if you have a large domain, is to simply use nested FOR statements to access the record. Again, the example below should help.


```
DTR> FN$CREATE_LOG("DTR$LIBRARY","[PAN.DECUS.WW.EXAMPLES]")
DTR> set def cdd$top.dtr$lib.demo
DTR> show families
DOMAIN FAMILIES
 USING FAMILY_REC ON DTR$LIBRARY:FAMILY.DAT;

DTR> show family-rec
RECORD FAMILY_REC
01 FAMILY.
   03 PARENTS.
      06 FATHER PIC X(10).
      06 MOTHER PIC X(10).
   03 NUMBER_KIDS PIC 99 EDIT_STRING IS Z9.
   03 KIDS OCCURS 0 TO 10 TIMES DEPENDING ON NUMBER_KIDS.
      06 EACH_KID.
         09 KID_NAME PIC X(10) QUERY_NAME IS KID.
         09 AGE PIC 99 EDIT_STRING IS Z9.
;

DTR> ready families write
DTR> print count of families

COUNT

    14

DTR> print first 2 families

                                NUMBER    KID
  FATHER     MOTHER     KIDS      NAME      AGE

JIM        ANN            2    URSULA       7    Note that DTR auto-
                               RALPH        3    matically formats
JIM        LOUISE         5    ANNE        31    the lists "pretty"
                               JIM         29
                               ELLEN       26
                               DAVID       24
                               ROBERT      16

DTR> show fields
FAMILIES
   FAMILY
      PARENTS
         FATHER         <Character string>
         MOTHER         <Character string>
      NUMBER_KIDS       <Number>
      KIDS      <List>
         EACH_KID
            KID_NAME (KID)      <Character string>
            AGE         <Number>
No global variables are declared.

DTR> find a in families cross kids     !This is a special case
```

```
[35 records found]                    !of the CROSS clause.  It
DTR> list first 2 families            !"flattens" the hierarchy

FATHER       : JIM     !Note the difference between this list
MOTHER       : ANN
NUMBER_KIDS :  2
    KID_NAME    : URSULA
    AGE         :  7
    KID_NAME    : RALPH
    AGE         :  3

FATHER       : JIM
MOTHER       : LOUISE
NUMBER_KIDS :  5
    KID_NAME    : ANNE
    AGE         : 31
    KID_NAME    : JIM
    AGE         : 29
    KID_NAME    : ELLEN
    AGE         : 26
    KID_NAME    : DAVID
    AGE         : 24
    KID_NAME    : ROBERT
    AGE         : 16


DTR> list first 2 a

FATHER       : JIM                 !...and THIS list...
MOTHER       : ANN
NUMBER_KIDS :  2
    KID_NAME    : URSULA
    AGE         :  7
    KID_NAME    : RALPH
    AGE         :  3
KID_NAME    : URSULA               !This is the "flattened" part
AGE         :  7                   !of the hierarchy.  Note
                                   !that it repeats some data!
FATHER       : JIM
MOTHER       : ANN
NUMBER_KIDS :  2
    KID_NAME    : URSULA
    AGE         :  7
    KID_NAME    : RALPH
    AGE         :  3
KID_NAME    : RALPH
AGE         :  3

 DTR> print father, kid-name of families
"KID_NAME" is undefined or used out of context.
                          !In the hierarchy, a seemingly innocent
DTR> print father, kid-name of first 2 a  !statement fails...
```

```
                KID
  FATHER        NAME

JIM         URSULA      !...but here it works,
JIM         RALPH       !because we've "flattened" it.

DTR> for a with kid-name eq robert print each-kid then
       modify age
"ROBERT" not field, assumed literal.

   KID         !This statement would not
   NAME    AGE !have worked on FAMILIES...

ROBERT     23
Enter AGE: 23
DTR> ! Of course, we can also get to the data the other way...
DTR> for f in families
[Looking for statement]
CON>   for k in kids with kid-name eq robert
[Looking for statement]
CON>           print then modify age
"ROBERT" not field, assumed literal.

   KID
   NAME    AGE

ROBERT     23
Enter AGE: 19
```

Now, there is always a good way to solve any problem in Datatrieve - that is to NOT CAUSE THE PROBLEM IN THE FIRST PLACE. In the case of hierarchies, the problem is not just DTR, it's that hierarchies are inherently nasty things. Personally, I avoid using them at all costs. Instead, you should fully normalize your data into the underlying natural relations. In the case of the FAMILIES domain, the underlying natural relations are the PARENTS and the KIDS. Since there are a variable number of kids for each family, and since the two things are fundamentally different, there is really no good reason to store them in the same domain (or relation.)

Instead, store the parents in a PARENTS relation and the kids in a KIDS relation. I've got a few examples of this here. In the examples that follow, typed input is shown in lowercase to aid readability.

```
DTR> ! Ready the demo files, but don't use the actual files
DTR> ! because we may wish to modify them.
DTR> fn$create_log("DTR$LIBRARY",
       "USER_DISK:[PAN.DECUS.WW.EXAMPLES]")
DTR> set dictionary cdd$top.dtr$lib.demo
DTR> show domains
```

```
Domains:
     ANNUAL_REPORT;1 FAMILIES;1      KETCHES;1       OWNERS;1
     OWNERS_SEQUENTIAL;1             PAYABLES;1      PERSONNEL;1
     PETS;1          PROJECTS;1      SAILBOATS;1     SALES;1
     YACHTS;1        YACHTS_SEQUENTIAL;1

DTR> ready families exclusive !Always use EXCLUSIVE if you can -
                              !for performance
DTR> show fields
FAMILIES
   FAMILY
      PARENTS
         FATHER         <Character string>
         MOTHER         <Character string>
      NUMBER_KIDS       <Number>
      KIDS      <List>
         EACH_KID
            KID_NAME (KID)      <Character string>
            AGE         <Number>
No global variables are declared.

DTR> print first 3 families

                           NUMBER    KID
  FATHER      MOTHER        KIDS     NAME     AGE

JIM         ANN              2    URSULA       7
                                  RALPH        3
JIM         LOUISE           5    ANNE        31
                                  JIM         29
                                  ELLEN       26
                                  DAVID       24
                                  ROBERT      19
JOHN        JULIE            2    ANN         29
                                  JEAN        26

DTR> show families
DOMAIN FAMILIES
 USING FAMILY_REC ON DTR$LIBRARY:FAMILY.DAT;

DTR> show family_rec
RECORD FAMILY_REC
01 FAMILY.
   03 PARENTS.
      06 FATHER PIC X(10).
      06 MOTHER PIC X(10).
   03 NUMBER_KIDS PIC 99 EDIT_STRING IS Z9.
   03 KIDS OCCURS 0 TO 10 TIMES DEPENDING ON NUMBER_KIDS.
      06 EACH_KID.
         09 KID_NAME PIC X(10) QUERY_NAME IS KID.
         09 AGE PIC 99 EDIT_STRING IS Z9.
```

```
DTR> !Now we're going to  define a VIEW DOMAIN with the same data
DTR> !as FAMILIES.  We'll first define  the two domains that will
DTR> !make up the view...
DTR> set dictionary cdd$default
DTR> define domain parents using parent on parents.dat;
DTR> define record parent using
DFN> 01 parent_data.
DFN>   03      family_id usage long
DFN>           query_name is fid.
DFN>   03      father pic x(10)
DFN>           query_name is dad.
DFN>   03      mother pic x(10)
DFN>           query_name is mom.
DFN>   03      filler.
DFN>           05      number_of_kids computed by
DFN>                   count of kids with fid = parent.fid
DFN>                   query_name is number_kids.
DFN> ;
[Record is 24 bytes long.]

DTR> !The FILLER field in the previous record  is a way to "hide"
DTR> !the  lower  level  field(s)  (in this case, NUMBER_OF_KIDS)
DTR> !from the users.   It will only be printed or otherwise used
DTR> !if specifically referenced.
DTR> !
DTR> define domain kids using kid on kids.dat;
DTR> define record kid using
DFN> 01 each_kid.
DFN>   03      family_id usage long
DFN>           query_name is fid.
DFN>   03      kid_name pic x(10) query_name is child.
DFN>   03      age pic 99 edit_string is z9.
DFN>   03      filler.
DFN>           05      mom computed by mom from parents with
DFN>                   fid = kid.fid.
DFN>           05      dad computed by dad from parents with
DFN>                   fid = kid.fid.
DFN> ;
[Record is 16 bytes long.]
DTR>
DTR> define file for parents key = fid(no dup, no change);
DTR> define file for kids key=fid(dup, no change);
DTR> !
DTR> !Use an ALLOCATION clause  on  DEFINE  FILE  if you know the
DTR> !file is going to be large.  Also, if  the  file is going to
DTR> !be  large, you will be better off defining the file WITHOUT
DTR> !keys, storing the data, then going back and adding the keys
DTR> !and  using  CONVERT  to  move the  data from  the  previous
DTR> !version of the file to the new, keyed version.   See June's
DTR> !WW for an example of this.
DTR> !
DTR> ready parents as new_parents exclusive write
DTR> ready kids as new_kids exclusive write
DTR> show fields
NEW_KIDS
   EACH_KID
      FAMILY_ID (FID)  <Number, indexed key>
      KID_NAME (CHILD) <Character string>
      AGE      <Number>
NEW_PARENTS
   PARENT_DATA
      FAMILY_ID (FID)  <Number, indexed key>
      FATHER (DAD)     <Character string>
      MOTHER (MOM)     <Character string>
FAMILIES
   FAMILY
      PARENTS
         FATHER        <Character string>
         MOTHER        <Character string>
      NUMBER_KIDS      <Number>
      KIDS     <List>
         EACH_KID
            KID_NAME (KID)     <Character string>
            AGE        <Number>
No global variables are declared.
```

```
DTR> !This procedure will store data from the demo FAMILIES
DTR> !domain into our new, normalized domains.
DTR> !
DTR> for f in families begin
CON>   declare id long.
CON>   id = running count
CON>   store new_parents using begin
CON>           father = f.father
CON>           mother = f.mother
CON>           fid = id
CON>   end
CON>   for k in kids begin
CON>           store new_kids using begin
CON>                   fid = id
CON>                   kid-name = k.kid-name
CON>                   age = k.age
CON>           end
CON>   end
CON> print "Family number", ID(-), "stored"
CON> end
Family number          1 stored
Family number          2 stored
Family number          3 stored
...
```

```
DTR> print first 3 new_parents

  FAMILY
    ID         FATHER     MOTHER

         1 JIM         ANN
         2 JIM         LOUISE
         3 JOHN        JULIE

DTR> print first 7 new_kids

  FAMILY       KID
    ID         NAME    AGE

         1 URSULA       7
         1 RALPH        3
         2 ANNE        31
         2 JIM         29
         2 ELLEN       26
         2 DAVID       24
         2 ROBERT      19
```

So, with two separate sources of data (PARENTS and KIDS), we perform any MODIFY or STORE operations on each part separately and thus avoid the difficulties of using hierarchies. If we need to get information about the two domains together, we can use the virtual fields we've defined or we can make a VIEW. And since oftentimes we really need to operate on just KIDS or just PARENTS, we avoid unnecessary complexity.

Everything seems pretty nice until this point. But there are some problems associated with the clever trick we've done with our COMPUTED BY fields in the PARENT and KID records. In particular, because they reference domains explicitly, we need to make sure those domains are READIED before we reference those files. So that makes it difficult if we use alias clauses (READY domain-name AS alias-name.) If we try to reference those COMPUTED BY FIELDS (NUMBER_OF_KIDS in NEW_PARENTS and MOM or DAD in KIDS), we'll get errors because PARENTS and KIDS aren't ready at this time. Only their aliases are ready. For example:

```
DTR> print mom, dad, number_of_kids of first 3 new_parents
"KID" is undefined or used out of context.     !Because NEW_KIDS
                                               !is ready....
DTR> print mom, dad, kid-name, age of first 7 new_kids
"PARENTS" is not a readied source, collection, or list. !likewise
                                               !  for NEW_PARENTS
```

Now, we can generally fix this part of the problem simply by readying the domains without any alias clause. For example:

```
DTR> finish new_parents, new_kids
DTR> ready parents shared
DTR> ready kids shared
DTR> print mom, dad, number_of_kids of first 3 parents

                      NUMBER
                        OF
  MOTHER     FATHER    KIDS

ANN        JIM            2
LOUISE     JIM            5
JULIE      JOHN           2

DTR> print mom, dad, kid-name, age of first 7 kids

                         KID
   MOM        DAD        NAME    AGE

ANN        JIM        URSULA      7
ANN        JIM        RALPH       3
LOUISE     JIM        ANNE       31
LOUISE     JIM        JIM        29
LOUISE     JIM        ELLEN      26
LOUISE     JIM        DAVID      24
LOUISE     JIM        ROBERT     19
```

But look what happens if we choose to re-READY the PARENTS domain.

```
DTR> finish parents
DTR> ready parents shared
DTR> print mom, dad, number_of_kids of first 3 parents

                      NUMBER
                        OF
  MOTHER     FATHER    KIDS

ANN        JIM            2   !No problem here...
LOUISE     JIM            5
JULIE      JOHN           2

DTR> print mom, dad, kid-name, age of first 7 kids
Internal error (expected block id 4, encountered id 79).! BLAM!
```

We can fix it by in turn FINISHing KIDS and re-READYing it, but in general we will have problems.

```
DTR> finish kids
DTR> ready kids shared
DTR> print mom, dad, kid-name, age of first 7 kids
```

```
                         KID
   MOM        DAD        NAME    AGE

ANN        JIM        URSULA      7
ANN        JIM        RALPH       3
LOUISE     JIM        ANNE       31
LOUISE     JIM        JIM        29
LOUISE     JIM        ELLEN      26
LOUISE     JIM        DAVID      24
LOUISE     JIM        ROBERT     19
```

These problems are caused by the circular dependency between the NUMBER_OF_KIDS field in PARENTS and the MOM and DAD fields in KIDS. Each requires the other be ready first, and it is obviously impossible that TWO things be ready FIRST. So, in general, we should try to avoid the use of the circular references we have made. However, as was just shown, it can often work and be very useful.

Now we've found we can address the parts of the old FAMILIES domain as separate pieces (fully normalized, with some references between them to make things a little simpler). But sometimes you really do want to have a single, hierarchical object. No problem, just define a VIEW.

```
DTR> define domain families_view of parents, kids by
DFN> 01        family_unit occurs for parents.
DFN>   03      fid   from parents.
DFN>   03      father  from parents.
DFN>   03      mother  from parents.
DFN>   03      children occurs for kids with fid = parents.fid.
DFN>           05      kid_name from kids.
DFN>           05      age from kids.
DFN> ;
DTR>
DTR> finish
DTR> ready families_view shared
DTR> ready cdd$top.dtr$lib.demo.families as hierarchy_families
DTR> show ready
Ready sources:
   HIERARCHY_FAMILIES:  Domain, RMS sequential, protected read
           <_CDD$TOP.DTR$LIB.DEMO.FAMILIES;1>
   FAMILIES_VIEW:  Domain, VIEW, shared read
           <_CDD$TOP.USER_CDD.PAN.WIZARD.FAMILIES_VIEW;1>
No loaded tables.

DTR> ! Just to show you that the two domains (the view and the
DTR> ! hierarchy) really contain the same thing...
DTR> show fields
HIERARCHY_FAMILIES
   FAMILY
      PARENTS
         FATHER        <Character string>
         MOTHER        <Character string>
      NUMBER_KIDS      <Number>
      KIDS      <List>
         EACH_KID
            KID_NAME (KID)     <Character string>
            AGE        <Number>
FAMILIES_VIEW
   FAMILY_UNIT
      FAMILY_ID (FID)  <Number, indexed key>
      FATHER (DAD)     <Character string>
      MOTHER (MOM)     <Character string>
      CHILDREN <List>
         KID_NAME (CHILD)      <Character string>
         AGE    <Number>
No global variables are declared.

DTR> print first 2 families_view

  FAMILY                             KID
    ID       FATHER     MOTHER       NAME    AGE

          1 JIM        ANN        URSULA      7
                                  RALPH       3
          2 JIM        LOUISE     ANNE       31
                                  JIM        29
                                  ELLEN      26
                                  DAVID      24
                                  ROBERT     19

DTR> print first 2 hierarchy_families

                      NUMBER    KID
  FATHER     MOTHER    KIDS     NAME    AGE

JIM        ANN          2   URSULA      7
                            RALPH       3
JIM        LOUISE       5   ANNE       31
                            JIM        29
                            ELLEN      26
                            DAVID      24
                            ROBERT     19
```

If you don't like the FAMILY_ID showing up, you can "hide" it by using the FILLER trick we used earlier.

Now, there are some things you'll want to do using the separate parts of the view and some things you'll want to do with the entire view. Here are some examples:

```
DTR> ready kids shared
DTR> ready parents shared
DTR> show ready
Ready sources:
```

```
   PARENTS:  Domain, RMS indexed, shared read
           <_CDD$TOP.USER_CDD.PAN.WIZARD.PARENTS;1>
   KIDS:  Domain, RMS indexed, shared read
           <_CDD$TOP.USER_CDD.PAN.WIZARD.KIDS;1>
   HIERARCHY_FAMILIES:  Domain, RMS sequential, protected read
           <_CDD$TOP.DTR$LIB.DEMO.FAMILIES;1>
   FAMILIES_VIEW:  Domain, VIEW, shared read
           <_CDD$TOP.USER_CDD.PAN.WIZARD.FAMILIES_VIEW;1>
No loaded tables.

DTR> ! This statement is easier using the KIDS domain than using
DTR> ! the FAMILIES_VIEW or FAMILIES domain.
DTR> print first 5 kids sorted by descending age

  FAMILY       KID
    ID         NAME    AGE

         13 HAROLD     35
         10 ERIC       32
         13 CHARLIE    31
          2 ANNE       31
         11 MARTHA     30

DTR> !  But we  have  to  be careful sometimes to make sure we're
DTR> !  not caught by the circular-reference gotcha...

DTR> print age, kid-name, mom, dad of first 5 kids sorted by
       descending age
"PARENTS" is not a readied source, collection, or list.
DTR> finish parents, kids
DTR> ready parents shared   !This needs to be READIED first in
                            !this case, since we are using KIDS.
DTR> ready kids shared
DTR> print age, kid-name, mom, dad of first 5 kids sorted by
       descending age

       KID
AGE    NAME       MOM        DAD

35  HAROLD     SARAH      HAROLD
32  ERIC       RUTH       JEROME
31  CHARLIE    SARAH      HAROLD
31  ANNE       LOUISE     JIM
30  MARTHA     BETTY      TOM
```

You can also use the DTR Report Writer on the VIEW domains, which allows you to flatten the hierarchy using AT TOP or AT BOTTOM. You could print the COUNT of the list items, as we've done here, or use the RW to get the MAX and MIN or other statistic of the list.

```
DTR> report families-view cross kids with family-id le 3
RW>    at top of fid print mom, dad, all children
RW>    at bottom of fid print col 10, "Number of kids", space 3,
       count (-)
RW>  end-report

                                                  14-Jul-1986
                                                  Page 1

                                                KID
  MOTHER              FATHER                   NAME            AGE

ANN                  JIM                   RALPH                 3
                                           URSULA                7

   Number of kids        2
LOUISE               JIM                   ANNE                 31
                                           DAVID                24
                                           ELLEN                26
                                           JIM                  29
                                           ROBERT               19

   Number of kids       5
JULIE                JOHN                  ANN                  29
                                           JEAN                 26

   Number of kids       2
```

One other neat thing to do with VIEWs is to define new relationships between the different relations in the view using virtual fields. For example:

```
DTR> ! Now, we can do some sexy stuff with virtual fields...
DTR> declare oldest_kid computed by choice
CON>   age = max age of kids with fid = k.fid then "*"
CON>   else " "
CON> end_choice.
DTR> !
DTR> !  This allows you to identify this kid as the oldest.  This
DTR> !  would be very useful,  for  example,  if  you  were doing
DTR> !  statistical analysis in an attempt to find  a correlation
DTR> !  between behavior  and whether  or not this  child was the
DTR> !  oldest  in  his  family.    And it works no what the sort
DTR> !  order or RSE you use...
DTR> !
DTR> print kid-name, age, oldest-kid of first 10 k in kids

   KID         OLDEST
   NAME    AGE  KID

URSULA      7    *
RALPH
ANNE       31    *
JIM        29
ELLEN      26
DAVID      24
ROBERT     19
```

```
ANN         29    *
JEAN        26
CHRISTOPHR   0    *


DTR> ! Here's another sexy little virtual field...
DTR> declare oldest_kids_age computed by max age of kids with
CON>   fid=parents.fid.
DTR> declare oldest_kids_name computed by
CON>   kid_name from kids with fid = parents.fid and
CON>   age = oldest_kids_age.
DTR> for first 3 parents print mother, father, oldest_kids_age,
          oldest_kids_name

                            OLDEST   OLDEST
                             KIDS     KIDS
  MOTHER      FATHER         AGE      NAME

ANN         JIM               7   URSULA
LOUISE      JIM              31   ANNE
JULIE       JOHN             29   ANN
```

One could also make a VIEW to accomplish the same thing as the previous statement. If this were something we wanted to do frequently, or use a lot on an ad-hoc basis, then we would want to make this special VIEW. To make the view, we would use the SORTED BY and FIRST 1 clauses in the OCCURS FOR clause. Can you figure out how to do it? This is a VERY important advantage of a VIEW over a hierarchy - it allows you to look at the same data in several different ways. You can have different sort orders, different RSE's to select sub-groups of the records in the list, etc. These are things you simply can't do in hierarchical domains.

```
DTR> ! Try this in a hierarchy!
DTR> redefine domain families-view of parents, kids by
DFN> 01         family_unit occurs for parents.
DFN>   03       fid       from parents.
DFN>   03       father    from parents.
DFN>   03       mother    from parents.
DFN>   03       children occurs for kids with fid =
DFN>                      parents.fid sorted by
DFN>                ascending kid_name.
DFN>                05      kid_name from kids.
DFN>                05      age from kids.
DFN> ;
DTR>
DTR> finish
DTR> show domains
Domains:
        FAMILIES_VIEW;2 FAMILIES_VIEW;1 KIDS;1          PARENTS;1

DTR> ready families-view as families shared
DTR> print first 3 families

  FAMILY                                    KID
    ID        FATHER      MOTHER            NAME     AGE

          1 JIM         ANN             RALPH        3
                                        URSULA       7
          2 JIM         LOUISE          ANNE        31
                                        DAVID       24
                                        ELLEN       26
                                        JIM         29
                                        ROBERT      19
          3 JOHN        JULIE           ANN         29
                                        JEAN        26

DTR> ready parents shared
DTR> ready kids shared
DTR> print first 3 parents

  FAMILY
    ID        FATHER      MOTHER

          1 JIM         ANN
          2 JIM         LOUISE
          3 JOHN        JULIE

DTR> print mom, dad, number-of-kids of first 3 parents
"KIDS" is not a readied source, collection, or list.
DTR> ! There's our circular "gotcha" again...but no big deal!...
DTR> finish parents
DTR> ready parents
DTR> print mom, dad, number-of-kids of first 3 parents

                           NUMBER
                             OF
  MOTHER      FATHER        KIDS

ANN         JIM                2
LOUISE      JIM                5
JULIE       JOHN               2

DTR> print mom, dad, number-of-kids, oldest-kids-name,
CON> oldest-kids-age of first 3 parents
Internal error (expected block id 4, encountered id 0).
DTR> ! Another flavor of the circular gotcha...Here's the fix:
DTR> ! Just re-declare the variable...
DTR> declare oldest-kids-age computed by
CON>   max age of kids with fid = parents.fid.
DTR> print mom, dad, number-of-kids, oldest-kids-age of
       first 3 parents

                           NUMBER OLDEST
                             OF    KIDS
  MOTHER      FATHER        KIDS   AGE

ANN         JIM                2    7
LOUISE      JIM                5   31
JULIE       JOHN               2   29

DTR> print mom, dad, number-of-kids, oldest-kids-name of
       first 3 parents
Internal error (expected block id 4, encountered id 0).
DTR> !...and once again...
DTR> declare oldest-kids-name computed by
CON>   kid-name from kids with age = oldest-kids-age and
       fid = parents.fid.
DTR> print mom, dad, number-of-kids, oldest-kids-name,
       oldest-kids-age of first 3 parents

                           NUMBER   OLDEST    OLDEST
                             OF      KIDS      KIDS
  MOTHER      FATHER        KIDS     NAME      AGE

ANN         JIM                2 URSULA         7
LOUISE      JIM                5 ANNE          31
JULIE       JOHN               2 ANN           29
```

Now, I've done some fairly pathological cases here by creating a circular reference. In most cases, there would be no need to do this. Or, at the least, we could use variables instead of virtual fields in the record and eliminate much of the hassle. However, I wanted you to be able to see both the problem and how simple the solution can be.

Even if you don't use these cute virtual field tricks I've demonstrated, using VIEW domains instead of hierarchies, you'll eliminate much of the difficulty of dealing with hierarchies and you'll gain a tremendous amount of flexibility to boot!

Sincerely,
WW

Self-Documenting Datatrieve Files

William Leal - Kentucky Geological Survey

I have been commissioned to do a redesign and rewrite a system to track oil and gas well permitting. Design objectives included:

- Include all history of key activities with a given well.

Since these are variable in nature, fixed space cannot be set aside and secondary files need to be set up.

- Minimize storage cost by reducing unused fields. The trade-off is to increase the number of files.

These goals had a profound implication: they dictated that the system would have some 30 files - a relatively large number to manage. This in turn gave rise to one of my major problems in doing the design: how to keep track of what files I was defining.

Compounding the problem was the fact that each file had at least two elements: a record definition, and a domain definition; some of the files also had table definitions. There were also some dictionary tables that did not have associated files.

Good design of a complex system begins with good descriptions; programming comes later. I began using a word processor to record the key elements in the file design. I wanted to see an overview of what I was working with, and I needed to record the information elements in detail as well. So I had something like this:

```
PRIMARY$WELL
     Contains the primary information for each permit.... {and
     other comments}
     --------------------
     | Permit # | County |
     --------------------

FARM-OWNER$WELL
     Historical File.  Shows the farm owner(s).... {and other
     comments}
     --------------------------------
     | Permit Number | Transfer Date |
     --------------------------------
```

In this overview, a bold entry signified a primary key; an underlined entry signified a secondary key. In another portion of the document I had corresponding detail:

PRIMARY$WELL

Permit #	9(7)
County	X(12)
.	
.	{and other detail}
.	

FARM-OWNER$WELL

Primary Key	
Permit Number	9(7)
Transfer Date	DATE
.	
.	{and other detail}
.	

Soon, keeping up with all this became a major clerical task. A small change in organization meant several hours at the word processor to make all the changes. Clearly some automation was called for.

Help arrived in the form of the Datatrieve record structure and the EXTRACT ALL command. First, I established some commenting conventions in the record structure and began to define the structures. Taking the above example for farm owner, we have

```
REDEFINE RECORD FARM_OWNER_REC
!
!. Historical File.  Shows the farm owner(s) associated
!. with each well.
!
!k Permit Number
!k Transfer Date
!
!=====================================================
!                                  !
!    Record Def                    !    Comments
!                                  !
!=====================================================
!                                  !
01 FARM_OWNER.                     !
   05 PRIMARY_KEY.                 !
      10 PERMIT_NO PIC 9(7).!
      10 TRANSFER-9DATE.           ! This special form of
         15 YEAR INTEGER.          ! keeping dates guarantees
         15 MONTH INTEGER.         ! that dates will be in
         15 DAY INTEGER.           ! order tho part of group
         .
         .    {rest of definition}
         .
```

Here I established commenting conventions to help me out. A line which begins with !. is taken to be a descriptive line which is used to produce an overview. A line which begins with !k indicates a primary key data item; this is also used in the overview. (!s is used for a secondary key, !b for an item which is both primary and secondary.)

Now, if I do an EXTRACT ALL, I can write a program which will input the record definitions and will automatically give me my reports, either in summary (as the first form) or in detail.

Given that I could do this, it would be very nice to also use the information available and produce a matrix showing the file name and whether I had made the domain, record structure, and table structures. Such a report would look like this:

```
          NAME                   DOMAIN  RECORD  TABLE
          ----------------------------------------
          PRIMARY                  *       *       *
          WELL_OWNER               *       *
```

I accomplished this based on the following:

- In this application, there is a one-to-one relationship between record structures and domains. Every record structure has a domain and vice versa.

- Several of the files also have associated tables.

Therefore, correct naming conventions combined with an associated report program would produce the report. Here are the conventions I established:

- Each file has a name. This same name is used in the domain definition, the record definition, and the table definition.

- Domain definitions have the form <name>$<type>. (See below for a description of <type>.)

- Record definitions have the form <name>_REC.

- Table definitions have the form <name>_TABLE.

For example, we have the following for PRIMARY:

```
         Domain Definition   PRIMARY$WELL
         Record Definition   PRIMARY_REC
         Table Definition    PRIMARY_TABLE
```

As mentioned above, the files are divided into groups, referred to as <type> above. Some files are concerned with well information directly; some are connected with bonds; others with violations, and so forth. By making <type> a part of the domain definition, each file name can be assigned a type, and the reports can be correspondingly ordered.

Some sample reports are shown below, followed by the actual Datatrieve code used to produce them.

```
         Example Matrix of OGD Well Record System Elements
                     Sorted by Domain Type

                                                           DOMAIN
      NAME                       DOMAIN  RECORD TABLE       TYPE
-------------------------------------------------------------

BLANKET_CASH                       *       *              $BOND
GAS_STORAGE_OWNER                  *       *      *       $INFO
OPERATOR                           *       *      *       $OP
ABANDON                            *       *              $VIOL
FARM_OWNER                         *       *              $WELL
-------------------------------------------------------------


    {$BOND}  BLANKET_CASH
        Shows the blanket cash bonds
    ---------------
    | Bond Number |
    ---------------

    {$INFO}  GAS_STORAGE_OWNER
        Reference File.  Contains the names of all gas storage
        owners.
    --------------------------
    | Gas Storage Owner Name |
    --------------------------

    {$OP}  OPERATOR
        Reference File.  Contains the valid operators recognized
        by OGD
    -----------------
    | Operator Name |
    -----------------

    {$VIOL}  ABANDON
        Shows the improper abandonment violations.
    --------------------
    | Violation Number |
    --------------------

    {$WELL}  FARM_OWNER
        Historical file.  Given a permit #, the current farm
        owner is shown.
    -----------------------------
    | Permit Ref | Transfer Date |
    -----------------------------

    !|...........................................................|
    !|...........................................................|
    !|                                                           |
    !|      Procedure to produce matrix summary                  |
!|                                                          |
!|..........................................................|
!|..........................................................!
!
REDEFINE PROCEDURE D_M
!
!       This procedure looks through the extracted file and
!      finds all domains, records, and tables in the file.
!       It builds up a summary data set consisting of  this
!       information.  Then it brings this summary set back
!       in, in sorted  order,  and  builds up a matrix data
!       set which has an entry filled in for the structures
!       (domains, records, tables) that are associated with
!       each name.  The resultant matrix  data  set is then
!       printed.

PRINT "{ Initializing }"
!
! Document - Matrix
!
SET COLUMNS_PAGE = 80
FN$WIDTH(80)
!
DECLARE GENTRY_TYPE PIC X(10).
DECLARE GENTRY_NAME PIC X(50).
DECLARE GDTYPE PIC X(10).
DECLARE GENTRY_LEN INTEGER.
DECLARE LAST_NAME PIC X(30).
DECLARE LAST_DOMAIN PIC X.
DECLARE LAST_RECORD PIC X.
DECLARE LAST_TABLE  PIC X.
DECLARE LAST_DTYPE  PIC X(10).
!
! Extract name & type from DOC.TMP
!
PRINT "{ Inputting from Extract }"
FINISH ALL
READY EXTRACT_DOM
READY SUMMARY WRITE
READY MATRIX WRITE
!
ERASE ALL OF SUMMARY;
FOR EXTRACT_DOM BEGIN
  :EXTRACT_ENTRY          ! Evaluate an entry, return thru
                          !      GENTRY  variables
  IF GENTRY_TYPE EQ "DOMAIN","RECORD","TABLE" THEN BEGIN
    STORE SUMMARY USING BEGIN
      SUMMARY.ENTRY_TYPE = GENTRY_TYPE
      SUMMARY.ENTRY_NAME = GENTRY_NAME
      SUMMARY.DTYPE = GDTYPE
      SUMMARY.ENTRY_LEN = GENTRY_LEN
      END
    END
```

```
  END
!
! Put results into matrix form
!
PRINT "{ Converting to Matrix Form }"
ERASE ALL OF MATRIX;
LAST_NAME = " "
FOR SUMMARY SORTED BY SUMMARY.ENTRY_NAME BEGIN
  IF SUMMARY.ENTRY_NAME NE LAST_NAME THEN BEGIN
    IF LAST_NAME NE " " THEN STORE MATRIX USING BEGIN
      MATRIX.NAME   = LAST_NAME
      MATRIX.DOMAIN = LAST_DOMAIN
      MATRIX.RECORD = LAST_RECORD
      MATRIX.TABLE  = LAST_TABLE
      MATRIX.DOMAIN_TYPE = LAST_DTYPE
      END
    LAST_NAME = SUMMARY.ENTRY_NAME
    LAST_DOMAIN = " "
    LAST_RECORD = " "
    LAST_TABLE  = " "
    LAST_DTYPE  = " "
    END
  IF SUMMARY.ENTRY_TYPE EQ "DOMAIN" THEN LAST_DOMAIN = "*"
  IF SUMMARY.ENTRY_TYPE EQ "RECORD" THEN LAST_RECORD = "*"
  IF SUMMARY.ENTRY_TYPE EQ "TABLE"  THEN LAST_TABLE  = "*"
  IF SUMMARY.DTYPE NE " " THEN LAST_DTYPE = SUMMARY.DTYPE
  END
STORE MATRIX USING BEGIN
  MATRIX.NAME   = LAST_NAME
  MATRIX.DOMAIN = LAST_DOMAIN
  MATRIX.RECORD = LAST_RECORD
  MATRIX.TABLE  = LAST_TABLE
  MATRIX.DOMAIN_TYPE = LAST_DTYPE
  END
!
!     Report  on the results.  References to  logical  name
!     OUTOPT  permit  the  procedure  SET_OUTOPT to set the
!     logical  name,  then  have the following code test it
!     for  alternative  output devices.  FYI - TXA12:  is a
!     Xerox 4045 Laser Printer

:SET_OUTOPT     ! Set logical name OUTOPT
IF OPTION NE 8 THEN ON OUTOPT BEGIN
  IF FN$TRANS_LOG("OUTOPT") = "TXA12:" THEN
          PRINT ESC|"+X" ! Reset ptr

  REPORT MATRIX SORTED BY DOMAIN_TYPE,NAME
    SET REPORT_NAME =
          "Matrix of OGD Well Record System Elements"/
              "Sorted by Domain Type"
    SET NO NUMBER
    AT TOP OF DOMAIN_TYPE PRINT
      "----------------------------------------" |
      "----------------------------------------", SKIP
    PRINT NAME, DOMAIN, RECORD, TABLE, DOMAIN_TYPE, SKIP;
    AT BOTTOM OF REPORT PRINT
      "----------------------------------------" |
      "----------------------------------------", SKIP
    END_REPORT
  END ! option ne 8
END-PROCEDURE

!.........................................................
!
!       Procedure to produce overview
!
!.........................................................
!
REDEFINE PROCEDURE D_O
PRINT "{ Initializing }"
!
! Document - Overview
!       This will produce one of three reports:
!               1.  Key Item display
!               2.  Comment listing
!               3.  Key Item display with Comments
!       Incoming global variables are
!               OVERVIEW_KEY
!               OVERVIEW_COMMENT
!      If OVERVIEW-KEY is non-blank, then produce Key Item
!      display  If  OVERVIEW_COMMENT  is  non-blank,  then
!      produce  Comment  listing  If  both  variables  are
!      non-blank,  then  produce  Key  Item  display  with
!      Comments

DECLARE CR PIC X DEFAULT "<CR>".
DECLARE ESC PIC X DEFAULT "<ESC>".
!
DECLARE GENTRY_TYPE PIC X(10).
DECLARE GENTRY_NAME PIC X(50).
DECLARE GDTYPE PIC X(10).
DECLARE GENTRY_LEN INTEGER.
!
DECLARE CUR_TYPE PIC X(10).
DECLARE CUR_NAME PIC X(50).
DECLARE CUR_DTYPE PIC X(10).
!
DECLARE LAST_NAME PIC X(50).
DECLARE LAST_DTYPE PIC X(10).
DECLARE KEY_LINE PIC X(255).
DECLARE KEY_LINE_DASH PIC X(255).
DECLARE KEY_LABEL PIC X(50).
DECLARE COM_LINE PIC X(132).
DECLARE NLINE INTEGER DEFAULT 0.
DECLARE MAXLINE INTEGER DEFAULT 50.
!
```

```
! Set widths
!
FN$WIDTH(80)
SET COLUMNS_PAGE = 80
!
FINISH ALL
READY EXTRACT_DOM
READY KEY_ITEMS WRITE
READY COMMENT WRITE
IF OVERVIEW_KEY NE " " THEN ERASE ALL OF KEY_ITEMS;
IF OVERVIEW_COMMENT NE " " THEN ERASE ALL OF COMMENT;
!=============================================================
! Input from extract
!=============================================================
PRINT "{ Inputting from Extract }"
FOR EXTRACT_DOM BEGIN
  :EXTRACT_ENTRY ! Return info re name & type in
                 ! GENTRY_ variables
  IF GENTRY_TYPE NE " " THEN BEGIN
    CUR_TYPE = GENTRY_TYPE
    CUR_NAME = GENTRY_NAME
    IF GDTYPE NE " " THEN CUR_DTYPE = GDTYPE
    END ! if gentry-type
!
!       Store into COMMENT if OVERVIEW_COMMENT switch set
!
  IF CUR_TYPE = "RECORD"  AND COMMENT.FLAG = "!."
                        AND OVERVIEW_COMMENT NE " "
                        THEN STORE COMMENT USING BEGIN
    COMMENT.COMMENT.ENTRY_NAME = GENTRY_NAME
    COMMENT.COMMENT.SEQ = RUNNING COUNT
    COMMENT.COMMENT.DTYPE = CUR_DTYPE
    COM_LINE = EXTRACT.COMMENT.BALANCE
    IF FN$STR_EXTRACT(COM_LINE,1,1) = " " THEN
      COM_LINE = FN$STR_EXTRACT(COM_LINE,2,999)
    COMMENT.COMMENT.COMMENT = COM_LINE
    END ! if cur-type ... store comment
!
!       Store into KEY-ITEMS if OVERVIEW_KEY switch set
!
  IF CUR_TYPE = "RECORD" AND COMMENT.FLAG EQ "!k", "!s",
          "!b" AND OVERVIEW_KEY NE " "
          THEN STORE KEY_ITEMS USING BEGIN
    KEY_ITEMS.NAME = CUR_NAME
    KEY_ITEMS.SEQ  = RUNNING COUNT
    KEY_ITEMS.DTYPE = CUR_DTYPE
    KEY_LABEL = EXTRACT.COMMENT.BALANCE
    WHILE FN$STR_EXTRACT(KEY_LABEL,1,1) = " " AND
          KEY_LABEL NE " "
      KEY_LABEL = FN$STR_EXTRACT(KEY_LABEL,2,999) ! Delete
                                                  ! leading spaces
    KEY_ITEMS.LABEL = KEY_LABEL
    CHOICE      ! Set switches as to Primary or Secondary
```

```
                 ! or Both
      COMMENT.FLAG EQ "!k" THEN KEY_ITEMS.PRIMARY = "X"
      COMMENT.FLAG EQ "!s" THEN KEY_ITEMS.SECONDARY = "X"
      COMMENT.FLAG EQ "!b" THEN BEGIN
        KEY_ITEMS.PRIMARY = "X"
        KEY_ITEMS.SECONDARY = "X"
        END ! comment.flag eq "!b"
      END_CHOICE
    END ! if comment.flag ... store
  END ! for extract-dom
!===========================================================
! Choose output target, create logical name OUTOPT
!===========================================================
:SET_OUTOPT      ! Set logical name OUTOPT so output decision
                 ! can be
made below
SET COLUMNS_PAGE = 255
FN$WIDTH(132)
!===========================================================
! Output the comments if OVERVIEW_KEY is set but
!       OVERVIEW_COMMENT is not set
!===========================================================
IF OVERVIEW_COMMENT NE " " AND OVERVIEW_KEY = " " AND
          OPTION NE 8 THEN ON OUTOPT BEGIN
  IF FN$TRANS_LOG("OUTOPT") = "TXA13:" THEN      ! C. Itoh
    PRINT ESC|"Q"|ESC|"L005"    ! Set 17 cpi, left margin
  IF FN$TRANS_LOG("OUTOPT") = "TXA12:" THEN
                                           ! Xerox 4045 Laser
    PRINT ESC|"+X"|ESC|"+1Titan10iso-P",
          SKIP,ESC|"+2XCP14iso-L",
      SKIP,ESC|"2"            ! Reset, then 14 cpi landscape
  FOR COMMENT BEGIN
    IF COMMENT.COMMENT.ENTRY_NAME NE LAST_NAME THEN BEGIN
      LAST_NAME = COMMENT.COMMENT.ENTRY_NAME
      PRINT SKIP 3, COMMENT.COMMENT.ENTRY_NAME (-), SKIP
      END
    PRINT COMMENT.COMMENT.COMMENT (-) USING X(132)
    END ! for comment
  IF FN$TRANS_LOG("OUTOPT") = "TXA13:" THEN      ! C. Itoh
    PRINT ESC|"N"|ESC|"L000"    ! Reset ptr (10 cpi,
                                ! 0 left margin)
  IF FN$TRANS_LOG("OUTOPT") = "TXA12:" THEN      ! Laser
    PRINT ESC|"+X"              ! Reset
  END ! Comment listing
!===========================================================
! Output the key info in box form if OVERVIEW_KEY is set.
!      Add comments if OVERVIEW_KEY switch is also set.
!===========================================================
IF OVERVIEW_KEY NE " " AND OPTION NE 8 THEN ON OUTOPT BEGIN
  IF FN$TRANS_LOG("OUTOPT") = "TXA13:" THEN
    PRINT ESC|"Q"|ESC|"L005"    ! 17 cpi, left margin
  IF FN$TRANS_LOG("OUTOPT") = "TXA12:" THEN      ! Laser
    PRINT ESC|"+X"|ESC|"+1Titan10iso-P",
```

```
          SKIP,ESC|"+2XCP14iso-L",
          SKIP,ESC|"2"        ! Reset, then 14 cpi landscape
  FOR KEY_ITEMS SORTED BY KEY_ITEMS.DTYPE, KEY_ITEMS.NAME,
          KEY_ITEMS.SEQ BEGIN
!
!       Handle new group
!
    IF KEY_ITEMS.NAME NE LAST_NAME THEN BEGIN
      IF KEY_LINE NE " " THEN BEGIN
        :KEY_ITEMS_PRINT        ! Print the box +
                                ! optionally comments
        IF KEY_ITEMS.DTYPE NE LAST_DTYPE THEN
                                ! Split types w/line
           PRINT SKIP, "------------------------------" |
                       "------------------------------" |
                       "------------------------------" |
                       "------------------------------",
                       SKIP
           NLINE = NLINE + 1
        END ! if key-line ne " "
      LAST_NAME = KEY_ITEMS.NAME
      LAST_DTYPE = KEY_ITEMS.DTYPE
      KEY_LINE = "|"; KEY_LINE_DASH = "-"
      END ! if key-items ne last-name
!
!       Handle bold/underline setup
!
    IF FN$TRANS_LOG("OUTOPT") = "TXA13:" THEN BEGIN
                                           ! C. Itoh
      IF KEY_ITEMS.PRIMARY NE " " THEN
        KEY_LINE = KEY_LINE ||| ESC | "!"
      IF KEY_ITEMS.SECONDARY NE " " THEN
        KEY_LINE = KEY_LINE ||| ESC | "X"
      END ! TXA13 init
    IF FN$TRANS_LOG("OUTOPT") = "TXA12:" THEN BEGIN
                                           ! Xerox 4045
      IF KEY_ITEMS.PRIMARY NE " " THEN
        KEY_LINE = KEY_LINE ||| ESC | "b"
      IF KEY_ITEMS.SECONDARY NE " " THEN
        KEY_LINE = KEY_LINE ||| ESC | "u"
      END ! TXA12 init
    IF FN$TRANS_LOG("OUTOPT") = "TT:" THEN BEGIN
                                           ! Term (VT100)
      IF KEY_ITEMS.PRIMARY NE " " AND
          KEY_ITEMS.SECONDARY EQ " " THEN
                KEY_LINE = KEY_LINE ||| ESC | "[1m"
      IF KEY_ITEMS.PRIMARY EQ " " AND
          KEY_ITEMS.SECONDARY NE " " THEN
                KEY_LINE = KEY_LINE ||| ESC | "[4m"
      IF KEY_ITEMS.PRIMARY NE " " AND
          KEY_ITEMS.SECONDARY NE " " THEN
                KEY_LINE = KEY_LINE ||| ESC | "[1;4m"
      END ! TT init
```

```
!
!       Append the label
!
    KEY_LINE = KEY_LINE || KEY_ITEMS.LABEL
!
!       Include the bars necessary
!
    REPEAT FN$STR_LENGTH(KEY_ITEMS.LABEL||"") + 3
      KEY_LINE_DASH = KEY_LINE_DASH || "-"
!
!       Reset output to normal
!
    IF FN$TRANS_LOG("OUTOPT") = "TXA13:" THEN
      KEY_LINE = KEY_LINE || ESC | '"' | ESC | "Y"
    IF FN$TRANS_LOG("OUTOPT") = "TXA12:" THEN
      KEY_LINE = KEY_LINE || ESC | "p" | ESC | "w"
    IF FN$TRANS_LOG("OUTOPT") = "TT:" THEN
      KEY_LINE = KEY_LINE || ESC | "[0m"
!
!       Append the closing |
!
    KEY_LINE = KEY_LINE ||| "|"
    END ! for key-items
!
!       Get the last one
!
  :KEY_ITEMS_PRINT
  IF FN$TRANS_LOG("OUTOPT") = "TXA13:" THEN
    PRINT ESC|"N"|ESC|"L000"      ! Reset ptr (10 cpi,
                                  ! 0 left margin)
  IF FN$TRANS_LOG("OUTOPT") = "TXA12:" THEN
    PRINT ESC|"+X"                ! Reset printer
  END ! Key Item output
END-PROCEDURE

!...........................................................
!
!       Procedure to list everything
!
!...........................................................
!
REDEFINE PROCEDURE LST
PRINT "{ Initializing }"
!
!  Document - Listing.  This gives a complete listing of the
!  extract,  sorted  by  type.  Record definitions  are  all
!  given on separate pages;  the rest are  printed one after
!  the other to conserve paper

SET COLUMNS_PAGE = 80
SET COLUMNS_PAGE = 132
FN$WIDTH(80)
!
```

```
DECLARE FF PIC X DEFAULT "<FF>".
DECLARE ESC PIC X DEFAULT "<ESC>".
!
DECLARE GENTRY_TYPE PIC X(10).
DECLARE GENTRY_NAME PIC X(50).
DECLARE GDTYPE PIC X(10).
DECLARE GENTRY_LEN INTEGER.
!
DECLARE LAST_TYPE PIC X(10).
DECLARE LAST_NAME PIC X(20).
DECLARE LAST_DOMAIN PIC X.
DECLARE LAST_RECORD PIC X.
DECLARE LAST_TABLE  PIC X.
DECLARE LAST_DTYPE  PIC X(10).
!
!============================================================
! Extract name & type from DOC.TMP
!============================================================
PRINT "{ Inputting from Extract }"
FINISH ALL
READY EXTRACT_DOM
DEFINE FILE FOR LISTING, KEY=SEQ
DCL_SPAWN("purge [bill.ogd.files]doc_listing.dat")
                   ! DCL_SPAWN is a function that executes a
                   ! DCL command out of DATATRIEVE
READY LISTING WRITE
!
!       Extract all the detail of the definitions, adding in
!       type information for sorting
!
FOR EXTRACT_DOM BEGIN
  :EXTRACT_ENTRY      ! Pull out the details about each entry
  IF GENTRY_TYPE NE " " THEN BEGIN
    LAST_TYPE = GENTRY_TYPE
    LAST_NAME = GENTRY_NAME
    IF GDTYPE NE " " THEN LAST_DTYPE = GDTYPE
    END
  IF LAST_TYPE NE " " AND EXTRACT.INPUT_LINE NE " " AND
      EXTRACT.DELETE_COMMAND NE "DELETE" THEN
                       STORE LISTING USING BEGIN
    LISTING.SEQ = RUNNING COUNT
    LISTING.ENTRY_TYPE = LAST_TYPE
    LISTING.ENTRY_NAME = LAST_NAME
    LISTING.DTYPE = LAST_DTYPE
    LISTING.INPUT_LINE = EXTRACT.INPUT_LINE
    END
  END
PRINT "{ Printing Report }"
!============================================================
! Print listing
!============================================================
SET COLUMNS_PAGE=132
FN$WIDTH(132)
```

```
LAST_TYPE = " "; LAST_NAME = " "
ON TXA12: BEGIN         ! TXA12 is Xerox 4045
        FOR LISTING SORTED BY LISTING.ENTRY_TYPE,
                LISTING.DTYPE,LISTING.ENTRY_NAME,
                LISTING.SEQ BEGIN
                IF LISTING.ENTRY_TYPE NE LAST_TYPE THEN
                  BEGIN
                        PRINT SKIP 2, FF (-), SKIP,
                                "====================" |||
                                LISTING.ENTRY_TYPE |||
                                "===================="
                        LAST_TYPE = LISTING.ENTRY_TYPE
                        LAST_NAME = " "
                  END
                IF LISTING.ENTRY_NAME NE LAST_NAME THEN
                  BEGIN
                        PRINT SKIP 2,
                                IF LAST_NAME NE " " AND
                                        LAST_TYPE = "RECORD"
                                        THEN FF ELSE
                                        "-----------" |
                                        "-----------" |
                                        "-----------" |
                                        "-----------" |
                                        "-----------" |
                                        "-----------",
                                SKIP, COLUMN 5, "{" |||
                                LISTING.DTYPE || "}" |||
                                LISTING.ENTRY_NAME |||
                                "(" | LAST_TYPE || ")", SKIP
                        LAST_NAME = LISTING.ENTRY_NAME
                  END
                PRINT LISTING.INPUT_LINE (-)
                END ! for listing
        PRINT ESC|"+X"  ! Reset printer
        END ! on
END-PROCEDURE

!...........................................................
!
!       Subroutine procedures
!
!...........................................................
!
REDEFINE PROCEDURE EXTRACT_ENTRY
!
! This is called by other procedures.
!
! Input is a record from EXTRACT-DOM.
! Output are global variables:
!    GENTRY-TYPE     has the values of DOMAIN, RECORD, TABLE
!    GENTRY-NAME     has the item's name
!    GDTYPE          is the postfix domain type descriptor
```

```
!                       if any ($INFO, etc)
!    GENTRY-LEN      is the length of GENTRY-NAME
!
! If input is not DOMAIN, RECORD, etc., GENTRY-TYPE is blank
!       on return.
!
DECLARE I INTEGER.
!
! Initialize GENTRY-TYPE
!
GENTRY_TYPE = " "
IF REDEFINE_COMMAND = "REDEFINE" THEN BEGIN
!
! Process RECORD
!
  IF RECORD_TYPE="RECORD" THEN BEGIN
    GENTRY_TYPE = RECORD_TYPE
    GENTRY_NAME = RECORD_10.ENTRY_NAME
    :EXTRACT_NAME
!       Remove the _REC if present
    IF FN$STR_EXTRACT(GENTRY_NAME,GENTRY_LEN - 3,4) =
      "_REC" THEN BEGIN
      GENTRY_LEN = GENTRY_LEN - 4
      GENTRY_NAME = FN$STR_EXTRACT(GENTRY_NAME,1,GENTRY_LEN)
      END
    END ! if record
!
! Process DOMAIN
!
  IF DOMAIN_TYPE="DOMAIN" THEN BEGIN
    GENTRY_TYPE = DOMAIN_TYPE
    GENTRY_NAME = DOMAIN_10.ENTRY_NAME
    :EXTRACT_NAME
    END ! if domain
!
! Process TABLE
!
  IF TABLE_TYPE="TABLE" THEN BEGIN
    GENTRY_TYPE = TABLE_TYPE
    GENTRY_NAME = TABLE_10.ENTRY_NAME
    :EXTRACT_NAME
    IF FN$STR_EXTRACT(GENTRY_NAME,GENTRY_LEN - 5,6)="_TABLE"
            THEN BEGIN
      GENTRY_LEN = GENTRY_LEN - 6
      GENTRY_NAME = FN$STR_EXTRACT(GENTRY_NAME,1,GENTRY_LEN)
      END
    END ! if table
!
! Process PROCEDURE
!
  IF PROCEDURE_TYPE = "PROCEDURE" THEN BEGIN
    GENTRY_TYPE = PROCEDURE_TYPE
    GENTRY_NAME = PROCEDURE_10.ENTRY_NAME
    :EXTRACT_NAME
    END ! if procedure
!
! If domain, look for GDTYPE
!
  GDTYPE = " "
  IF GENTRY_TYPE EQ "DOMAIN" THEN BEGIN
    I = GENTRY_LEN
!       Search for last $
    WHILE FN$STR_EXTRACT(GENTRY_NAME,I,1) NE "$" AND I > 0
      I = I - 1
    IF I NE 0 THEN BEGIN
      GDTYPE = FN$STR_EXTRACT(GENTRY_NAME,I,50)
      GENTRY_LEN = I - 1
      GENTRY_NAME = FN$STR_EXTRACT(GENTRY_NAME,1,GENTRY_LEN)
      END
    END ! if domain
  END ! if redefine
END-PROCEDURE

!...........................................................

REDEFINE PROCEDURE EXTRACT_NAME
!
!   Given  a  definition  in GENTRY_NAME, locate first blank
!   and strip  off  following characters.  Set GENTRY_LEN to
!   length of result.

DECLARE I INTEGER.
I=1
WHILE FN$STR_EXTRACT(GENTRY_NAME,I,1) NE " "
  I=I+1
GENTRY_LEN=I - 1
GENTRY_NAME=FN$STR_EXTRACT(GENTRY_NAME,1,GENTRY_LEN)
END-PROCEDURE

!...........................................................

REDEFINE PROCEDURE KEY_ITEMS_PRINT
!
! Print key items for overview
!
IF NLINE > MAXLINE THEN BEGIN
  PRINT NEW_PAGE
  NLINE = 0
  END
PRINT "{" | LAST_DTYPE || "}  "| LAST_NAME (-)
NLINE = NLINE + 1
!
!       If OVERVIEW_COMMENT switch set, print comments
!
IF OVERVIEW_COMMENT NE " " THEN FOR COMMENT WITH
  COMMENT.COMMENT.ENTRY_NAME = LAST_NAME BEGIN
```

```
    PRINT COL 5, COMMENT.COMMENT.COMMENT (-) USING X(125)
    NLINE = NLINE + 1
    END
!
!       Print the box w/primary & secondary key stuff
!
PRINT COL 5, KEY_LINE_DASH (-), SKIP,
  COL 5, KEY_LINE (-), SKIP,
  COL 5, KEY_LINE_DASH (-), SKIP; NLINE = NLINE + 4
END-PROCEDURE

!...........................................................
!
!       Domains
!
!...........................................................
!
REDEFINE DOMAIN COMMENT USING
     COMMENT_REC ON
     [BILL.OGD.FILES]DOC_COMMENT.DAT;

REDEFINE DOMAIN EXTRACT_DOM USING
     EXTRACT_REC ON
     [BILL.OGD.FILES]DOC.TMP;

REDEFINE DOMAIN KEY_ITEMS USING
     KEY_ITEMS_REC ON
     [BILL.OGD.FILES]DOC_KEY_ITEMS.DAT;

REDEFINE DOMAIN LISTING
  USING LISTING_REC
  ON [BILL.OGD.FILES]DOC_LISTING.DAT;

REDEFINE DOMAIN MATRIX USING
     MATRIX_REC ON
     [BILL.OGD.FILES]DOC_MATRIX.DAT;

REDEFINE DOMAIN SUMMARY USING
     SUMMARY_REC ON
     [BILL.OGD.FILES]DOC_SUMMARY.DAT;

!...........................................................
!
!       Record Definitions
!
!...........................................................
!
REDEFINE RECORD COMMENT_REC
01 COMMENT.
   05 PRIMARY_KEY.
      10 ENTRY_NAME PIC X(50).
      10 SEQ PIC 9999.
   05 DTYPE PIC X(10).
   05 COMMENT PIC X(132).
;

REDEFINE RECORD EXTRACT_REC
!
!       This record definition sets up a line of 132
!       characters and then provides multiple definition
!       overlays via REDEFINES to allow easy pre-parsing of
!       the input. The overlays are based on the fact that
!       DATATRIEVE follows a fixed format for putting out
!       the various elements. For example, doing EXTRACT of
!       YACHTS results in
!
!               DELETE YACHTS;
!               REDEFINE DOMAIN YACHTS ....
!
!      Thus, in the definitions below, we can test if
!      DELETE_COMMAND = "DELETE" to see if we have a line
!      coming through with DELETE in it. Similarly, we can
!      test for REDEFINE_COMMAND = "REDEFINE" to test for a
!      REDEFINE line. The REDEFINE line will be followed by
!      one of RECORD, DOMAIN, PROCEDURE, TABLE. So we can
!      further test for RECORD_TYPE = "RECORD", for example.
!      If true, then the entry name follows in ENTRY_NAME
!      and we can make the assignment GENTRY_NAME =
!      RECORD10_ENTRY_NAME.

01 EXTRACT.
   05 INPUT_LINE PIC X(132).
!
!       These redefinitions are for name & type extract
!
   05 DELETE_COMMAND REDEFINES INPUT_LINE PIC X(6).
   05 REDEFINE_LINE REDEFINES INPUT_LINE.
      10 REDEFINE_COMMAND PIC X(8).
      10 FILLER PIC X.
      10 TYPE_NAME PIC X(120).
      10 RECORD_10 REDEFINES TYPE_NAME.
         15 RECORD_TYPE PIC X(6).
         15 FILLER PIC X.
         15 ENTRY_NAME PIC X(50).
      10 DOMAIN_10 REDEFINES TYPE_NAME.
         15 DOMAIN_TYPE PIC X(6).
         15 FILLER PIC X.
         15 ENTRY_NAME PIC X(50).
      10 TABLE_10 REDEFINES TYPE_NAME.
         15 TABLE_TYPE PIC X(5).
         15 FILLER PIC X.
         15 ENTRY_NAME PIC X(50).
      10 PROCEDURE_10 REDEFINES TYPE_NAME.
         15 PROCEDURE_TYPE PIC X(9).
         15 FILLER PIC X.
         15 ENTRY_NAME PIC X(50).
```

```
!
!       These redefinitions are for comment lines
!       They are preceded by !., !k, !s, !b . Thus,
!       FLAG can be tested with BALANCE giving the rest.
!
   05 COMMENT REDEFINES INPUT_LINE.
      10 FLAG PIC XX.
      10 BALANCE PIC X(130).
;

REDEFINE RECORD KEY_ITEMS_REC
01 KEY_ITEMS.
   05 PRIMARY_KEY.
      10 NAME PIC X(30).
      10 SEQ PIC 9999.
   05 DTYPE PIC X(10).
   05 LABEL PIC X(30).
   05 PRIMARY PIC X DEFAULT " ".
   05 SECONDARY PIC X DEFAULT " ".
;

REDEFINE RECORD LISTING_REC
  01 LISTING.
     05 SEQ INTEGER.
     05 ENTRY_TYPE PIC X(10).
     05 ENTRY_NAME PIC X(50).
     05 DTYPE PIC X(10).
     05 INPUT_LINE PIC X(132).
;

REDEFINE RECORD MATRIX_REC
01 MATRIX.
   05 NAME PIC X(30).
   05 DOMAIN PIC X.
   05 RECORD PIC X.
   05 TABLE PIC X.
   05 DOMAIN_TYPE PIC X(10).
;

REDEFINE RECORD SUMMARY_REC
  01 SUMMARY.
     05 TYPE_NAME.
       10 ENTRY_TYPE PIC X(10).
       10 ENTRY_NAME PIC X(50).
     05 DTYPE PIC X(10).
     05 ENTRY_LEN INTEGER.
  ;
```

Wombat Magic: Part II - Dallas Symposium - May 1, 1986
Session Editor: Donald E. Stern, Jr. - Warner Lambert Company

This is the second in a series reporting on the Wombat Magic Session which was held at the Dallas Symposium. The speakers comments are quoted where appropriate and the magic is not necessarily presented in the original order.

For those of you new to DECUS and/or the newsletter, Wombat Magic is a session, sponsored by the Datatrieve/4GL SIG at each symposium, at which users of Datatrieve share information about unique and/or unexpected ways to use Datatrieve.

****

Nested Menu Link

Gary W. Burton - E.I. Dupont, Parkersburg, WV

****

At past Wombat Magic sessions, the topic of how to create a Datatrieve menu procedure has come up repeatedly. This comes up because Datatrieve will object if a command (as opposed to a statement) is placed inside a compound statement. For example, the following compound statement is NOT valid Datatrieve syntax.

```
IF SELECTION = 1 THEN READY YACHTS ELSE READY PERSONNEL
```

Gary's presentation of a Datatrieve menu procedure is one way around this limitation. The procedure takes advantage of Datatrieve's ability to execute a DCL command procedure which contains valid Datatrieve commands and statements. In this sample implementation, a directory called [USER.HELPER] is created with a version limit of one. (This means that files of the same filename.extension as a newly created file will automatically be purged.)

```
$ CREATE/DIR/VERSION_LIMIT=1 [USER.HELPER]
```

The example Gary presented is a procedure called MENU. This procedure had three variables declared, SELECTION, ACTION, and LOOP. After declaring these variables, a menu is printed after which the user is prompted for a value for SELECTION. In this sample procedure the menu lists seven possibilities (1-7) with 6 and 7 being escapes to DATATRIEVE and DCL respectively. Now, based upon the value of SELECTION, a value is assigned to the value action. For example, if SELECTION=4 then the variable action is set equal to the string "EXECUTE PROC4". Now if the selection is 1 through 5 the variable LOOP is set equal to the string "EXECUTE MENU" else it is set equal to the string "! NO ACTION". At this point, and here's where the magic begins, the Datatrieve command OPEN is used to create a file called HELPER.COM. The variables ACTION and LOOP are printed to this file. After closing the file and releasing the variables SELECTION, ACTION, and LOOP, the contents of HELPER.COM are executed. Since LOOP="EXECUTE MENU" for selections 1 through 5, the MENU procedure will be restarted after executing the contents of HELPER.COM.

```
REDEFINE PROCEDURE MENU
DECLARE SELECTION PIC X.
DECLARE ACTION PIC X(13).
DECLARE LOOP PIC X(12).
PRINT '<FF>'
PRINT '**************************************************...'
PRINT SKIP,'(1) READ'
PRINT SKIP,'(2) MODIFY'
PRINT SKIP,'(3) ERASE'
PRINT SKIP,'(4) PRINT'
PRINT SKIP,'(5) ADD'
PRINT SKIP,'(6) EXIT TO DATATRIEVE'
PRINT SKIP,'(7) EXIT TO DCL'
PRINT '**************************************************...'
PRINT SKIP
SELECTION = *.'selection'
CHOICE
       SELECTION = '1' THEN ACTION = "EXECUTE PROC1"
       SELECTION = '2' THEN ACTION = "EXECUTE PROC2"
       SELECTION = '3' THEN ACTION = "EXECUTE PROC3"
       SELECTION = '4' THEN ACTION = "EXECUTE PROC4"
       SELECTION = '5' THEN ACTION = "EXECUTE PROC5"
       SELECTION = '6' THEN ACTION = "EXECUTE PROC6"
       SELECTION = '7' THEN ACTION = "EXECUTE PROC7"
       ELSE  ACTION = "! ERROR ENTER A NUMBER FROM SELECTION"
END_CHOICE
IF SELECTION LT 6 THEN LOOP = "EXECUTE MENU"
IF SELECTION GE 6 THEN LOOP = "! NO ACTION"
OPEN [USER.HELPER]HELPER.COM
PRINT ACTION (-)
PRINT LOOP (-)
CLOSE
RELEASE SELECTION
RELEASE ACTION
RELEASE LOOP
@[USER.HELPER]HELPER
FINISH
END_PROCEDURE
```

Example of PROC6 -
! Escape to DTR

```
REDEFINE PROCEDURE PROC6;
PRINT 'EXIT FROM MENU'
END_PROCEDURE
```

Example of PROC7 -
! Escape to DCL

```
REDEFINE PROCEDURE PROC7
EXIT
END_PROCEDURE
```

Example of HELPER file produced when SELECTION=6

```
EXECUTE PROC6
! NO ACTION
```

<<WARNING>>

Nesting Limit Kicks You Out if You Don't Exit

****

? Datatrieve Does Windows ?

Jeff Leving - Abilene Christian University, Abilene, TX

****

Jeff presented a technique for using ASCII cursor control in Datatrieve PRINT statements to provide a "Window" of Help Text.

Scenario: A record field often requires a code value instead of a full description and there are several value from which to choose. An example may be a department code for a company with only 12 major departments. Because the departments are fairly static, you decide to use the VALID IF clause in the record definition, rather than creating a dictionary table.

Problem: When you create a procedure for someone to add employees (records) to the personnel file, you need to display the valid department codes from which he/she can choose.

Solution: Use the PRINT statement to manipulate the screen regions and cursor control to display code options and their related descriptions during the prompt for input.

Method:

1.) Hint: to embed ASCII escape sequences (e.g. <ESC>[2J = clear screen), bring your procedure into EDT and use the following sequence of keystrokes to create the <ESC> character: PF1,27,PF1,KP3

2.) At the beginning of the ADD procedure, set the scrolling region for lines 1 through 20 (<ESC>[1;20r), clear the screen, and set the cursor to the top of the screen (<ESC>[1;1H)

3.) When prompting the user for fields during the ADD, I use the following method:

```
        STORE domain USING BEGIN
                field1 = *."prompt text"
                field2 = *."prompt text"
                    ...
                fieldn = *."prompt text"
            END
```

4.) For the field that requires one of the 12 code values, "prompt text" gets messy. So I show the values in a "window" at the bottom of the screen (lines 21 thru 24) as follows:

a.) save the current cursor position and attrib's (<ESC>7).
b.) position the cursor in the top left of window (<ESC>[21;1H) and clear the window (<ESC>[J)
c.) display the codes (Hint: if you need more than 80 columns, embed a <LF><CR> sequence as follows: PF1,10,PF1,KP3,PF1,13,PF1,KP3)
d.) restore cursor position and attrib's (<ESC>8) and finish the prompt text.

Example:

1.) Record definition:

```
...
03 DEPT_CODE    PIC X(3)
   VALID IF DEPT_CODE = "A01","B02","C03", etc.
...
```

2.) Procedure ADD text:

```
! Set up scrolling region, clear screen, home
PRINT "<ESC>[1;20r<ESC>[2J<ESC>[1;1H"

PRINT "Please enter new employee data:, skip
STORE PERSONNEL USING BEGIN
  EMPLOYEE_ID = *."5-digit ident no."
  LAST_NAME = *."last name (max. 15 letters)
      ...
  DEPT_CODE = *."<ESC>7<ESC>[21;H<ESC>[JValid -
     options<LF><CR>   A01=Advertising; -
     B02=Bus. Office;C03=Computer Services; -
     ...<ESC>8dept. code"
      ...
 END
```

3.) What is displayed:

```
-----------------------------------------------------------------
Please enter new employee data:

Enter 5-digit ident. no.:  12345
Enter last name (max. 15 letters):  Leving
```

```
        ...
Enter dept. code:  #



       (line 20)
Valid options:
A01=Advertising;B02=Bus. Office;C03=Computer Services;...
```

Other Uses:

- o Displaying special error messages (from complex validation, invalid menu options, etc.)
- o "I'm working" messages for routines that may be a bit slow
- o Status messages

Caution!:

- o Maximum length for text with a PRINT statements is 247 chars.
- o You may need <ESC>< to set the terminal to ASCII
- o If you modify the VALID IF clause, be sure to update your windows!
- o There may be unexpected results on VT100's w/o the AVO cards.

****

The Case of the Missing Quotes
Dale Weathers - Summer Institute of Linguistics

****

"This is barely magic and barely Datatrieve and I can probably barely communicate what I do but I do it all the time because I can barely type. The problem is the 'missing quote'. You make this big long procedure (940 lines). (You might wonder why I have a procedure which is 900 lines or longer but that's because of some other magic that I do.) When your done, you try to run it and you get

```
                - Expected Quote ... encountered ...
```

so you say 'Oh no, every PRINT statement's got all kinds of literals in it.' So you go in and edit the procedure [using screen mode, go to the top, and find out how many lines there are.]"

```
               $ EDIT procedure
               * SHO BUFFER
                 =MAIN 940 LINES
```

"The first thing I want to do is to make sure that all the quotes are the same kind, either single or double quotes."

```
               * S /'/"/REST           ! ' -> "
```

"Then I substitute double quotes for double quotes and it tells me how many times it replaced it."

```
               * S /"/"/REST           ! on 940 lines
                 REPLACED 401 TIMES
```

"You say, 'uh oh, I obviously have an odd number.' That was the problem to begin with but where is the odd quote in the 940 lines of code? Well, if VMS can do a binary search so can I!"

```
               * +400                  ! move down 400 lines
               * S/"/"/REST            ! on 540 lines
                 REPLACED 201 TIMES
               * +240                  ! move down 240 lines
               * S/"/"/REST            ! on 300 lines
                 REPLACED 100 TIMES    ! oop's went too far
               * -120                  ! move up 120 lines
               * S/"/"/REST            ! on 420 lines
                 REPLACED 150 TIMES    ! didn't move up enough
               * -60                   ! move up 60 lines
               * S/"/"/REST            ! on 480 lines
                 REPLACED 173 TIMES
```

"I could do this one more iteration but I know that the problem is between here and the bottom of the buffer." The problem quote is within the next 23 occurrences (173 - 150) so it's easy at this point to enter screen mode and manually locate the errant quote.

One problem... What do you do if you've included comments like...

```
       ! Don't ...
```

What you can do is, at the beginning, search for occurrences of N'T and substitute N`T.

[Question from audience:] "Why do you have 900 line procedures?"

[Dale's response:] "That's because I write more compact code than one of the other guys in my shop that has..."

****

SELECT in BEGIN-END
Tom Greer - Boeing, Seattle, WA

****

"They always told me that you can't use a SELECT within a BEGIN-END loop. One day, by mistake, I tried it.

We know you can do this:

```
        FIND A IN FOO
        SELECT
```

but you can also do this and it still works!

```
        BEGIN
                PRINT
                MODIFY NAME,...
        END
```

[Comment from Phil Naecker:] "... in the last issue (April 1985) of the newsletter, you'll want to read about the internals of Datatrieve to find out why you can do that and when you can't.

****

More Select Inside BEGIN_END Block
Frank Schipani - ACSC, Decatur, GA

****

"The application is a data entry system where there are multiple data entry operators and each one has to be assigned a unique data entry reference number. The mechanism is to have a file of forms with two elements, form name (FORMNAME) and data entry reference number (DERNO)."

```
       01 FORMLIST.
          03 FORMNAME PIC X(8).
          03 DERNO PIC 9(8) COMP.
```

"It is accessed in the following way. The FIND statement for FORMLIST occurs before the beginning of the block. Within the block I have a SELECT (which locks the record.)" The data entry reference number is assigned to a temporary variable and is incremented by one. The record is then de-SELECTed to release the record thereby making it available to the next operator.

```
       DECLARE TEMPDERNO PIC 9(8) COMP.
       FIND FORMLIST WITH FORMNAME = "MAINFORM"
       WHILE DOMORE EQ "Y"
       BEGIN
```

```
               SELECT
               TEMPDERNO = DERNO
               MODIFY USING DERNO = TEMPDERNO + 1
               SELECT NONE
               PRINT "Data Entry Ref #:",TEMPDERNO (-)
               STORE MAINFORM USING
               BEGIN
                       OPNAME = *.'Operator Name'
                               .
                               .
                               .
```

****

"NOT FIND"
Warren Alkire - Abilene Christian University, Abilene, TX

****

"My registrar came to me and said, 'We're a little low on registration this term and what the administration as asked me to do is to increase our registration by pre-registering people who are already here.' Being a math major, I always liked Boolean logic and I figured that if you can do a FIND why can't you do a NOT FIND? What I want to do is this... I want to find all the people that are in this group (861) but not in this group (864)." "This is done with an extract and two or three passes through the file."

```
       01 RE-REC.
          03 KEY.
             05 TERM   PIC X(03).  !The term registered
             05 CAMPUS PIC X(02).
             05 SID    PIC X(09).  !Student ID number
          03 DATA.
                .
                .
                .
       ;

       01 EXTRACT.
          03 KEY.
             05 TERM   PIC X(3).
             05 CAMPUS PIC X(2).
             05 SID    PIC X(9).
          03 FLAG      PIC X(1).
       ;

       READY RT SHARED
       DEFINE FILE FOR EXTRACT         !Create a temporary file
       READY EXTRACT WRITE

       !After the  next statement all students registered in term
```

```
!one with have a record in EXTRACT.

FOR RT WITH KEY STARTING WITH "861"
        STORE EXTRACT USING
         BEGIN
                TERM    = "864"
                CAMPUS  = CAMPUS
                SID     = SID
                FLAG    = "Y"
         END

!After the next  statement,  all  students  registered  in
!term one AND in  term  four  will  have  the  FLAG = "N";
!students registered ONLY in term one will have FLAG="Y".

FOR EXTRACT CROSS RT OVER KEY MODIFY USING FLAG = "N"

!Since  all  records  in  extract   came  from  term  one,
!re-modify TERM to reflect this.

FOR EXTRACT MODIFY USING TERM = "861"

!The following RSE essentially does a NOT FIND!

REPORT EXTRACT CROSS RT OVER KEY WITH FLAG = "Y"
        .
        .
        .
```

****

"Fun With Your LA-50"

Donald E. Stern, Jr. - Warner Lambert Co., Milford, CT

****

"I've got a couple of things to share... First, a couple of months back I published an article showing how to center the result of the PLOT HARDCOPY statement on your attached LA-50 printer. (One of the things that you get to do, as the newsletter editor, is publish your own material.)"

As delivered, the DEC version of the HARDCOPY plotting routine is made up of the following code.

```
DEFINE PLOT HARDCOPY
ENTRY 0
    BEGIN
        PLOT HOUSEKEEP 4
        OUTPUT_SEGMENT 5
        CLEAR_SEGMENT 4
        SET_SEGMENT 4
        PRINT 'S(H)S(E)P[100,200]@BS(H)'
        OUTPUT_SEGMENT 5,4,6
        PLOT HOUSEKEEP 2
    END
END_PLOT
```

In the "vanilla" version, the default offset is 50 pixels off the left margin. If one simply modifies the 'PRINT' line as shown below the output will be more or less centered on 8.5 x 11 in. tractor feed paper.

```
DEFINE PLOT HARDCOPY_CENTERED
ENTRY 0
    BEGIN
        PLOT HOUSEKEEP 4
        OUTPUT_SEGMENT 5
        CLEAR_SEGMENT 4
        SET_SEGMENT 4
        PRINT 'S(H(P[275,0]))S(E)P[100,200]@BS(H)'
        OUTPUT_SEGMENT 5,4,6
        PLOT HOUSEKEEP 2
    END
END_PLOT
```

At this and past Wombat Magic sessions, it was shown how escape sequences can be programmed into Datatrieve procedures in order to achieve screen control. Escape sequences can also be used to control your LA-50 (or LA-100, etc.) printer. The following code fragment from my DTR startup file shows some of these escape sequences. (Please note that <ESC> represents a single byte which is the ASCII code for escape, decimal 27.)

```
DECLARE BOLD COMPUTED BY "<ESC>[1m" QUERY-HEADER IS -.
DECLARE FF   COMPUTED BY "<FF>" QUERY_HEADER IS -.
DECLARE ROTATE   COMPUTED BY "<ESC>[?47h" QUERY_HEADER IS -.
DECLARE COMPRESS COMPUTED BY "<ESC>[?43l" QUERY_HEADER IS -.
DECLARE EXPAND   COMPUTED BY "<ESC>[?43h" QUERY_HEADER IS -.
DECLARE LAON  COMPUTED BY "<ESC>[5i" QUERY-HEADER IS -.
DECLARE LAOFF COMPUTED BY "{{ESC}}[4i" QUERY-HEADER IS -.
DECLARE LA132 COMPUTED BY "<ESC>[4w" QUERY-HEADER IS -.
DECLARE LA100 COMPUTED BY "<ESC>[2w" QUERY-HEADER IS -.
DECLARE LA80  COMPUTED BY "<ESC>[1w" QUERY-HEADER IS -.
DECLARE LA64  COMPUTED BY "<ESC>[8w" QUERY-HEADER IS -.
DECLARE LA48  COMPUTED BY "<ESC>[6w" QUERY-HEADER IS -.
DECLARE LA40  COMPUTED BY "<ESC>[5w" QUERY-HEADER IS -.
DECLARE VP12  COMPUTED BY "<ESC>[3z" QUERY-HEADER IS -.
DECLARE VP8   COMPUTED BY "<ESC>[2z" QUERY-HEADER IS -.
DECLARE VP6   COMPUTED BY "<ESC>[1z" QUERY-HEADER IS -.
DECLARE VP4   COMPUTED BY "<ESC>[6z" QUERY-HEADER IS -.
DECLARE VP3   COMPUTED BY "<ESC>[5z" QUERY-HEADER IS -.
DECLARE VP2   COMPUTED BY "<ESC>[4z" QUERY-HEADER IS -.
```

The following code demonstrates the use of four of these escape sequences to cause the Datatrieve report writer to produce a hardcopy report, on an attached printer, in which the title and column headers are written with bolded print.

```
REPORT FIRST 40 YACHTS
SET REPORT_NAME="Demonstration Report of YACHTS"/"Using Bolding"
AT TOP OF REPORT PRINT LAON, NEW_PAGE
AT TOP OF PAGE PRINT BOLD, SKIP 1, REPORT_HEADER, SKIP 2,
  COLUMN_HEADER, OFF
PRINT YACHT
AT BOTTOM OF REPORT PRINT LAOFF
END_REPORT
```

The new VT240 and VT241 terminals are capable of producing graphics output in which the image produced is COMPRESSED (normal), EXPANDED, or ROTATED. The escape sequences assigned to the variables COMPRESS, EXPAND, and ROTATE can be used to select any of the modes. For example, the Datatrieve statement

```
PRINT COMPRESS THEN PLOT HARDCOPY_CENTERED
```

will insure that a centered plot of normal size will be printed regardless of the default hardware setup of the terminal.

If another hardcopy plot is created in which the offset is explicitly set to zero (See HARDCOPY_NOOFFSET below) then the following Datatrieve statement will result in a rotated plot being printed. [Pat Scopaletti points out that rotated plots are 12 inches long and therefore might not fit on 11 inch paper.]

```
PRINT ROTATE THEN PLOT HARDCOPY
```

Here is the code for a hardcopy plot with no offset.

```
DEFINE PLOT HARDCOPY_NOOFFSET
ENTRY 0
    BEGIN
        PLOT HOUSEKEEP 4
        OUTPUT_SEGMENT 5
        CLEAR_SEGMENT 4
        SET_SEGMENT 4
        PRINT 'S(H(P[0,0]))S(E)P[100,200]@BS(H)'
        OUTPUT_SEGMENT 5,4,6
        PLOT HOUSEKEEP 2
    END
END_PLOT
```

It is possible to control the horizontal and vertical pitch of an attached LA-50 printer using some of the escape sequences given above. Examples of their use are given below.

```
!  The following  statement  will  turn  the attached printer on,
!  print the first  10  yachts using the current default vertical
!  and horizontal pitch, and finally turn the printer off.
!
```

```
PRINT LAON THEN
PRINT MANUFACTURER, MODEL, RIG, PRICE OF FIRST 10 YACHTS THEN
PRINT LAOFF

!  The following statement turns  the  printer on, sets the pitch
!  to 16.5 chars/inch (132 columns/8 inches), prints the first 10
!  yachts, and turns the printer  off.   (NOTE:  The printer will
!  be left in compressed print mode.)
!
PRINT LAON, LA132 THEN
PRINT MANUFACTURER, MODEL, RIG, PRICE OF FIRST 10 YACHTS THEN
PRINT LAOFF

!  The following statement turns the attached printer  on,  sets
!  the horizontal pitch to 5.0 chars/inch (40 columns/8 inches),
!  the vertical pitch to 3 lines/inch, prints the first 10 yachts
!  and then resets the horizontal pitch to 10  chars./inch  (80
!  columns/8 inches), the vertical pitch to 6 lines/inch, and
!  finally turns the printer off.
!
PRINT LAON, LA40, VP3 THEN
PRINT MANUFACTURER, MODEL, RIG, PRICE OF FIRST 10 YACHTS THEN
PRINT LA80, VP6, LAOFF
```

Datatrieve/4GL SIG Sponsored Sessions in San Francisco

Symposium Coordinator - Chris Wool

Dallas Symposium attendees indicated to the SIG officers that more technically oriented sessions were needed. This is what we have done for San Francisco. In addition, a healthy mix of non-DTR 4GL sessions have been included in the program. San Francisco promises to have something for everyone in a full five days of sessions! What follows is a chronological listing of scheduled sessions together with abstracts for those sessions.

DT040 - DATA MANAGEMENT SYSTEMS AND DATATRIEVE/4GL SIGS OPENING SESSION
Monday October 6, 1986 9:00 a.m. - 10:00 a.m.

This is the joint opening and roadmap session for the Data Management Systems SIG and DATATRIEVE/4GL SIG. Come to get a look at what these SIGs do, and what sessions and activities are planned by them for the week of the symposium. SIG Steering Committee members and Digital representatives are introduced.

DT019 - DECREPORTER - A BUSINESS REPORTING PACKAGE
Monday October 6, 1986 1:00 p.m. - 2:00 p.m.

DECREPORTER is a VAX/VMS product which allows end-users to create reports by responding to prompts. DECREPORTER is dictionary-driven, using the VAX Common Data Dictionary (CDD), and may be linked with VAX DATATRIEVE for access to Rdb/VMS and VAX DBMS data. This session outlines DECREPORTER's functional characteristics and new features, as well as presents examples on how to customize CDD records and domains to enhance user friendliness.

DT042 - WHAT ARE FOURTH GENERATION LANGUAGES ANYWAY?
Monday October 6, 1986 1:00 p.m. - 2:00 p.m.

Fourth generation languages (4GLs) are many things to different types of users. To programmers, 4GLs are prototyping and productivity tools; to casual users, they are an easy means to manipulate and report data; and to data managers, they provide methods to easily access data required by a variety of applications.

This panel consists of 4GL users and developers with different perspectives on the attributes of 4GLs and how they use these tools. These discussions represent generic 4GLs and are not a comparison of available products.

DT035 - POSITIONING DIGITAL'S FOURTH GENERATION PRODUCTS
Monday October 6, 1986 2:00 p.m. - 3:00 p.m.

This session compares various fourth generation language (4GL) products which have been developed by Digital, discussing the environments for which they are intended, the intended user of each product, and the capabilities of each.

DT007 - BEGINNER'S GUIDE TO DATATRIEVE
Monday October 6, 1986 3:00 p.m. - 4:00 p.m.

DATATRIEVE is currently one of Digital's most popular products. It is extraordinarily effective in providing a robust means for interactive access to data in Record Management System (RMS) files, VAX/Rdb databases, or VAX DBMS databases on VAXes, PDP-11s, PROs and DECSYSTEM-20s. This session presents a basic description of the essential characteristics of DATATRIEVE and describes situations in which it is effective. At the end of this session, attendees should understand the capabilities of DATATRIEVE, strategies for using it, and typical situations in which it is effective.

DT033 - APPLICATION OF TOP-DOWN DESIGN TO SOPHISTICATED DATABASE APPLICATIONS USING CDD, DATATRIEVE, AND TDMS
Monday October 6, 1986 5:00 p.m. - 6:00 p.m.

This session presents methods for applying top-down design and other sound principles from database theory to sophisticated database applications implemented using the Common Data Dictionary (CDD), DATATRIEVE, and TDMS. Fourth generation data management tools present inherent difficulties to applying such principles. On the other hand, there is great power in these tools if applications are built to enhance their advantages instead of fighting their limitations. Digital's "layered software" approach to its VAX Information Architecture also introduces difficulties in applying these principles. The tradeoffs involved and recommended approaches to resolving these contradictions are explained using a liberal number of examples from two database applications - one supporting aircraft maintenance, the other a personnel information system.

DT025 - HOW I SOLVED MY CODE MANAGEMENT NIGHTMARE USING FMS, DATATRIEVE, AND RDB
Monday October 6, 1986 6:00 p.m. - 7:00 p.m.

Code management, or the tracking of software distribution on a multitude of VAXes, can quickly turn into a nightmare without some form of automated assistance. This session briefly describes the problems associated with tracking products and versions onto respective systems, and also the steps we have taken to solve these problems through DATATRIEVE applications.

The utilization of the Forms Management System (FMS) and Rdb/VMS through DATATRIEVE is explored in depth with examples and instructions for building applications using these tools. This discussion is aimed at users with some experience using DATATRIEVE and Rdb. No prior experience with FMS is needed.

DT043 - DATATRIEVE APPLICATION DESIGN TUTORIAL
Monday October 6, 1986 7:00 p.m. - 8:00 p.m.

This session describes the way in which a typical application might be analyzed and implemented using DATATRIEVE. The presentation includes initial design considerations, implementation, and performance trade-offs.

This session presents information from Digital performance analysts concerning various aspects of VAX DATATRIEVE. It discusses design issues (e.g., FOR versus FIND, query optimization, CROSS, nested FOR loops, etc.), Record Management System (RMS) file design and tuning, and some Common Data Dictionary (CDD) considerations.

DT037 - PERFORMANCE MANAGEMENT FOR DATATRIEVE APPLICATIONS
Monday October 6, 1986 8:00 p.m. - 9:00 p.m.

This presentation discusses the management of performance (speed and resource consumption) for DATATRIEVE applications. It presents specific information to assist you in determining if you have a DATATRIEVE performance problem; and if so, how to fix it. Topics covered include an overview of performance management; the tuning and monitoring tools available to you; how DATATRIEVE's architecture and internals affect performance; and strategies for improving performance. Hints and kinks for the developer interested in building efficient DATATRIEVE applications also are presented.

Since much of the session material applies to almost any data management application on a VAX (including applications in COBOL, BASIC, and other third generation languages), attendees who do not use DATATRIEVE will also be interested in this talk. Bring your System Manager as well, as many of the topics are pertinent to the management of systems with heavy DATATRIEVE usage.

DT041 - ADVANCED RMS FILE DESIGN AND TUNING FOR DATATRIEVE PERFORMANCE
Monday October 6, 1986 9:00 p.m. - 10:00 p.m.

DATATRIEVE performance is dominated by file design and file disorganization. This presentation uses DATATRIEVE graphs to show how the performance monitoring techniques within DATATRIEVE can be used to determine which factors affect optimal file design and tuning. This presentation discusses the following topics in-depth: how to use the Record Management System (RMS) utilities to determine the level of disorganization of a file; how to optimize files using the File Definition Language (FDL); and how to reload files.

DT036 - RMS BASED 4GL DEVELOPMENT TOOLS
Tuesday October 7, 1986 9:00 a.m. - 11:00 a.m.

This session is an overview session which describes application development tools, with emphasis on a database manager and an application generator, and how these tools make it possible to create sophisticated applications without Rdb, FMS, CDD, etc.

There is an in-depth discussion of the A-to-Z Database Manager and A-to-Z Application Generator and how these products are used to reduce the amount of development effort involved in creating fully integrated applications. This session includes a live demonstration of each product to help show the power of these products as application development tools.

DT048 - APPLICATION DEVELOPMENT USING INTELLECT/RDB
Tuesday October 7, 1986 1:00 p.m. - 2:00 p.m.

This session briefly introduces the capabilities of INTELLECT, a natural language front end to VAX Rdb/VMS, which allows English language query update and definition of Rdb databases. The primary focus of the session is on the application development cycle for INTELLECT.

Development of a fluent and robust INTELLECT application which meets the needs of its target audience is similar to any other end-user application development process. In this session we examine the lexicon development cycle and consider what should be done in each phase to ensure success. Topics include introducing potential users to INTELLECT and assessing their needs; choosing or designing a database for English query; developing the lexicon; introducing the application to end users; assessing its performance; and strategies for effective lexicon maintenance. Attendees should gain an understanding of the overall implementation process and receive practical guidelines for success with INTELLECT.

DT013 - AN EXAMINATION OF USES OF ACCENT R
Tuesday October 7, 1986 2:00 p.m. - 3:00 p.m.

The first portion of this presentation discusses the selection process. A brief outline is given of the following criteria used in the selection of a database system: a) user friendliness, b) programmer productivity, c) ease of use, d) DECSYSTEM-20/VAX syntax similarity, and e) performance.

Next, the session presents the current uses of Accent R at Fairfield University: a) academic, including teaching course; b) accounts receivable; c) student records, including biography, grading, transcripts, registration, class lists, and housing; d) financial aid; e) purchasing; f) alumni, including pledge processing, biography, and reunion events; g) admissions, including biography, inquiry, reporting, acceptance letters, and survey; h) personnel, including biography and reporting; and i) survey analysis.

Each application is described in the light of Accent R, which permitted rapid and elegant programming. Wherever possible, a comparison is made with conventional programming tools.

There is brief outline of the details of Accent R to introduce the audience to the features of this database system, so that attendees may leave with concrete information about the strengths and weaknesses of Accent R, thereby assisting them in making a choice.

DT009 - DATATRIEVE SOFTWARE CLINIC
Tuesday October 7, 1986 3:00 p.m. - 5:00 p.m.

This session allows the DATATRIEVE user to obtain one-on-one help from an experienced DATATRIEVE wizard. A DATATRIEVE/Fourth Generation Languages Special Interest Group (DTR/4GL SIG) representative meets you at the door and directs you to an appropriate expert, who spends as much time with you and your problem as you require. All levels of inquiries are accepted, from the beginner to the 'seasoned' DATATRIEVE professional. The experts consist of Digital developers and experienced members of the DTR/4GL SIG.

DT006 - ADVANCED DATATRIEVE RECORD DEFINITIONS
Wednesday October 8, 1986 9:00 a.m. - 10:00 a.m.

The record definition is the basis for all data retrieval within DATATRIEVE. There are many possibilities for the manipulation of data, and the presentation of that data, which are not immediately obvious nor presented in the documentation. This session is intended for those persons who are familiar with DATATRIEVE and are looking for additional methods of describing data, who have to interface with files written by other programs/products, or who need additional options in developing special applications.

DT027 - ADVANCED REPORT WRITING TECHNIQUES IN VAX DATATRIEVE
Wednesday October 8, 1986 10:00 a.m. - 11:00 a.m.

The DATATRIEVE Report Writer is made up of just a few single statements that may be used to create simple or complex reports. Because of the many defaults, it is easy for a novice to get started; however, users soon find they need more than just the standard defaults. With a little imagination and creativity, the power and flexibility of VAX DATATRIEVE makes it possible to create almost any report you need. A series of examples are discussed which demonstrate advanced and unusual applications.

DT008 - USING THE DATATRIEVE CALL INTERFACE
Wednesday October 8, 1986 11:00 a.m. - 12:00 noon

This session explains and demonstrates the use of the DATATRIEVE call interface. Using the call interface, the user can incorporate a highly effective interactive capability into an application; add functionality to DATATRIEVE by using user-defined keywords; provide very effective menu-driven applications; and, in general, combine the best of both third generation languages, such as COBOL, BASIC or FORTRAN, and the powerful capabilities of DATATRIEVE. Although this session is an advanced session, the emphasis is on demonstrating how easy it is to use the call interface.

DT005 - MANAGING YOUR ALL-IN-1 SYSTEM WITH DATATRIEVE
Wednesday October 8, 1986 12:00 noon - 12:30 p.m.

The tools provided by Digital for managing an ALL-IN-1 system can be tedious and cumbersome. This session provides techniques for managing an ALL-IN-1 system using DATATRIEVE.

DT030 - SPATIAL GRAMMAR - LEAVING QUERY LANGUAGES SPEECHLESS?
Wednesday October 8, 1986 2:00 p.m. - 3:00 p.m.

Both fourth generation languages and natural language systems have linear, word-oriented grammars which the user must type. Like other command languages, they are sensitive to typographic, syntactical, and semantic errors. A spatial grammar, by contrast, is graphic, two-dimensional, and uses a pointing device for command input. PICTOR is a Digital Standard Relational Interface (DSRI)-based query/application language implemented with a spatial grammar. This session compares spatial grammar concepts and linear grammar concepts by comparing PICTOR and DATATRIEVE, with particular emphasis on human interface issues.

DT044 - 4GL APPLICATION DEVELOPMENT GUIDELINES
Wednesday October 8, 1986 3:00 p.m. - 4:00 p.m.

There are no current comprehensive guidelines for developing applications with fourth generation languages (4GLs) which have achieved even a minimal level of acceptance in the commercial data processing world. Time and time again we have seen that performance problems are directly related to a good approach toward development and the quality of the application design. We have seen a failure to follow a development approach that effectively utilizes a 4GL as a development tool. Lately, experienced system developers have joined together to agree on a common approach to application development. The result is a set of guidelines which have been successfully proven.

This session provides a basic overview of Application Development Guidelines (ADG), and why they are important. It includes a set of overhead slides and covers the steps of a basic approach using 4GLS.

DT017 - VAX DATATRIEVE'S INTERFACE INTO RELATIONAL DATABASES: HOW IT WORK?
Wednesday October 8, 1986 4:00 p.m. - 5:00 p.m.

This session presents a technical overview of how VAX DATATRIEVE interfaces into Digital's family of relational database products. The first portion of the talk covers the user-level interface into DATATRIEVE, and how it relates to relational databases. The second portion of the talk covers the more technical aspect of the interface, specifically, covering the internal architecture and design of DATATRIEVE's interface into the relational database itself. Some general areas of coverage include transaction considerations, locking and access considerations, and a comparison of DATATRIEVE commands and how they map into relational databases.

DT047 - USING RDB AND DATATRIEVE TOGETHER: LESSONS LEARNED
Wednesday October 8, 1986 5:00 p.m. - 6:00 p.m.

This session addresses lessons learned using the VAX layered products Rdb and DATATRIEVE together. People with novice to intermediate knowledge of either product will want to attend. Topics include: a) how using both products expands your math function capabilities; b) a comparison of CROSS clauses between Rdb and DATATRIEVE; c) ways we expanded DATATRIEVE graphics capabilities to meet the requirements of scientific applications; d) lessons learned regarding DATATRIEVE host language call interfaces; e) school of hard-knocks knowledge regarding Rdb host interfaces.

DT003 - THE BEST OF WOMBAT MAGIC
Wednesday October 8, 1986 6:00 p.m. - 7:00 p.m.

In the past, Wombat Magic sessions have provided an enormous wealth of information regarding clever ways to use DATATRIEVE. This session consolidates the best of the magic from past sessions into a single session. Items of interest for both beginning and advanced users are presented at this session.

DT031 - CALLING DATATRIEVE-11 FROM HIGH-LEVEL PDP-11 LANGUAGES
Thursday October 9, 1986 9:00 a.m. - 9:30 a.m.

The DATATRIEVE-11 call interface facility allows for interfacing high level PDP-11 languages with DATATRIEVE. This interface allows access to DATATRIEVE-11's unique dictionary and its facilities to manipulate and manage data. Merging DATATRIEVE-11's data handling capability and a high level language's speed and ability to do complicated functions allows new or revised systems to be easily brought up and tested. This also allows for more prototyping of new applications and greater programmer productivity.

This session provides a comprehensive look at the call interface facility including an overview, a detailed look at each of the calls, some common problems users encounter, and some hints on how you can better utilize the facility. A general knowledge of writing DATATRIEVE procedures is assumed.

DT032 - DATATRIEVE-11 INTERNALS AND PERFORMANCE OPTIMIZATION
Thursday October 9, 1986 9:30 a.m. - 10:30 a.m.

DATATRIEVE-11 is an interactive query, report, and data maintenance system that is well suited for low-cost, multi-user systems. At this session, a software engineer from Digital's DATATRIEVE-11 Development Group will provide helpful hints on how to optimize DATATRIEVE-11 applications for improved performance.

DT016 - VAX DATATRIEVE INTERNALS
Thursday October 9, 1986 10:30 a.m. - 11:30 a.m.

This session first addresses the status of current versions of the VAX DATATRIEVE product. Following the status report, this session provides an overview of some of the internals of VAX DATATRIEVE, and provide some suggestions for optimizing DATATRIEVE applications based on these internals. Some of the areas covered include DATATRIEVE invocation and start-up, procedure and loop organization alternatives, and trade-offs during the execution phase of DATATRIEVE.

DT024 - VAX TEAMDATA - END-USER INFORMATION MANAGEMENT
Thursday October 9, 1986 4:00 p.m. - 5:00 p.m.

This session describes VAX TEAMDATA, a new end-user information management product from Digital. VAX TEAMDATA provides powerful, yet easy to use, information management capabilities to those who need to use data in their work, but who do not want to do data processing. The session presents how TEAMDATA lets users easily store and modify data, using a screen-oriented, text-editing style, in a powerful relational database management system (VAX Rdb/VMS). Examples of how TEAMDATA users can perform their tasks by selecting items from strip menus, using a command language, or a combination of the two, is presented. This discussion covers how TEAMDATA allows users to manipulate their personal data as well as shared or remote databases, as simple tables, in spreadsheets, reports, and graphs, and to perform complex query and data reduction operations. This session illustrates how VAX TEAMDATA is the solution to the need many non-computer professionals have for access to information for use in a broad range of decision support activities.

DT023 - VAX RALLY TECHNICAL OVERVIEW
Thursday October 9, 1986 5:00 p.m. - 6:00 p.m.

This session provides a technical overview of a new Digital fourth generation product for rapid prototyping and development of forms, memos, reports, and relational databases. The session also describes how RALLY can be used to solve sophisticated information management problems in a distributed VAX environment.

DT020 - DESIGNING AN APPLICATION USING THE VAX FORMS MANAGEMENT SYSTEM AND DATATRIEVE
Thursday October 9, 1986 6:00 p.m. - 7:00 p.m.

This session takes a top-down approach to the design, coding, and implementation of a software package created using the VAX Forms Management System (FMS) and VAX DATATRIEVE. The attendee should gain a knowledge of creating forms and form libraries, and using VAX DATATRIEVE to add, modify, and retrieve data stored in Record Management System (RMS) sequential and indexed-sequential files using the VAX forms management interface.

DT012 - WRITING MENU-DRIVEN SYSTEMS IN VAX DATATRIEVE
Thursday October 9, 1986 7:00 p.m. - 8:00 p.m.

There are four commonly used methods for providing menus for systems written in VAX DATATRIEVE: writing the menu using interactive DATATRIEVE; writing the menu using logicals; writing the menu using DCL; writing the menu using callable DATATRIEVE. This session demonstrates how to create a menu using each method. In addition, a comparison of the methods is presented. This comparison emphasizes the cause and length of the user's waiting time, and the impact that each menu method has on system resources.

DT002 - WOMBAT MAGIC
Thursday October 9, 1986 8:00 p.m. - 10:00 p.m.

Wombat Magic is an enchanted time when DATATRIEVE wizards and novices alike gather to share their magic. Magic can include special or unusual solutions to problems, unique or new ways to use DATATRIEVE, etc. Bring your magic, simple or esoteric, to share with novices or wizards. Remember, what may not seem like magic to you may be the answer to another user's problem. Prizes will be awarded for best magic, and often just for being present.

DT011 - WHAT'S WRONG WITH FOURTH GENERATION LANGUAGES
Friday October 10, 1986 9:00 a.m. - 10:00 a.m.

This is a general discussion of the current state of fourth generation languages (4GLs). It briefly touches upon the definition and use of 4GLs and how they fit into, and have changed, the entire spectrum of typical application development.

This session examines the two basic strategies that 4GL vendors have taken in solving the traditional problems of application development: information center 4GLs (intended primarily for end-users) and development center 4GLs (intended primarily for data processing professionals). Included in this discussion is a look at the two different prototyping philosophies typically found in each type of 4GL.

This presentation categorizes the different types of products that are referred to as 4GLs. A closer look is made at two of these categories: relational database systems and file-independent languages. Lastly, the audience is left with some general topics to consider when investigating various 4GLs.

DT029 - EVALUATING FOURTH GENERATION LANGUAGES ACROSS THREE ENVIRONMENTS
Friday October 10, 1986 10:00 a.m. - 10:30 a.m.

This presentation discusses the process used to examine and choose a fourth generation language (4GL). Special emphasis is placed on evaluating those packages which are capable of running across three different environments: the IBM/MIV, VAX/VMS, and the PC/DOS environments. Topics include: how to identify 4GLs; how to determine company requirements; matching requirements to products; and how to identify the best product out of those which met the requirements.

DT018 - USING VAX DATATRIEVE GRAPHICS
Friday October 10, 1986 1:00 p.m. - 2:00 p.m.

The VAX DATATRIEVE PLOT statement gives users a quick and easy-to-use method of displaying information as a graph. A wide range of predefined plot formats are used to enhance decision support applications and can be produced in hardcopy form for presentations and reports. This session presents effective techniques for producing graphs from data managed with VAX DATATRIEVE.

DT004 - INTEGRATING DATATRIEVE WITH DECALC AND DECGRAPH
Friday October 10, 1986 2:00 p.m. - 3:00 p.m.

One is able to integrate DATATRIEVE with a number of other Digital products. This session presents ways in which to use DATATRIEVE with DECalc, Digital's electronic spreadsheet product, and DECgraph, a business graphics product.

DT022 - VAX DATATRIEVE SECURITY USING ENVIRONMENT ACCOUNTS AND ACCESS CONTROL LISTS
Friday October 10, 1986 3:00 p.m. - 4:00 p.m.

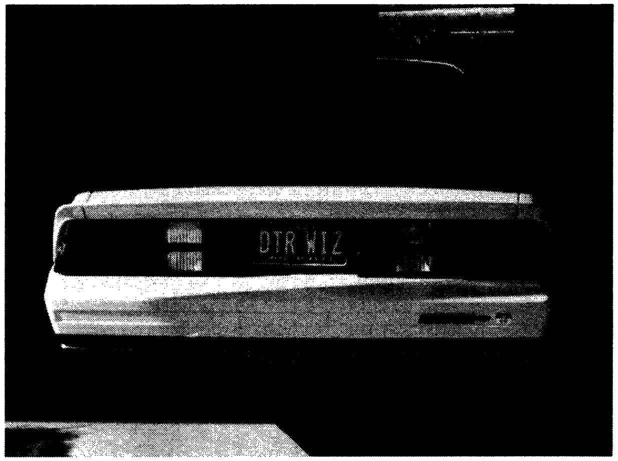
DATATRIEVE system and data security are paramount, particularly where personnel or financial data are maintained, or where sensitive failure analysis is performed. This is true for DATATRIEVE systems with single or multiple users. A relatively straightforward system design and protection scheme has been developed to assist in optimizing data integrity. Examples of several different systems are examined, together with the "CASE" method of data protection.

DT039 - DATATRIEVE/4GL SIG CLOSING SESSION
Friday October 10, 1986 4:00 p.m. - 5:00 p.m.

This session is the final session for the DATATRIEVE/4GL SIG. In this session, we wind up any loose ends that have developed during the symposium and establish plans for the next symposium. At this session, Digital developers respond to the product improvement request (PIR) items which accumulate in the Campground during the symposium.

FLASH!!!

Pheonix (WNS) - Local residents are wondering who exactly is the owner of this vehicle. Reports generated by different people tell of a mysterious, short, hairy man seen driving the car. Eye witnesses say the driver acts "...as if the road was his domain... crossing lanes, plowing through fields, and acting like a man out of context driving on the road".

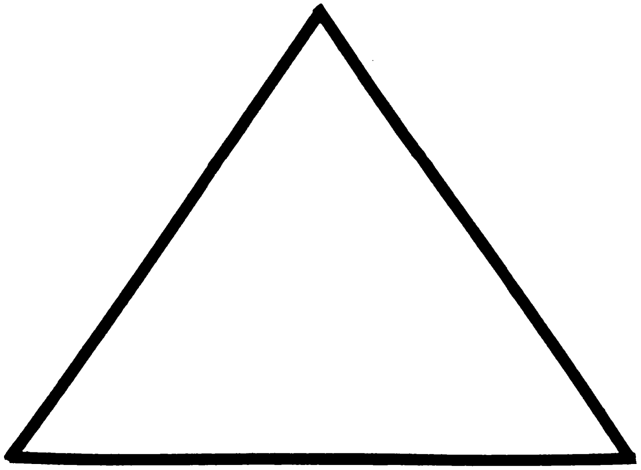


Summary of DTR/4GL SIG Sponsored Sessions in San Francisco

Monday		
	9:00-10:00am	DMS & DTR/4GL Opening Session
	1:00- 2:00pm	DECREPORTER
	1:00- 2:00pm	What are 4GLs
	2:00- 3:00pm	Positioning DEC's 4GL Products
	3:00- 4:00pm	Beginner's Guide to DATATRIEVE
	5:00- 6:00pm	Top-down Design and 4GL Tools
	6:00- 7:00pm	How I Solved My Code Management Nightmare
	7:00- 8:00pm	DATATRIEVE Application Design Tutorial
	8:00- 9:00pm	Performance Management for DATATRIEVE
	9:00-10:00pm	File Tuning for DATATRIEVE Performance
Tuesday		
	9:00-11:00am	RMS Based 4GL Development Tool
	1:00- 2:00pm	INTELLECT/Rdb Application Development
	2:00- 3:00pm	Uses of Accent R
	3:00- 5:00pm	DATATRIEVE Software Clinic
Wednesday		
	9:00-10:00am	Advanced DATATRIEVE Record Definitions
	10:00-11:00am	Advanced DATATRIEVE Reports
	11:00-12:00pm	DATATRIEVE Call Interface
	12:00-12:30pm	Managing All-In-1 with DATATRIEVE
	2:00- 3:00pm	Spatial Grammar
	3:00- 4:00pm	4GL Application Development Guidelines
	4:00- 5:00pm	DATATRIEVE Interface into Rdb
	5:00- 6:00pm	Rdb and DATATRIEVE Lessons Learned
	6:00- 7:00pm	Best of Wombat Magic
Thursday		
	9:00- 9:30am	Calling DATATRIEVE-11 from Languages
	9:30-10:30am	DATATRIEVE-11 Internals and Perf. Opt.
	10:30-11:30am	VAX DATATRIEVE Internals
	4:00- 5:00pm	VAX TEAMDATA Overview
	5:00- 6:00pm	RALLY Technical Overview
	6:00- 7:00pm	Designing Applications with DTR and FMS
	7:00- 8:00pm	VAX DATATRIEVE Menus
	8:00-10:00pm	Wombat Magic
Friday		
	9:00-10:00am	What's Wrong with 4GLs
	10:00-10:30am	Evaluating 4GLs
	1:00- 2:00pm	VAX DATATRIEVE Graphics
	2:00- 3:00pm	DATATRIEVE, DECalc, and DECgraph
	3:00- 4:00pm	DATATRIEVE Security
	4:00- 5:00pm	DTR/4GL SIG Closing

DECUS

edusig



EDUSIG

Chairman
Robert A. Shive, Jr.
Millsaps College
Jackson, MS

Symposium Coordinator
Sue Bates
Northwestern Michigan College
Traverse City, MI

Communications Committee Representative
Robert W. McCarley
Millsaps College
Jackson, MS

Newsletter Editor
Fred Bell
Taft College
Taft, CA

PSS Coordinator VAX Systems SIG Liaison
Donald C. Fuhr
Tuskegee Institute
Tuskegee Institute, AL

Administrative Applications Coordinator
Dave Cothrun
Taft College
Taft, CA

Courseware Coordinator
Mary Jac Reed
Off Comp Based Instruction
Newark, DE

DEC Counterpart
Gary Finerty
Digital Equipment Corporation
Marlboro, MA

Hello Merry-men;

My initial address may seem a bit strange but after coming from a Woods meeting it may seem more appropriate. I'm not practicing for San Francisco. As Tony used to say, that is where I left my heart. Perhaps some of you do not know what takes place in a woods meeting.

Unlike Robin Hood we do not rob from the rich and give to the poor. That is the Government's job. I wonder if they understand the concept. Anyway, in this meeting Bob Shive ,alias Robin Hood of the South, took his field generals into the woods for three days. They discussed mapping out strategy for the oncoming year. We map out the territory for the next two DECUS's. Here are some of the key issues that were brought before the table for discussion:

1. Are we offering quality presentations?

2. Should we offer the bulk of the EDUSIG presentations at the beginning of the week or the end?

3. Should speakers have to pay for transportation, lodging and registrations?

4. Should one day of registration be so costly?

5. What can we do to attract more expert speakers?

6. Should we have an speaker evaluation process for the audience to use?

7. Should there be more sessions with shorter time are less sessions with more time?

8. Why are half of the DECUS attendees first timers? Why is there such an attrition rate?

9. How do we involve more people in the EDUSIG group?

10. Could the evening bust be better spent?

11. Should there really be a DECUS theme?

12. Would it be worth the time and effort to supply a tour of the DEC floor show?

13. When is the best time to have the wrap-up-sections?

14. Should there be an EDUSIG software swap?

15. Do we need presymposia?

16. What is the interest in the Clearing House concept?

17. Is it right for your EDUSIG president to work his merry men 12 hours a day during the Woods meeting?

This should certainly erase some of the cobwebs in your brain and give you something to chew on.

Thanks, Jan, for your article on word processing. There are certainly lots of them to choose from. Also if you have not already given the Clearing House concept thought you might pursue the idea. DEC has placed DAL & CAS in the Market Basket at a reasonable price. Contact a DEC person or one of the the EDUSIG officers. We are more than willing to help.

Once again I invite your thoughts and ideas on these issues or any others. We can always use better ideas. Who has a better idea? Send your comments (60-65 characters per line, 55 lines per page) to me, we need input.

Your editor,

Fred Bell
Taft College
Taft, CA 93268
(805-763-4282)

CLEARINGHOUSE

The first edition of THE CLEARINGHOUSE Catalog is now available, offering several interesting VAX/VMS software packages for use across campus.

THE CLEARINGHOUSE FOR ACADEMIC SOFTWARE was established earlier this year by Iowa State University to provide you with moderately priced VAX/VMS software for your classrooms and research.

The educational community will gain access to a broad range of academic software for disciplines across campus, as well as the incentive to continue developing and sharing more and better academic software.

All software in this not-for-profit distribution center has been developed by educators and researchers. Owners receive a 25 percent royalty and are responsible for telephone support of their packages.

The catalog describes programs available, including their area of instructional use, the topics covered or functions performed, the targeted audience, and the hardware and software requirements. The catalog is available in electronic format as well, and can be accessed by dialing via modem (515) 294-6085. Your Username is CHCAT and your Password is also CHCAT.

The catalog also explains how to use THE CLEARINGHOUSE's online previewing system which allows you the opportunity to view a software package before ordering.

There are several interesting packages now available, including a statistics course from the University of Delaware.

The University of Delaware VAX Statistics Course

The University of Delaware VAX statistics Course is a thirty-module, one-semester, college-level tutorial on the basic concepts of statistics. It is intended for the non-math major. Under a grant from Digital, the courseware was developed by three faculty members from three different academic departments on campus. The University of Delaware has been involved with computer-based courseware development for ten years using a variety of vendors' equipment. The development of this statistics course has given the university an understanding of, and appreciation for, the flexibility and capabilities of the VAX/VMS system and our computer-based education products.

Some of the features of the statistics package are:

o colorful, graphical illustrations of mathematical and statistical relationships using examples and exercises;

o databases built into the lessons which are used extensively in the examples and exercises to illustrate various distributions;

o a comprehensive glossary of statistical terms and symbols, accessible from any point within the lesson;

o an instructor-controlled symbol substitution table that gives each instructor the ability to match symbols seen on the computer to the same symbols use in the textbook of their choice;

o software that controls student access to the lesson;

o a commenting facility which allows students to communicate with the instructor via computer mail while in the lessons;

o a matrix index that allows students to access not just individual lessons, but also subsections of a lesson.

In addition to the Delaware statistics course, packages currently available from THE CLEARINGHOUSE span the broad range of applications on campus, from scientific simulations to humanities computer-based education modules.

[Mail in card:]

THE CLEARINGHOUSE FOR ACADEMIC SOFTWARE
The Computation Center
104 Computer Science Building
Iowa State University
Ames, Iowa 50011
(515) 294-1832

SELECTING A WORD PROCESSING PACKAGE

Jan Bickerstaff
Millsaps College

In the spring of 1984, Millsaps College entered into a quantity discount agreement with Digital Equipment Corporation for the purchase of Rainbow personal computers. Rather than trying to support myriad word processing software packages for the Rainbow, the Computer Services Department at Millsaps decided to select one word processing package that would be supported by the department. The purpose of this article is to describe the word processing software evaluation that was conducted as a result of that decision.

The following steps were followed in the evaluation:

1. Evaluate word processing workload
2. Determine necessary features
3. Select software packages to be evaluated
4. Perform evaluation of each package
    a. Feature check list
    b. Subjective evaluation
5. Select package, based on evaluation

1. Evaluate word processing workload

At Millsaps, it was necessary to select software that could be used by our administrative offices as well as our faculty. We characterized our workload as follows:

Academic applications

Examinations
Lecture notes
Books and articles

Administrative applications

Promotional/informational mailings
Reports
Correspondence
Contracts and briefs

Business/financial applications

Collection letters
Budgets
Financial statements

2. Determine necessary features

The computer services staff compiled lists of desirable features for each of the types of documents mentioned in step one. In addition, a subjective evaluation form was developed.

3. Select group of packages to be evaluated

The packages to be evaluated were selected on the basis of reviews in current periodicals and conversations with users at other colleges. After compiling a list of potential software for evaluation, an educational sales representative for each vendor was contacted. A free thirty-day evaluation period was arranged with the vendor and information on prices and license policies was noted at that time. The word processing packages selected for evaluation were WPS-80, FinalWord, WordPerfect, WordMARC, and Samna Word III.

4. Perform evaluation of each package

a. Feature checklist

Each package was compared with the list of features prepared in step two. If the package contained the feature a one was assigned. If not, a zero was assigned. After the features were evaluated, a composite score was assigned to each package. This feature check enabled a preliminary comparison of the packages and pointed out whether any of them was lacking several features in a particular area.

A copy of the feature checklists is included at the end of this article.

b. Subjective evaluation

This part of the evaluation was designed to assess performance and ease of use for each package. The subjective evaluation of the software was conducted by a variety of users, including, when possible, naive users.

A copy of the subjective evaluation form is included at the end of this article.

5. Select package based on evaluation

None of the packages was eliminated from our list based on the feature check. Samna Word III was eliminated, after evaluation, because of price. We, relied heavily on our subjective evaluation in making our decision. As mentioned earlier, we have a small support staff at Millsaps. This caused us to emphasize ease of use and some type of introductory tutorial, rather than execution speed and sophistication.

The package that we selected, based on our evaluation, was WordMARC. In retrospect, we believe that our evaluation was effective. This does not mean that we have received no complaints or have been completely satisfied. Several users have complained about the amount of time necessary to load the software and the speed at which it executes print operations. In future evaluations, more consideration will be given to the feel of the package to the experienced user. It's never possible to please all of the users at a site, but, we feel that at a small college, the idea of supporting only a few software packages is essential.

FEATURE CHECKLIST

Basic Features

Word/line cursor movement
Sentence/paragraph cursor movement
Screen paging(back/forward)
Auto-hyphenation
Word/line delete
Search
Search and replace
Word wrap
Directory maintenance
Right justification
Line centering
Underscore
DEC printer/NEC printer support

Advanced Features

Search, ignoring case
Search and replace backwards
Replace with query
Hyphenation with query
Auto-report of search and replace
Undelete
Block operations
Multiple active blocks
Widow/orphan control
Index generation
Table of contents generation
Automatic backup files
1.5, double and triple spacing
User defined macros
User defined glossary
Training tutorial
Reference card or key labels
Menu driven
Conditional end of page
Dual document editing

FEATURE CHECKLIST

Document Features

Letters/Memos
- Interactive/typewriter print
- Background printing
- Spooling
- Pause between pages
- Print multiple copies

Short/Medium Documents
- Header/footer
- Footnotes
- Subscript/superscript
- Automatic outline formatting
- Dynamic format adjustment
- Continuous/broken underline
- Page break display
- Jump to page
- Global search/replace
- Status line
- Spelling checker

Long Documents
- Block merge
- Partial document print
- Printer interrupt/cancel
- Unlimited document size
- Disk buffering

Business/Financial
- Horizontal scrolling
- Decimal tab
- Center tab
- Right justified tab
- Math mode
- Column insert/move

Mail-Merge Documents
- ASCII formatted data files
- "Forms" data editor
- Sort/select
- Labels

SUBJECTIVE EVALUATION FORM

This segment of the evaluation requires three documents: a two page text document, a simple columnar report, and an outline.

The purpose of this segment of the word processor evaluation is to provide a general feeling for the convenience and efficiency of the package. This evaluation neither tests all of the relevant features of a word processor; nor comprehensively evaluates the features that it does test. The objective is to determine the overall implementation philosophy of the program, including such aspects as documentation quality, command sequence organization, function key placement, menu selections, and execution efficiency.

Before You Begin

Prior to conducting the evaluation, become familiar with the package and work through any tutorial material provided. Review the reference manual, reading those parts which reinforce the points covered in the tutorial. You should also become familiar with the manual's index and table of contents for quick reference to the relevant sections.

Practice entering a few short documents and become familiar with the command sequence format and the special function keys. Test the commands and features discussed in the tutorial.

The Evaluation

You will be asked to enter and modify several documents. After each step of the procedure, rate the package's performance on two points: Learning and Efficiency.

I. Learning

"Learning" refers to the ease or difficulty with which you were able to discover the command sequences necessary to perform the operation and implement it the first time. This question does not refer to any specific aspect of the documentation or command structure. It is a measure of your feeling towards the difficulty of learning how to use the features of the package. Two different packages may have very different ways of performing the given operation an both be rated "Good".

II. Efficiency

After you have performed the operation and (if possible) practiced several methods of implementing it, rate the performance of the package for this type of operation. Assume, in making you decision, that you would have to use this feature frequently. We are attempting to determine whether the features provided by the package will be effective during daily use.

Package Name: ______________________________

Evaluator: ______________________________

1. Entry and Editing

Enter document number 1 into the using the word processor and save it. Proofread the document, practicing the cursor movement and general editing commands.

Learning: Poor ___ Fair ___ Good ___ Excellent ___

Efficiency: Poor ___ Fair ___ Good ___ Excellent ___

Comments:

2. Page Format Modification

Change the format of the document so taht it is double spaced with indented paragraphs rather than single spaced with block paragraphs. Modify the margins making them narrower.

Learning: Poor ___ Fair ___ Good ___ Excellent ___

Efficiency: Poor ___ Fair ___ Good ___ Excellent ___

Comments:

3. Special Modes

Change all words in the document enclosed in double quotations to BOLD and UNDERLINE all words enclosed in single quotations.

Learning: Poor ___ Fair ___ Good ___ Excellent ___

Efficiency: Poor ___ Fair ___ Good ___ Excellent ___

Comments:

4. Headers, Footers and Page Numbering

Put a page number centered at the bottom of the page. Insert the header, "Artificial Intelligence: Technology or Ideology?", blocked to the left on every page except the first.

Learning: Poor ___ Fair ___ Good ___ Excellent ___

Efficiency: Poor ___ Fair ___ Good ___ Excellent ___

Comments:

5. Footnotes

Insert the footnote:

> Smith, Thosmas P., The History of Artificial Intelligence. Harper and Row, 2983, pasges 202-207.

at the end of the second paragraph.

Learning: Poor ___ Fair ___ Good ___ Excellent ___

Efficiency: Poor ___ Fair ___ Good ___ Excellent ___

Comments:

6. Special Feature Editing

Change the underlining added in step three back to single quotations and remove all bolding. Delete the footnote added in step five.

Learning: Poor ___ Fair ___ Good ___ Excellent ___

Efficiency: Poor ___ Fair ___ Good ___ Excellent ___

Comments:

7. Spelling Check

Use the spelling checker of the package (if one is available) to check your document for errors. Make the necessary corrections and add any special terms to the package's dictionary.

Learning: Poor ___ Fair ___ Good ___ Excellent ___

Efficiency: Poor ___ Fair ___ Good ___ Excellent ___

Comments:

8. Financial Tables

Set up document two as a simple columnar report and enter the first lines of the table. Make the report more than eighty columns wide.

Learning: Poor ___ Fair ___ Good ___ Excellent ___

Efficiency: Poor ___ Fair ___ Good ___ Excellent ___

Comments:

9. Math mode

If the package that you are using has a Math feature, modify the table prepared above so that the column totals are calculated automatically.

Learning: Poor ___ Fair ___ Good ___ Excellent ___

Efficiency: Poor ___ Fair ___ Good ___ Excellent ___

Comments:

10. Outlines and Lists

Enter the outline provided as document three. Make your verion appear as close to the format of the original document as possible.

Learning: Poor ___ Fair ___ Good ___ Excellent ___

Efficiency: Poor ___ Fair ___ Good ___ Excellent ___

Comments:

DECUS

GRAPHICS

GRAPHICS

Chairman
William Kramer
University of Delaware
Newark, DE

Symposium Coordinator
Bijoy Misra
Smithsonian Institution
Cambridge, MA

Newsletter Editor
Michael P. Anton
Houston, TX

Associate Newsletter Editor
Charles D. Carter
Huntington Alloys, Inc.
Technology Department
Huntington, West Virginia

Workstation Working Group Coordinator
Bob McCormick
Video Communications, Inc.
Feeding Hills, MA

Engineering Graphics Working Group Coord.
Eric Rehm
Gonzaga University
SPOCAD
Spokane, WA 99258

Session Note Editor
Carol Schwob
Florida Atlantic University
Academic Computing
Boca Raton, Florida 33431

Library Coordinator
Mike McPherson
Michigan State University
East Lansing, MI

Standards Coordinator
Jim Flatten
Ames Lab
Ames, IA

Volunteer Coordinator
Dick McCurdy
University of Florida
Gainsville, FL

Library Committee
James M. Turner
Saber Technology
San Jose, CA

DEC Counterpart
Rick Berzle
Digital Equipment Corporation
Spit Brook, NH

Information Officer
Mike York
Boeing Computer Services
Seattle, WA

Human Interface Working Group Coord
Dottie Elliott
Research Triangle PK, NC

Data Display Working Group Coord.
Joy Williams
Eaton Corp.
Southfield, MI

Pre-Symposium Seminars in San Francisco

Bill Kramer

The GAPSIG has expanded its Pre-Symposia Seminar offerings to three exciting seminars for the Fall Symposium Two seminars are brand new and are being offered to the DECUS members for the first time. However both are special version of seminars which have been given across the country to a large number of groups. It is the hoped that these seminars will all be attractive. In fact, the harder time you have making up your mind, the better. Just make up your mind to attend one of them!

We try to provide seminars which speak to different interests. The first, a seminar on developing programs and applications for the VAXstation with UIS and GKS, was quite a success at the last symposium. This seminar will be updated to include new information and new products.

We are offering a high level seminar on how to create effective graphics. Elenor Matthews is an expert in using computer graphics hardware and software to create effective visual presentations. This seminar was developed at the request of some DECUS members, because the techniques which combine to make effective slides, charts and graphs is difficult to research. Having the proper tools (the hardware and software) is one thing, knowing how to use the tools effectively and efficiently is another This seminar is designed to provide the guidance which will allow the attendees to effectively use their graphics resources.

The third seminar, also new to DECUS corresponds to the GAPSIG Symposium theme Graphics Hardcopy" It is aimed at people interested in understanding the details of laser printers As you already realize, these devices are revolutionizing the graphics hardcopy world because the are fast, inexpensive, and extremely flexible This seminar will expose the inner logic of the Postscript family of printers. It will help people who are writing their own interfaces, or trying to install and understand laser printers. While the seminar deals specifically with the LN family of Digital printers, Postscript is a vendor independent interface, which is available on many laser printer products, so the seminar should be of general interest.

Detailed reviews of the seminars follow. We encourage you to register for one of these seminars or one of the other sixty-nine seminars in the registration book. You will find the Pre-Symposium Seminars to add tremendously to the DECUS experience.

GRAPHICS ON THE VAXSTATION USING GKS AND UIS

This seminar explores in detail the two major graphics interfaces on the VAXstation, GKS and UIS. GKS is an implementation of the GKS graphics standard, and is a tool designed to create sophisticated device independent computer graphics application programs. UIS is a lower level VAXstation specific interface which allows complete control of the VAXstation graphics abilities. UIS allows device specific operations such as window control and pixel operations

This seminar will entail a detailed explanation of GKS and UIS The two interfaces will be compared and an explanation of when to use each will be given Then, the details of programming using GKS or UIS will be presented How to create graphics using both systems will be explained and many examples will be provided

The overall intent of this seminar is to explain effective methods of creating computer graphics on the VAXstation using two powerful interface tools The application of this seminar spans all ranges of graphics and human interface design

Presenters Jake VanNov is a principal software engineer in the VMS Development Group who wrote the viewport/window management layer of the workstation software for the VS/I and the VS/II Since joining Digital, Jake has worked a range of VMS projects. Currently Jake is a project leader working on enhancements to the workstation programming interface and applications for the VAXstation. Jake received his B.S. and M.S. in Computer Science from the University of Pittsburgh. He will be presenting the UIS section of the seminar.

Garry Poegel is a member of the VAX GKS Development Group, working on the next version of VAX GKS Garry graduated from the University of New Hampshire in Computer Science.

Attendees should be familiar with the basic concepts of interactive graphics and with VMS. Anyone who is interested in using the VAXstations to create graphics - either simple or complex - should benefit from attending this class.

PRESENTATION GRAPHICS

The ultimate goal of presentation graphics is to communicate ideas clearly and accurately with precise and well-thought pictures. It is possible to make graphics design decisions that will help an audience grasp complex or abstract information, facts, plans, processes and statistics. The process of creating effective graphics with automated devices is a step-by-step progression of carefully considered decisions, such that the quality of the final product reflects the care given to each of these decisions. This seminar, divided into four sessions, will focus on different aspects of decision-making while working on presentation graphics. The first session will be devoted to the elements of graphics and will cover aspects of color, size, ruling, frames and borders. The second session on information graphics will cover topics on the production of graphs, maps and illustrations. The third session will be devoted to the design aspects of the graphics relating to media, clarity and credibility. Finally, the last session will discuss the common problem areas and "don't do's".

Attendees of this tutorial seminar will learn practical and effective methods of visual communication. Emphasis will be on working within the constraints of automated graphics devices, including color limitation and degradation, image transfer, resolution limitations, presentation media and systems compatibility.

Presenter: Eleanor Mathews has a B.A in Geography/Automated Cartography from the University of Washington and a Fifth Year in Graphic Design from the same school. She has designed information graphics for books, film, television and court-room presentations. Her graphics design experience includes book design signage, identity development and a variety of marketing literature. Ms. Mathews has taught Graphics Design, Elementary Statistics Statistical Illustration and Principles of Computer Graphics She has designed interfaces for illustration graphing and mapping software. She has been a Program Manager with Tektronix Inc and has directed the Presentation Graphics Software Group Currently, she works as a free-lance consultant based in Seattle area.

Who should attend: Anyone with an interest in learning about Presentation Graphics.

GENERATING POSTSCRIPT DOCUMENTS

For high quality text and graphics applications, the Postscript page description language is coming into wide use as the interface to laser printers. Are you developing a software application that writes output pages in the form of Postscript files? This seminar will recommend many techniques to help you make the best use of available printing resources.

Improper use of PostScript can cause applications to suffer in function or performance Whether your application is a text formatter, a WYSIWYG editor, a graphics package or a CAD system, speed and accuracy of the printed output is important to you.

Topics: A model PostScript driver is discussed, including

- o **what is ideal PostScript**
- o **structuring conventions, modularity and specifics for DEC printers**
- o **how to use font metrics**

An overview of the PostScript Interpreter, including

- o **the graphics image model**
- o **data-types and stacks**
- o **the font cache and virtual memory**

Advanced uses of PostScript

- o **dictionaries and name scoping**
- o **the saving and restoring mechanism**
- o **changing and preserving coordinate transforms**
- o **digitized and synthesized images**
- o **halftoning**
- o **font management**
- o **customization**
- o **variations on the show verb**
- o **debugging aids and techniques**

Presenter: Matt Foley is a customer support engineer for Adobe Systems, the company that invented PostScript. Matt developed the Advanced PostScript course for Adobe. He consults with manufacturers, OEMs and third party software developers to help develop and optimize PostScript drivers. Matt has a BS and MA from Stanford. He will be assisted in the seminar by a Digital Developer involved with developing PostScript drivers and translators.

Who Should attend: This seminar is intended for programmers who are actively involved in writing applications that emit PostScript.

GKS Working Group Report

Jim Flatten

There is not a great deal to report on the activities of the GKS Working Group. At this time no one has successfully moved the GKS system to any other operating system. I am working (ever so slowly) on the RT-11 conversion. Others are working on conversions to VMS and RSX.

Mike McPherson our tape librarian, has submitted the GKS tape (ULTRIX version) to the DECUS library It didn't make the latest catalogue, but it is available we will publish ordering information next month

There will be a session in San Francisco on the activities of the GKS working group. The session will be very informal and if you want to find out more about what this working group is doing, or if you want to participate, please attend this session. If you can't get to San Francisco, please write to me:

Jim Flatten
2581 Metals Development
Ames Laboratory
Ames, IA 50011

GAPSIG Q & A

Abstracted from an ARPAnet discussion on "X", a windowing system for Ultrix.

Does anyone have any hard figures on the CPU overhead of running "X" on a VAXstation II which also runs competing timesharing applications? I've been playing with one, and with one or two additional jobs running on other terminals, I can see the timeslicing during window-scrolling. It's a new experience getting swapped out in the middle of drawing a character or re-drawing half a line!

The VS-2 and VS-2/RC (VAXstation II) uses the QVSS display, which is memory on the Q-Bus. There is NO hardware assist When scrolling or painting text, the CPU is compute bound while it manipulates the display memory. X is a user level process, and so will be timeshared like any other process. This means that in a timeshared environment, the window system will slow down in direct proportion to the timesharing (causing the pauses in the middle of character painting when some other process is running). If you are actually swapping, you need more memory in your machine.

If you want to give preference to the user of the display, and don't care about penalizing other users on the machine, you can simply raise the priority on the window system server. The moral here is that the VS-2 is intended to be used as a single user workstation.

Displays with outboard processing power (for example, the DEC VS100 on unibus Vaxes) do much better in a timeshared environment. We have run X on as many as four of them on a single VAX 11/750 without much problem. A VAXstation II/GPX should also perform better since it has a separate display processor. (Jim Gettys – Digital Equipment Corporation – MIT Project Athena)

Questions? Send them to:

Michael Anton
GAPSIG Q & A
P.O. Box 591293
Houston, Texas 77259-1293

More SIGGRAPH Tapes

Bill Kramer

The GAPSIG SIGGRAPH tape library now contains four new video tapes of the latest computer graphics research and developments which were shown at the 1985 ACM/SIGGRAPH conference.

We show these and other tapes during symposia: in sessions, in the GAPSIG campground and in the suite. They are great fun and exciting as well as educational. So if you're in San Francisco next month, stop by one of the shows to relax and to see some of the very best in computer graphics.

San Francisco Symposium!

Bijoy Misra

The Graphics Applications SIG welcomes you to a potpourri of sessions and activities in San Francisco! You will learn about new advances on workstations, graphics software programming, design and implementation of human interface methods, and other miscellaneous topics including engineering graphics displays, image analysis and presentation graphics. The theme for the Graphics SIG at the Symposium is "Graphics Hardcopy Systems" and a full day of sessions on the theme is planned by invited speakers who are recognized experts in the field Besides the Symposium offerings the SIG is also sponsoring three pre-symposium seminars on Postscript, GKS and Presentation Graphics The campground will give you a chance to use new equipment and get "hands-on" demonstrations from the developers themselves

Monday will be devoted to sessions announcing new products on the VAXstation, including software design and implementation of the programming interface. Other topics, include tutorial sessions on VAXstation internals and windowing software using UIS – the User Interface System. Tuesday is the theme day for the SIG, and we run all day sessions related to "Hardcopy Systems". The keynote speaker is Jim Warner from the Precision Visuals. Mr. Warner will present a review of the hardcopy system technology and give an overview of future trends. The follow-up sessions in the day cover tutorials on various hardcopy related topics including Postscript, ink-jet printing, graphics standards and laser printers.

Wednesday's sessions are devoted to programming in GKS and design of human interface methods. The GKS sessions comprise of tutorials on advanced graphics concepts, GKS programming and design of GKS device- handlers. There are sessions on using VAX GKS with plenty of examples. The human interface sessions will address display technology, graphics design, screen management and review of interface methods.

Thursday has a variety of sessions concentrating on graphics standards, use of local segments, graphics software, image analysis and presentation graphics. Several sessions relating to engineering graphics, on CAD/CAM, IGES, and AUTOLISP are also scheduled on Thursday.

On Friday, we will wrap up with sessions on low-end VAXstations, VAXstation network management graphics terminals and software libraries.

Computer graphics video-tapes will be shown in the campground all week, and demos and clinics will be held on the hardware and software discussed in the sessions.

The graphics activity at San Francisco will be more exciting than ever! We hope you can join us in celebrating the SIG's activities!

Graphics Standards in San Francisco

Jim Flatten

There will be two sessions at the San Francisco DECUS Symposium on the topic of computer graphics standards. One session will be about the Computer Graphics Metafile (CGM) and the other will be on the Programmers Hierarchical Interactive Graphics System (PHIGS)

The CGM is the second standard to be completed by X3H3, the ANSI technical committee on computer graphics. CGM will be published as an ANSI standard late this year The session will address the details of CGM It will also show how the CGM relates to other proposed and adopted graphics standards like GKS, PHIGS and the Computer Graphics Interface (CGI). CGM will be important to other standards like MAP/TOP IGES/PDES and Composite Document Standards. I will present this session.

PHIGS is a proposed viewing standard that is designed to be highly interactive and which supports a three dimensional hierarchical graphics database. PHIGS allows editing of the graphical database. It is an important standard to applications requiring powerful, interactive viewing software, such as CAD/CAM, molecular modeling and other highly interactive applications. The GAPSIG is sponsoring a session on PHIGS in San Francisco. The speaker will be Brian Axtell of DEC one of the VMS software engineers. This session should be a good place to learn about PHIGS and also to get an idea of how DEC views this graphics standard

I will have many of the standards documents available at the Symposium – in the GAPSIG Suite and the campground area. If you have questions about graphics standards, or if you want to express your point of view to ANSI, please look me up. I am always looking for input from the DECUS membership on standards issues. You may also write to me at this address:

Jim Flatten
2581 Metals Development
Ames Laboratory
Ames, IA 50011

HMS

HMS

HMS

Chairman VAX SIG Liaison
Thomas J. Provost
MIT/LNS Bates Linac Facility
Middletown, MA

Product Planning Coordinator
George Hamma
Synergistic Technology
Cupertino, CA

Symposium Seminar Coordinator
Mike Allen
Lawrence Livermore National Labs
Livermore, CA

Communications Coordinator
John G. Hayes
Information Systems - S. Central Bell
Birmingham, AL

Publications Coordinator (Editor)
Bill K. Walker
Monsanto Research Corp.
Miamisburg, OH

Session Notes
DAARC SIG Liaison
Bill Tippie
Kinetic Systems Corp.
Lockport, IL

Standards Coordinator
CAMAC Working Group Coordinator
Peter Clout
Los Alamos National Lab
Los Alamos, NM

LUG Coordinator
Gregg Giesler
Los Alamos Science Lab
Los Alamos, NM

Pre-Symposium Seminar Coordinator
Mike Allen
Lawrence Livermore National Labs
Livermore, CA

TOEM (Chips % Boards)
Jack J. Peterson
Horizon Data Systems
Richmond, VA

HHK (Hardware Hints & Kinks)
Wayne Kesling
Monsanto Research Corp.
Miamisburg, OH

UNIBUS Hardware
Ron Bogue
LIV Aerospace & Defense Co.
Dallas, TX

Performance Measurement Coordinator
William Wallace
600 W. Washington St.
Peoria, IL

CAMAC Coordinator
Peter Clout
Los Alamos National Lab
Los Alamos, NM

CSS Coordinator
Pratap Gohel
E.I. Dupont
Ingleside, TX

Networks SIG Liaison
Sandra Traylor
Target Systems
Yorba Linda, CA

VAX SIG Liaison
Dave Schmidt
5100 Centre Avenue
Pittsburgh, PA

DAARC SIG Liaison
Bill Tippie
Kinetic Systems Corp.
Lockport, IL

UNISIG SIG Liaison
Jim Livingston
1 Results Way
Cupertino, CA

SITE SIG Liaison
Emily Kitchen
A.H. Robbins Co.
Richmond, VA

RT-11 SIG Liaison
Gary Sallee
Sallee Software Consulting
Yorba Linda, CA

RSX SIG Liaison
Hans Jung
Associated Press
New York, NY

Members-At-Large
Mike Rembis
American Dade
Costa Mesa, CA

Hans Dahlke
Richland, WA

Jim Cutler
EDS Tower, 26533 Evergreen
Southfield, MI

DEC Counterparts
Terminals
Nina Abramson
Digital Equipment Corporation
Maynard, MA

TOEM (Chips & Boards)
Art Bigler
Digital Equipment Corporation
Marlboro, MA

Diagnostic
George D. Cooke
Digital Equipment Corporation
Maynard, MA

Storage
Marilyn Fedele
Digital Equipment Corporation
Maynard, MA

MSD (Micro Systems Development)
Roy Rodgers
Digital Equipment Corporation
Maynard, MA

Printer Products
Frank Orlando
Digital Equipment Corporation
Maynard, MA

DECUS Europe Liaison
Hans Zoller

THE DEVIAS LETTER

IAS

IAS SIG Steering Committee

Chairman
John Roman
McDonnell Douglas
Department N436
Hazelwood, Missouri 63042
(314) 234-0984

Newsletter Editor
Frank R. Borger
Radiation Therapy
Michael Reese Medical Center
Lake Shore Drive @ 31st St
Chicago, IL 60616

WHIMS Coordinator
Kathleen Anderson
Eaton Information Management
System Division
Hampton, VA

RSX Liaison
Ray French
Boeing Computer Services
Seattle WA

Member-at-Large
Doug Reno
Abbot Laboratories
North Chicago, IL

DEC Counterpart
Mike Reilly
Digital Equipment Corp.
Maynard, MA

Chairman Emeritus
Bob Curley
Division of Medical Physics
University of Pennsylvania
Philadelphia PA

Symposium Coordinator
Mike Robitaille
Grumman - CTEC, Inc.
McLean, VA

Library Coordinator
Bob Schuldt
INCO INC.
McLean, VA

Member-at-Large
Kerry Wyckoff
Salt Lake City, UT

DEC Counterpart
Tim Leisman
Digital Equipment Corp.
Stow, MA

DEC Counterpart
Bob Mack
Digital Equipment Corp.
Landover, MD

CONTRIBUTION GUIDELINES

Contributions for the newsletter can be sent to either of the following addresses:

Editor, IAS SIG Newsletter
DECUS U.S. Chapter
219 Boston Post Road, BP02
Marlboro, MA 01752

Frank R. Borger
Michael Reese Medical Center
Department of Radiation Therapy
Lake Shore Drive at 31st St
Chicago, IL 60616

Contributions of letters, articles, important SPR's etc will be accepted in any form, (including notes jotted in pencil on gravy-stained tablecloths.) Contributions will be much more graciously accepted in one of the following formats:

1. Non machine readable sources, (SPR's etc,) should be reasonably dark to insure good photocopying. Text whatever should be the equivalent of 66 lines at 6 lpi, with 4-line top margin, 5-line bottom margin, left-margin 10, right margin 74 at 10cpi.

2. Machine readable sources may be submitted on 9-track Mag-tape, (800,1600, or 6250 BPI,) DEC-tape II, DecMate floppies, or whatever. We're not fussy, we'll even accept paper tape or cards. Preferred format is DOS or BRU for tapes, Files-11 for DEC-tape II.

3. 1200 baud dial-up modems are available on our IAS system and our VAX, with various servers available. Give the editor a call at (312)-791-2515 (preferably later in the day,) to obtain access information, etc.

4. If long distance dialout is not possible on your system, we'll be willing to call your system and do the work, (unless you want to transfer the entire manual set at 300 baud.)

```
             ___      From the Editor's Terminal        ___
            /   \                                      /   \
           / o o \ All the News That Fits, we print   / o o \
          (_\   /_)                                  (_\   /_)
            |\@/  \__ _                       _    __/  \@/|
      IAS   | U    \  \\                      \\  / /    U |
            \      /\ '))--Network--|o o o o|---\\' /\      /
  _______    \\  //  \//            |       |   \/   \\  //
 /Mr Spot\  _//_// _| /      Mr Vax |       |     \ |_ \\_\\_
(_________)(_/(_/ (__/          VMS |_______| Rover\__) \_)\_)
```

The summer doldrums are definitely here. I find myself loath to do anything more than I have to. This languor must be affecting other members of the SIG, since contributions have been less than overwhelming. As usual, a couple of stalwarts have been doing more than their share. How about the rest of you? Send us a really good contribution, and get a collector's item, an IAS pen.

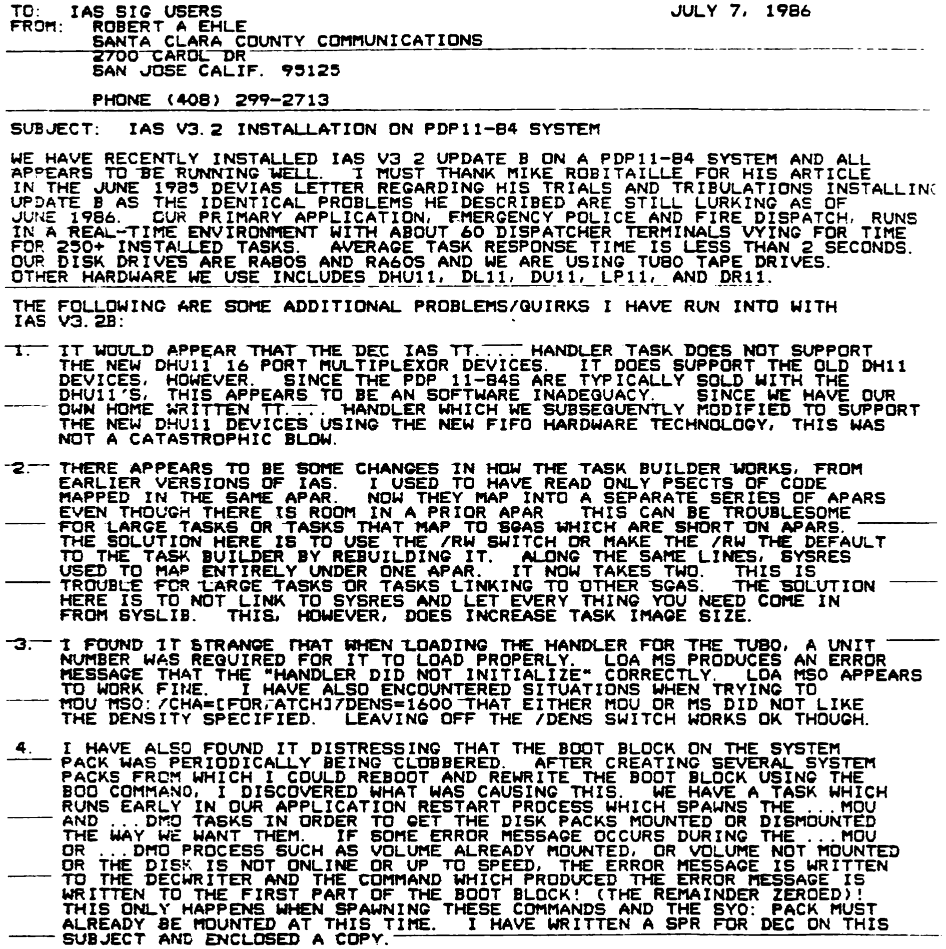
Thanks to Robert Ehle of Santa Clara County Communications for his feedback. Some problems never go away. (The first time I heard of boot blocks getting clobbered it was RSX11D version 4, the clobberer was the card reader handler. with a "Card reader not ready" message.) Don't complain too hard Bob, every time it happened to CY Mead, he had to do a full SYSGEN, since he was running a 1 disk system. (He did his sysgens using a batch stream from the card reader.) A lot of people still write terminal IO using IO.WLB instead of IO.WVB. DEC should explain the difference better in the manuals.

10 Years Ago This Month

The Multi-tasker contained mostly a series of SPR's for RSX11D (version 6.B) Among the more notable problems:

1. If the system saw memory parity errors during a DMA disk write, it assumed that the current task was the culprit. This lead to the system trying to terminate the NULL TASK, and other good things.

2. A user was surprised that although PREserve created bootable backup tapes, (containing a simple 11S system, useful for restoring your system disk on a 1 disk system,) Standalone PREserve, (the one you were supposed to use to back up system disks because of checkpoint files, etc,) only put a Dummy system on the tape that just said, "This volumn does not contain a bootable system." It was wonderful things like PREserve that lead many users to write their own backup/restore programs.



TO: IAS SIG USERS   JULY 7, 1986
FROM: ROBERT A EHLE
SANTA CLARA COUNTY COMMUNICATIONS
2700 CAROL DR
SAN JOSE CALIF. 95125

PHONE (408) 299-2713

SUBJECT: IAS V3.2 INSTALLATION ON PDP11-84 SYSTEM

WE HAVE RECENTLY INSTALLED IAS V3 2 UPDATE B ON A PDP11-84 SYSTEM AND ALL APPEARS TO BE RUNNING WELL. I MUST THANK MIKE ROBITAILLE FOR HIS ARTICLE IN THE JUNE 1985 DEVIAS LETTER REGARDING HIS TRIALS AND TRIBULATIONS INSTALLING UPDATE B AS THE IDENTICAL PROBLEMS HE DESCRIBED ARE STILL LURKING AS OF JUNE 1986. OUR PRIMARY APPLICATION, EMERGENCY POLICE AND FIRE DISPATCH, RUNS IN A REAL-TIME ENVIRONMENT WITH ABOUT 60 DISPATCHER TERMINALS VYING FOR TIME FOR 250+ INSTALLED TASKS. AVERAGE TASK RESPONSE TIME IS LESS THAN 2 SECONDS. OUR DISK DRIVES ARE RA80S AND RA60S AND WE ARE USING TU80 TAPE DRIVES. OTHER HARDWARE WE USE INCLUDES DHU11, DL11, DU11, LP11, AND DR11.

THE FOLLOWING ARE SOME ADDITIONAL PROBLEMS/QUIRKS I HAVE RUN INTO WITH IAS V3.2B:

1. IT WOULD APPEAR THAT THE DEC IAS TT.... HANDLER TASK DOES NOT SUPPORT THE NEW DHU11 16 PORT MULTIPLEXOR DEVICES. IT DOES SUPPORT THE OLD DH11 DEVICES, HOWEVER. SINCE THE PDP 11-84S ARE TYPICALLY SOLD WITH THE DHU11'S, THIS APPEARS TO BE AN SOFTWARE INADEQUACY. SINCE WE HAVE OUR OWN HOME WRITTEN TT.... HANDLER WHICH WE SUBSEQUENTLY MODIFIED TO SUPPORT THE NEW DHU11 DEVICES USING THE NEW FIFO HARDWARE TECHNOLOGY, THIS WAS NOT A CATASTROPHIC BLOW.

2. THERE APPEARS TO BE SOME CHANGES IN HOW THE TASK BUILDER WORKS, FROM EARLIER VERSIONS OF IAS. I USED TO HAVE READ ONLY PSECTS OF CODE MAPPED IN THE SAME APAR. NOW THEY MAP INTO A SEPARATE SERIES OF APARS EVEN THOUGH THERE IS ROOM IN A PRIOR APAR THIS CAN BE TROUBLESOME FOR LARGE TASKS OR TASKS THAT MAP TO SGAS WHICH ARE SHORT ON APARS. THE SOLUTION HERE IS TO USE THE /RW SWITCH OR MAKE THE /RW THE DEFAULT TO THE TASK BUILDER BY REBUILDING IT. ALONG THE SAME LINES, SYSRES USED TO MAP ENTIRELY UNDER ONE APAR. IT NOW TAKES TWO. THIS IS TROUBLE FOR LARGE TASKS OR TASKS LINKING TO OTHER SGAS. THE SOLUTION HERE IS TO NOT LINK TO SYSRES AND LET EVERY THING YOU NEED COME IN FROM SYSLIB. THIS, HOWEVER, DOES INCREASE TASK IMAGE SIZE.

3. I FOUND IT STRANGE THAT WHEN LOADING THE HANDLER FOR THE TU80, A UNIT NUMBER WAS REQUIRED FOR IT TO LOAD PROPERLY. LOA MS PRODUCES AN ERROR MESSAGE THAT THE "HANDLER DID NOT INITIALIZE" CORRECTLY. LOA MS0 APPEARS TO WORK FINE. I HAVE ALSO ENCOUNTERED SITUATIONS WHEN TRYING TO MOU MS0:/CHA=[FOR,ATCH]/DENS=1600 THAT EITHER MOU OR MS DID NOT LIKE THE DENSITY SPECIFIED. LEAVING OFF THE /DENS SWITCH WORKS OK THOUGH.

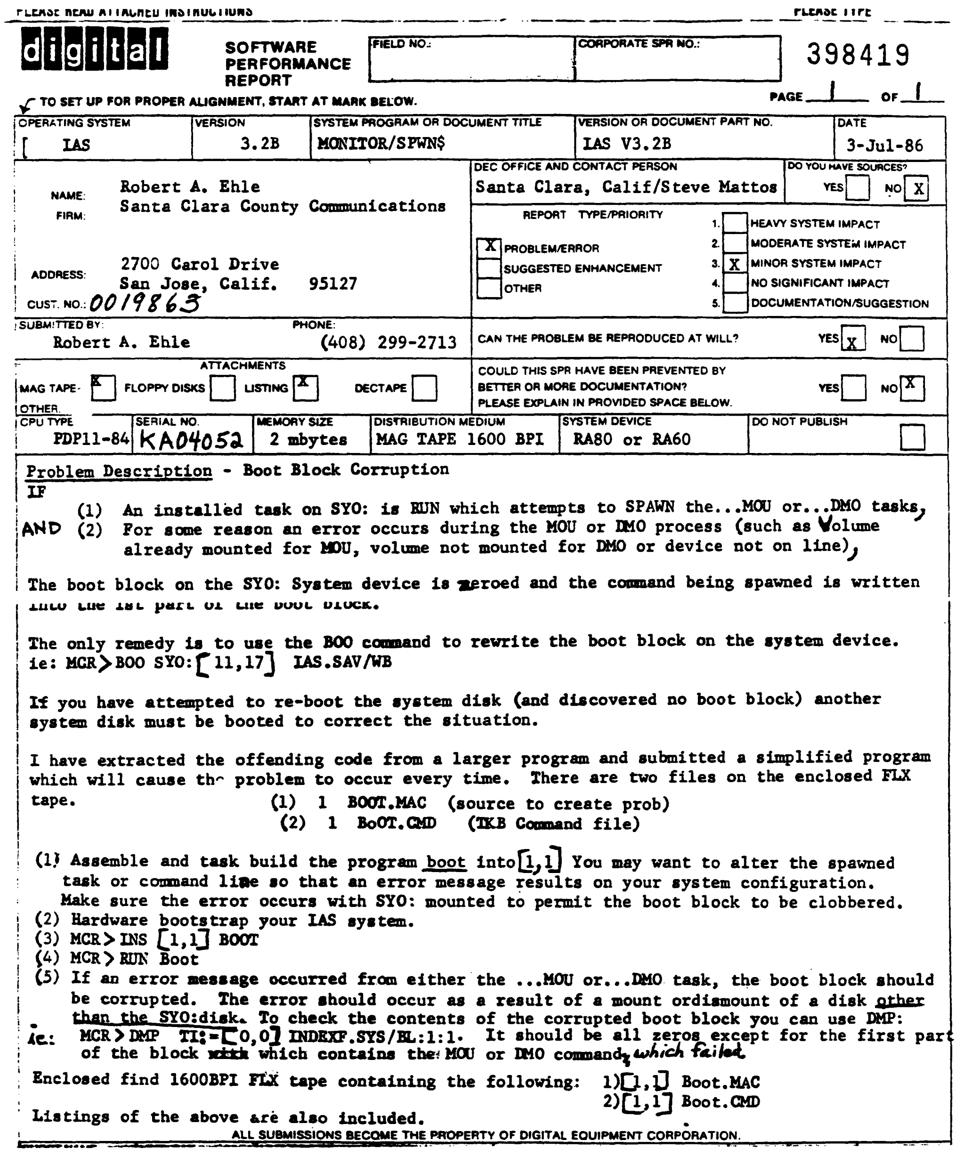
4. I HAVE ALSO FOUND IT DISTRESSING THAT THE BOOT BLOCK ON THE SYSTEM PACK WAS PERIODICALLY BEING CLOBBERED. AFTER CREATING SEVERAL SYSTEM PACKS FROM WHICH I COULD REBOOT AND REWRITE THE BOOT BLOCK USING THE BOO COMMAND, I DISCOVERED WHAT WAS CAUSING THIS. WE HAVE A TASK WHICH RUNS EARLY IN OUR APPLICATION RESTART PROCESS WHICH SPAWNS THE ...MOU AND ...DMO TASKS IN ORDER TO GET THE DISK PACKS MOUNTED OR DISMOUNTED THE WAY WE WANT THEM. IF SOME ERROR MESSAGE OCCURS DURING THE ...MOU OR ...DMO PROCESS SUCH AS VOLUME ALREADY MOUNTED, OR VOLUME NOT MOUNTED OR THE DISK IS NOT ONLINE OR UP TO SPEED, THE ERROR MESSAGE IS WRITTEN TO THE DECWRITER AND THE COMMAND WHICH PRODUCED THE ERROR MESSAGE IS WRITTEN TO THE FIRST PART OF THE BOOT BLOCK! (THE REMAINDER ZEROED)! THIS ONLY HAPPENS WHEN SPAWNING THESE COMMANDS AND THE SY0: PACK MUST ALREADY BE MOUNTED AT THIS TIME. I HAVE WRITTEN A SPR FOR DEC ON THIS SUBJECT AND ENCLOSED A COPY.

5. THE PDP11-84 ENVIRONMENT WITH RA80/RA60 TYPE DRIVES, PERMITS YOU TO BOOT A SYSTEM PACK FROM ANY DRIVE, DU0, DU1, DU2, ETC AS LONG AS YOU HAVE THAT DEVICE SYSGENED INTO THE SAVED SYSTEM. WHEN THIS HAPPENS, THE STD IS APPARANTLY UPDATED IN MEMORY TO SHOW THE TASKS INSTALLED FROM THE APPROPRIATE DEVICE NUMBER. SOMEHOW BETWEEN SYSGEN TIME AND PLAYING WITH THE SYSTEM, SAVING IT SEVERAL TIMES, I ENDED UP WITH ALL TASKS SAVED ON DU0 EXCEPT THE NEW TASK BBR... (NEW BAD BLOCK HANDLER TASK WHICH WAS SAVED ON DU1. I THOUGHT I COULD FIX THIS BY REMOVING BBR AND RE-INSTALLING. BBR... HOWEVER, IS ALWAYS RUNNING AND MUST BE ABORTED BEFORE IT CAN BE RE-INSTALLED. ABORTING BBR... HOWEVER, KILLS THE SYSTEM (AS FAR AS FURTHER COMMUNICATIONS IS CONCERNED). THE ONLY WAY I COULD CORRECT THIS SITUATION WAS TO RE-SYSGEN.

6. WILL BRU EVER WORK RIGHT FOR LARGE MULTI-VOLUME SYSTEM BACKUPS? MAYBE I JUST HAVE A LOT OF BAD TAPES. ALSO, IT APPEARS THAT DEC HAS MADE SOME EFFORTS TO CONSOLIDATE THE RSX11M+ BRU WITH THE IAS VERSION. THE DOCUMENTATION PROVIDED WITH IAS V3.2 IS LOADED WITH REFERENCES TO RSX11M+. IT SURE SEEMS DIFFICULT TO FIND THE RIGHT COMBINATION OF SWITCHES TO GET BRU GOING. I'D BE INTERESTED IF ANYONE HAS EVER GOT MULTIVOLUME BRU DATA SETS TO ROLL OUT TO TAPE AND BACK TO DISK WITHOUT SOME SORT OF I/O ERRORS WHILE USING THE VERIFY SWITCH.

7. THE USE OF THE SENPAR PARTITION TO GAIN ADDITIONAL NODE SPACE FOR SYSTEM DIRECTIVES, SEND REQUESTS, ETC., APPEARS TO BE A STEP IN THE RIGHT DIRECTION FOR OUR APPLICATION. IT WOULD BE NICE IF THERE WERE A CONVENIENT WAY FOR MONITORING THE NODE USAGE IN SENPAR.

AS FAR AS PERFORMANCE IS CONCERNED, WE DO NOT HAVE THAT MUCH TESTING TO REPORT ON. WE USED TO RUN ON THE PDP11-45 WITH CDC9762 (LARGE RP03) TYPE DISK DRIVES. THINGS APPEAR TO BE RUNNING NEARLY TWICE AS FAST, BUT MOST OF OUR MEASUREMENTS ARE EMPIRICALLY DERIVED AT THIS TIME.



PLEASE READ ATTACHED INSTRUCTIONS

PLEASE TYPE

digital SOFTWARE PERFORMANCE REPORT

FIELD NO.:

CORPORATE SPR NO.:

398419

↙ TO SET UP FOR PROPER ALIGNMENT, START AT MARK BELOW.

PAGE 1 OF 1

OPERATING SYSTEM IAS	VERSION 3.2B	SYSTEM PROGRAM OR DOCUMENT TITLE MONITOR/SPWN$	VERSION OR DOCUMENT PART NO. IAS V3.2B	DATE 3-Jul-86

NAME: Robert A. Ehle
FIRM: Santa Clara County Communications
ADDRESS: 2700 Carol Drive
San Jose, Calif. 95127
CUST. NO.: 0019863

DEC OFFICE AND CONTACT PERSON: Santa Clara, Calif/Steve Mattos

DO YOU HAVE SOURCES? YES □ NO ☒

REPORT TYPE/PRIORITY

☒ PROBLEM/ERROR
□ SUGGESTED ENHANCEMENT
□ OTHER

1. □ HEAVY SYSTEM IMPACT
2. □ MODERATE SYSTEM IMPACT
3. ☒ MINOR SYSTEM IMPACT
4. □ NO SIGNIFICANT IMPACT
5. □ DOCUMENTATION/SUGGESTION

SUBMITTED BY: Robert A. Ehle PHONE: (408) 299-2713

CAN THE PROBLEM BE REPRODUCED AT WILL? YES ☒ NO □

ATTACHMENTS
MAG TAPE ☒ FLOPPY DISKS □ LISTING ☒ DECTAPE □
OTHER

COULD THIS SPR HAVE BEEN PREVENTED BY BETTER OR MORE DOCUMENTATION? PLEASE EXPLAIN IN PROVIDED SPACE BELOW. YES □ NO ☒

CPU TYPE PDP11-84	SERIAL NO. KA04052	MEMORY SIZE 2 mbytes	DISTRIBUTION MEDIUM MAG TAPE 1600 BPI	SYSTEM DEVICE RA80 or RA60	DO NOT PUBLISH □

Problem Description - Boot Block Corruption

IF
(1) An installed task on SY0: is RUN which attempts to SPAWN the...MOU or...DMO tasks,
AND (2) For some reason an error occurs during the MOU or DMO process (such as Volume already mounted for MOU, volume not mounted for DMO or device not on line),

The boot block on the SY0: System device is zeroed and the command being spawned is written into the 1st part of the boot block.

The only remedy is to use the BOO command to rewrite the boot block on the system device.
ie: MCR>BOO SY0:[11,17] IAS.SAV/WB

If you have attempted to re-boot the system disk (and discovered no boot block) another system disk must be booted to correct the situation.

I have extracted the offending code from a larger program and submitted a simplified program which will cause th^ problem to occur every time. There are two files on the enclosed FLX tape.
(1) 1 BOOT.MAC (source to create prob)
(2) 1 BoOT.CMD (TKB Command file)

(1) Assemble and task build the program boot into [1,1] You may want to alter the spawned task or command line so that an error message results on your system configuration. Make sure the error occurs with SY0: mounted to permit the boot block to be clobbered.
(2) Hardware bootstrap your IAS system.
(3) MCR>INS [1,1] BOOT
(4) MCR>RUN Boot
(5) If an error message occurred from either the ...MOU or...DMO task, the boot block should be corrupted. The error should occur as a result of a mount ordismount of a disk other than the SY0:disk. To check the contents of the corrupted boot block you can use DMP:
ie: MCR>DMP TI:=[0,0] INDEXF.SYS/BL:1:1. It should be all zeros except for the first part of the block ~~with~~ which contains the MOU or DMO command, which failed.

Enclosed find 1600BPI FLX tape containing the following: 1)[1,1] Boot.MAC
2)[1,1] Boot.CMD

Listings of the above are also included.

ALL SUBMISSIONS BECOME THE PROPERTY OF DIGITAL EQUIPMENT CORPORATION.

Macro-16 to Macro-32
A Tale of two Systems

Frank R. Borger
Michael Reese Medical Center

In our second installment of this series, I think it is time to outline the basic differences between the two machines, and hence the machine language program. The main differences center around the expanded data types available. (Since I am discussing transporting a working machine language program, I will not discuss use of the really new data structures. Obviously decimal string and ascii string instructions, etc. sould be studied before one starts writing new programs, but they are of less use in converting an existing program.) This expansion makes itself felt in three ways.

For Integer arithmetic.

PDP11's only support BYTE and WORD variables, (and at times use two concatenated registers for multiply and divided instructions,) the VAXes support BYTE, WORD, LONGWORD (32-bits), QUADWORD (64-bits) and OCTAWORD (128 bits.) In reality, since the VAX registers are 32 bits long rather than 16, one usually will deal with BYTE, WORD and LONGWORD. The QUADWORD and OCTAWORD usually only appear in more esoteric instructions or VMS system calls, one usually can ignore them for now.

The real problem one will have to face is that you must decide if you want to convert your program to run in 32-bit mode, or keep it running in 16 bit mode. The decision must be made based upon several factors, among them:

1. Addresses must be handled in 32-bit mode. This causes several problems.

    1. If your code does a lot of work involving tables of addresses, math must take into account the increased length of addresses.

    2. JSR's of course, take twice as much stack space to store the return address. Any code that uses the stack to pass parameters to subroutines or store parameters must be carefully re- worked to get the stack offsets adjusted properly.

    3. Code like the following

        ```
        MOVL    #145,R3          or       MOVB    #145,r3
        ```

        Ends up taking up different amounts of storage on the VAX. On an 11, the constant takes 1 word to store, no matter wether you used a MOV or a MOVB instruction. On VAXes, the constant takes 1 byte of storage for a BYTE variable, 2 for a WORD variable, or 4 for a LONG variable. This accordion effect on the length of your code can have one serious consequence. Any conditional branch instructions, (which are still limited to a byte offset) may no longer work, due to the expanded code storage requirements. (This is one good reason to not re-write everything to use LONG instructions and data.)

2. On the PDP11, if one does a MOVB to a register, the sign bit is extended. On VAXes, the rest of the register is left with its previous value. VAXes also have a set of CVT instructions, that convert BYTE, WORD or LONG to any of the other values, and a MOVZ instruction, which does a MOVE Zero-extended Byte to Word, Byte to Long, or Word to Long. Unfortunately, VAXes do NOT have a MOV sign extend instruction. The result is that you could change all MOV or MOVB code to say CVTBL, CVTBW, CVTWL, and it would work, but it would not be good code, and I pity anyone having to maintain that software later.

3. The odd address limitation that exists on the PDP11 is gone on the VAX. This means that the .EVEN directive is no longer needed after an ASCII string containing an odd number of bytes. Nicely enough, the VAX does support the .EVEN directive, (and also a .ODD directive.) Some example translations follow.

    ```
    PDP                        VAX

    movb    R2,R3              cvtbl   R2,R3

    movb    (R2),R3            movz    (R2),R3
    bic     #177400,R3
    ```

Use the above to decide between 16 0r 32 bit mode. (We initally chose to do Reese Basic in 16-bit mode.)

If your programs have a lot of data areas, and most of the data is .WORD data now, it probably makes sense to stay in 16 bit mode. For large data areas, going to double sized variables makes your program twice as big, and even VAXes tend to run out of room. Its probably better to set your translator program to change all MOV instructions to say MOVW, etc. You will have to go in and change and storage areas that end up holding address pointers rather than data to be .LONG rather than .WORD. Also, code that uses a 16-bit variable as an index into a data table will have to be re-written to take that initial 16-bit offset and convert it to a 32 bit offset before adding in the table base. Problems like this will occur throuought your program.

If most of your program is pure code, it probably makes sense to go to 32 bit mode, and translate MOV instructions to MOVL. This generally has less problems, but always runs into the problem of having no MOV with sign extend instruction, (except for CVT,) and will mean that you have to re-do any data structures.

In addition, there are several other differences.

Many one or two operand PDP11 instructions appear on the VAX with an additional operand. On PDP11's, the second operand (or first operand for single operand instuctions,) was usually replaced by the result. On a VAX, the result replaces the extra operand. For many instructions, (BIS, BIC) the extra operand is optional, and conversion is straightforward. For some instructions, (ASH for example,) the extra operand is NOT optional, and the conversion program will have some fun syntax to deal with.

ASH

ASH on the PDP11 put the bit shifted out of the Register into the C bit, and on a Right shift, shifts 0's into the highest bit. ASH on the VAX just throws away the bit shifted out of the register, and replicates the sign bit into the most significant bit. In addition, ASH on the VAX also can only be a Long or Quad instruction.

ROR and ROL

The PDP11 Rotate instructions shifted the highest or lowest bit out of the destination into the C bit, and shifted the C bit into the highest or lowest bit of the destination. The VAX ROTL instruction is a 3 operand instruction where one specifies the number of bits to rotate. It also does not include the C bit in the rotation, so is not useable for stripping bits from a variable.

SOB

The Subtract One and Branch if not equal to zero PDP11 instruction has been kept and augmented. One now has SOBGTR, (Subtract One and Branch if Greater than,) plus 3 new ones, SOBGEQ, (Subtract One and Branch if Greater Than or equal,) AOBLEQ, (Add One and Branch if Less Than or Equal,) and AOBLSS, (Add One and Branch if Less than.) Note that these 4 use a longword counter. If you do a MOVW #N, R3 and then do a SOBGTR R3, 11$ you may execute the loop a lot more times than you expect.

Floating point arithmetic.

The PDP11 floating point code is essentially the same on the VAX. Although the names have been slightly changed. The main differences are:

1. The floating point accelerator on the VAX is much better integrated into the CPU. You no longer have to Load the floating point registers or Store them away. The unit will operate directly on the data in memory or registers. (Essentially a MOVF or MOVD instruction works the same way on a floating point variable that a MOVW or MOVB instruction works on integer data.) Similarly the load and convert instructions are no longer needed. The CVT instruction handles conversion from any data type to any other.

2. Since there are no floating point registers on the machine, one must emulate them by code such as the following.

```
                    .PSECT  FPREGS  rd,wrt,noexe

 AC0::              .double 0
  .
 AC5::              .double 0
```

   The CFCC instruction is no longer needed, there are no FPP condition codes. (This actually blew up a couple of pieces of code on me. The code did not expect the C bit to be changed by a Floating Point instruction.)

   New data types are supported, including .G_Floating, (64-bits of data, 11 bits of exponent, 52 bits of fraction,) and .H_Floating, (128 bits of data, 15 bits of exponent, 112 bits of fraction.)

3. On the PDP11 one had to set the calculation mode via the SETF, SETD, SETI and SETL instructions. Instructions such as ADDF and ADDD actually generated the same machine instruction. On the VAX, one does not need to use the SETx instruction, ADDF and ADDD generate different machine instructions.

   A few instructions, (STEXP, LDEXP, and MODF/D and ABSF/D) have also disappeared. In a few cases they will be missed. STEXP was occasionally useful, and the MODF was a very easy way to separate the integer and fraction.

Conditional Branches

The conditional branches are all there, the main difference is that the mnemonics are (in my opinion,) much better. BGTR is Branch on Greater Than, (signed,) BGTRU is Branch on Greater than, Unsigned. The one kicker is that there is often not an exact replacement for a given PDP instruction. The replacements as given in the first article of the series are a best bet replacement, but may not work 100% of the time.

That about wraps up the main differences between PDP and VAX macro. I at least was rather suprised. Outside of getting used to the idea that instructions or data can start at any byte, and remembering that a MOV to a register does not change the most significant bits in byte or word mode, there really is no major difference. VAX macro is really an expanded vocabulary, with many of the idiosyncrasies removed. Next time we will talk about some examples of replacing common system calls.

Leverage

Languages & Tools

STEERING COMMITTEE MEMBERS
LANGUAGES AND TOOLS SIG

Susan Abercrombie
Ventrex Laboratories
217 Read Street
Portland, ME 04103
Library Rep
(207) 773-7231

Bob Awde
General Mills
9000 Plymouth Ave N
Minneapolis, MN 55427
Steering Committee
(612) 540-4432

Mark Bartelt
HSC - Research Development Ctr
555 University Avenue
Toronto, Ontario, Canada M5G 1X8
UNISIG Interface
(416)598-5955

Doug Bohrer
Bohrer & Co.
903 Ridge Rd. Suite 3
Wilmette, IL 60091
Publications Committee
(312) 251-9449

Gordon Brimble
Bldg 180 Labs Area
Defence Research Centre
Box 2151 GPO
Adelaide, S.A. Australia 5001
Australian L&T Interface
(61)(8)259-6119 (office)

Barb Chase
Hughes Aircraft
P O Box 92426
Bldg R1 MSC327
Los Angeles, CA 90009
Human Interfaces Coordinator
(213) 606-1601 (work)

Earl Cory
Cory Computer Systems
366 North Nueve Court
Camarillo, CA 93010
Symposium Coordinator
(818) 706-5385 (work)

Jack Davis
Philips Home Interactive Systems
1111 North Shore Drive
Knoxville, TN 37919
Modula II Coordinator
(615) 558-5206 (work)
(615) 588-5800 (switchbd)

Celeste LaRock
110 Spit Brook Rd ZK02-3/Q08
Nashua, NH 03062
DEC Counterpart

Jim Livingston
Measurex Corporation
1 Results Way
Cupertino, CA 95014
Past Chair
(408) 255-1500 X4468

Dave Martin
Hughes Aircraft Company
PO Box 92426
Bldg R1, MS C320
Los Angeles, CA 90009
Tape Librarian
STUG Interface
(213) 648-9927

Shava Nerad
Systems Alternatives
43 State St
Montpelier, VT 05602
Steering Committee
(802) 229-0823

Al Rizzuto
EMC Control, Inc
PO Box 242
Cockeysville, MD 21030
Wishlist Coordinator
(301) 628-8167 (work)
(717) 456-5014 (recorder)

Don Rosenthal
Space Telescope Science Institute
Homewood Campus
Baltimore, MD 21218
LISP/AI Coordinator
(301) 338-4844 (work)

Tony Scandora
Argonne National Laboratory
CMT 205
Argonne, Illinois 60439
RSX Interface
(312) 972-7541

Bill Segal
Digital Equipment Corp.
110 Spit Brook Rd.
ZK01
Nashua, NH 03062
Counterpart Emeritus
(603) 881-1263

Jim Flatten
Ames Lab
304 Metallurgy
Ames, IA 50001
GAPSIG Interface
(515) 294-4823 (work)

Alan Folsom, Jr.
Fischer & Porter Co.
E. County Line Rd.
Warminster, PA 18974
Newsletter Editor
(215) 674-7154 (work)

Bob Gable
Lear Siegler, Instrument Division
4141 Eastern SE MS 121
Grand Rapids, MI 49508
Ada Coordinator
(616) 241-8273

Dorothy Geiger
Wollongong Logistics Group
49 Showers Drive # 451
Mountain View, CA 94040
Intersig Coordinator
(415) 962-7160

Bernd Gliss
Max-Planck-Institute
Heisenbergstraße 1
7000 Stuttgart 80, W. Germany
European Methods, L&T Interface
(711) 686-0251 (work)

Keith Hare
JCC
PO Box 381
128 West Broadway
Granville, Ohio 43023
DMS & DTR Liaison
(614) 587-0157 (work)

Howard Holcombe
RCA
Front & Cooper St.
Camden, NJ 08055
DEC Personnel Coordinator
(609) 338-4946 (work)

Kathy Hornbach
Lear Siegler/Instrument Division
4141 Eastern SE MS 121
Grand Rapids, MI 49508
Chair
Productivity Tools Coordinator
Pre-Symposium Seminar Coordinator
(616) 241-8800

Mark Kats
GTE Govt Systems
100 First Ave.
Waltham, MA 02154
Session Notes Editor
(617) 466-3437

Kathy Tamer
Rockwell International
1840 Nasa Rd./MS ZC01
Houston, TX 77058
Ada Packages Project
(713) 333-0827 (work)

Pat VanMunn
Measurex Inc.
One Results Way
Cupertino, CA 95014
Methods Coordinator
PSS Committee Representative
(408) 255-1500

Jay Wiley
Bechtel Power Corp
12400 East Imperial Highway
Norwalk, CA 90650
Standards Coordinator
Fortran Coordinator
(213) 807-4016 (work)

JR Westmoreland
Custom Software Products
6748 Acoma Rd
Midvale, UT 84047
Assistant to the Chair
C Coordinator
Commercialism Task Force
TeX/LaTeXCoordinator
(801) 535-4784 (work)

Melodee Westmoreland
Custom Software Products
6748 Acoma Rd
Midvale, UT 84047
Recording Secretary
(801) 533-2350 (work)

Sam Whidden
American Mathematical Society
201 Charles St
PO Box 6248
Providence, RI 02940
36 bit Coordinator
Store Liaison
(401) 272-9500 (work)

Ed Whipple
Lawrence Berkeley Labs
University of California
Berkeley, CA 94720
CMS/MMS Coordinator
Session Chair Coordinator
(415) 486-7167 (work)

Louise Wholey
Measurex Corp
One Results Way
Cupertino, CA 95014
VMS Interface
(408) 255-1500 X4452 (work)

Jim Wilson
QZ Division
PO Box 88
Terre Haute, IN 47808
Commercial Languages Interface
(812) 299-2121 X271 (work)

Table of Contents

Editor's Notes'

Well, the L & T section of the newsletters has been rather thin lately, and I can't even place the blame on a lack of submissions. In fact, I've had a number of good items sent in, and they've been stashed on my desk, because of a recent work crisis. At any rate, think of this issue as a reward, for putting up with the paucity of material over the last few months. There are quite a variety of interesting and informational articles this time.

XD-Ada details a development in the Ada world which I'm sure will be of interest to Ada users.

"Uing VAX PCA To Find Performance Problems" is a transcript of a very popular presentation at last Fall's Symposium. If you like this type of article, let me know and I'll try to arrange more of them.

Next is one of our favorite ongoing topics; the status of the Fortran 8X standard. This article is DEC's response to the question of submitting the proposed standard to X3 for adoption.

To keep up to date with new tools, we have a description of a newly enhanced debugger for the PDP-11, and a status report on the various VAX tools.

If you have not filled in the wishlist questionaire which appeared in the July issue, please do so now. While the deadline is listed as August 15, I'm sure your input can be used, if not in the current wishlist, then in the upcoming one.

Finally, I have a variety of information relating to the forthcoming symposium. This includes a description of a panel discussion regarding text formatters, a description of the seminars to be offered by the L & T SIG, and the abstracts for the L & T sessions.

By now, your plans are probably underway for the Fall Symposium. I hope the information here can help in your planning. Keep the letters and articles coming. Have a nice fall, and I'll see you in San Francisco!

1961-1986 25 Years Leading Toward The Future

Digital Equipment Computer Users Society

DECUS, U. S. Chapter
219 Boston Post Road, (BP02)
Marlboro, MA 01752

U. S. Activities (617) 480-3259 or 3302
Finance & Admin. (617) 480-3634
Library (617) 480-3521

April 15, 1986

TO: APL SIG Members

CC: APL Newsletter
L&T Newsletter
Steering Committee Members

This letter is to recognize and thank the members of the APL SIG and its Steering Committee for their conscientious service to DECUS over many years. The SIG will be dissolving after the Dallas Symposium, having successfully guided the DEC APL products and associated hardware to thei current mature state. The APL product will be welcomed into the Languag & Tools SIG, thus rounding out the technical languages supported by L&T.

The SIG Council and its officers want to salute the APL SIG for these ma years of service. Of special recognition is the level of leadership activity by APL steering committee members. APL SIG has consistently provided a disproportionately large number of very active DECUS leaders for its size. We are sure that, though the affiliations may change, the APL Steering Committee will continue to work hard, promoting the interes of the APL user community and the DECUS community as a whole.

Congratulations on a job well done!

Sincerely,

Sandy Krueger — Jim Welborne — Paula Morin

Sandy Krueger
SIG Council Chair

Jim Welborne
SIG Council Vice-Chair

Paula Morin
DECUS Staff

(on behalf of all SIG Chairs on the SIG Council)

The Population Council

Center for Policy Studies

One Dag Hammarskjold Plaza
New York, New York 10017
Cable: Popcouncil, New York
Telephone (212) 644-1300
Telex: 234722 POCO UR

14 May 1986

Mr. Alan Folsom Jr.
Language & Tools Newsletter Editor
Fischer & Porter Company
E. County Line Road
Warminister, PA 18974

Dear Mr. Folsom:

I would like to make a comment (editorial?) concerning the prospect of FORTRAN-8X and beyond, as discussed in the May newsletter (and on ANSI language standards in general). It appears to me that a major problem in defining a new language standard is the lack of concern for upward compatibility. One of the features of DEC software which makes it extremely attractive is the high priority DEC places on upward compatibility. Going from one version of RSTS to the next has always been very painless. Similarly going from one version of BASIC+2 to the next is no problem -- all my V1.6 programs work perfectly well with V2.3. While DEC understandably wants to encourage better programming by flagging features as declining (CVT, FIELD, and jumping into and out of blocks in BASIC+2 for example) it is to their credit that they continue to support these features in new releases. It would be a disaster for us if we had to go through and re-design all of our applications developed over the years to take out old standard features for which support is dropped.

And now this leads me to FORTRAN-77, FORTRAN-8X, and beyond. ANSI apparently has no concern for the problems it causes when it disregards upward compatibility. Since our organization widely distributes software world-wide, it has always been our policy to write our programs in a very low-level FORTRAN. Our programs strictly adhere to a very minimal standard, even, at times sacrificing some efficiency for portability of code. No special features are ever used even if these features are available on the majority of FORTRAN compilers. Things worked out very well for us for years and years -- and then came FORTRAN-77. And what happens -- no upward compatibility. DO loops work differently (no problem for us because we would always check bounds before entering into one -- but a potential disaster for those who rely on things working the way they always did). And the most disasterous of all -- you can't assign alphanumeric data to non-CHARACTER data types. So, now what is one supposed to do? In FORTRAN-IV there was no CHARACTER data type, the only way to use alphanumberic data was to assign it to any of the available data types. All of a sudden all (and I do mean all) of our carefully designed FORTRAN-IV programs do not work under FORTRAN-77 and would take years of re-writing and re-debugging to make them work.

But, of course, re-writing old FORTRAN-IV programs is far from the answer. What of all the hundreds (thousands?) of installations that have not yet purchased FORTRAN-77 and only have FORTRAN-IV? What happens when they try to compile a newly written FORTRAN-77 program containing CHARACTER statements? Piles of output containing error statements is what happens.

So what does a program developer have to do? We can't insist that users of our software have a FORTRAN-77 compiler. The vast majority of the users of our software are from the developed world with very limited budgets.

And what is a computer installation supposed to do? Upgrade to FORTRAN-77? It is now 1986 and how many computer installations have upgraded from FORTRAN-IV to FORTRAN-77. From my experience not many. Don't get me wrong, FORTRAN-77, with its IF-THEN-ELSE and file handling statements, is a great improvement over FORTRAN-IV -- so why hasn't it become the standard FORTRAN-IV was? The answer is obvious -- you probably can't use it with the majority of your old programs.

While many installations have installed FORTRAN-77, it is clear that if they have been writing in FORTRAN for any length of time, and they do not want to do a lot of re-designing, they are forced to maintain two separate and distinct and incompatible FORTRAN compilers on their system. And now what happens when FORTRAN-8X comes along with its non-upward-compatible features (such as significant blanks)? Are we now supposed to re-rewrite all our FORTRAN-77 code to be runnable on FORTRAN-8X, or are we supposed to maintain 3 separate and distinct and incompatible versions of FORTRAN?

And what happens when FORTRAN-9X comes along without and COMMON, EQUIVALENCE, DATA, DOUBLE PRECISION, etc statements. Are we then supposed to re-re-rewrite all our FORTRAN-8X code to be runnable on FORTRAN-9X, or are we than supposed to maintain 4 separate and distinct and incompatible versions of FORTRAN.

As a developer of software I have resorted to writing a "pre-compiler" program and placing "pre-compiler" commands into my FORTRAN code so that before I send out a program, it is passed through the "pre-compiler" to comment-out and un-comment appropriate lines depending on what FORTRAN it will be run under -- in effect maintaining two versions of the same program. What happens by the year 2000, will I be maintaining 4 versions of every program we distribute? Our reason for choosing FORTRAN as our language of choice for portability of code is no longer valid.

These are obviously serious problems which, I suspect are causing many people many problems. Some companies, such as Microsoft, are extremely helpful and responsive to this problem by making as part of their FORTRAN a command such as $NOTSTRICT which allows most FORTRAN-IV programs to compile and work properly without modification. This has been extremely useful to us in our down-loading of tens of thousands of lines of PDP-11 FORTRAN-IV code to an IBM/PC.

So, what is the solution? The solution is easy -- do what DEC does. Go

ahead and improve FORTRAN -- everything can use improving. Encourage use of new features over old features -- good programming should be encouraged. Even, if necessary print out annoying little warning messages about using older features -- but (1) DO NOT remove a documented standard feature from an earlier version of a language when designing a newer version, and (2) DO NOT make a documented standard feature from a earlier version of a language work differently in a newer version. In other words -- above all, maintain upward compatibility.

So, let me make a suggestion to the ANSI committee designing FORTRAN-8X (or if that's too late, 9X) -- make your highest priority the restoration of features removed from FORTRAN-IV. Let me be able to maintain and program for only one standard FORTRAN -- the latest one.

Sincerely,

Robert Sendek
Computer Specialist

EPA-QAD
P.O.Box 15027
Las Vegas, NV 89114
21-Feb-86

Alan J. Folsom
Dept. 431, Fischer & Porter Co.
County Line Road
Warminster, PA 18974

Dear Mr. Folsom,

Not too long ago I purchased a copy of the "Swedish" Pascal compiler (11-346) and I have been pleased with its performance, however there are some points that I have not been able to resolve. Perhaps one of the newsletter readers could direct me to a solution.

I would like to generate two versions of the compiler, one that runs in and generates code for the EIS, FIS (11/40) environment, and the other for the EIS, FPP (11/60) environment under RSX11-M V 4.2. I have not been able to do this. Only the distributed task image (non EIS, FIS, FPP) operates properly.
Any suggestions?

Sincerely,

Lawrence E. Holboke
702-798-2119

May 20, 1986

Alan Folsom
"LEVERAGE"
Dept. 431, Fischer & Porter Co.
E. County Line Road
Warminster, PA 18974

Dear Mr. Folsom:

The Languages & Tools SIG is the DECUS forum for TECO, and has been ever since the demise of the TECO SIG. But anyone interested in TECO will look long and hard these days to find out anything about it from the SIG, or from anyplace else for that matter. This is not the SIG's fault. TECO users seem to be a tight-lipped group, and DEC is not exactly getting the word out, either.

In fact, TECO is really an endangered species. Now that that the TPU/EVE/LSE family of editing environments has arrived, boredom seems to be the proper attitude to take towards earlier editing products. Well, TECO was never just an editor, as anyone who has used it knows. If it does not merit consideration as an editor (which I would dispute), it certainly does as a language in its own right.

DEC will eventually abandon this un-copyrighted product and its future will be in the hands of the public at large. All along, the main benefits of DEC's stewardship have been to standardize the product and port it to new environments. These benefits will soon cease.

To prepare for the day when DEC abandons TECO altogether, there should be at least a loosely-organized group to pick up the pieces. And for the present, there certainly should be an easier way for TECO enthusiasts to compare notes. I would like to correspond with anyone who is interested in TECO and believes it should be preserved.

Sincerely yours

Phil Wettersten
Borden, Inc.
CIS/ BB-16
180 E. Broad St.
Columbus, OH 43215

February 24, 1986

Dear Editor:

Here are 2 bug fixes that we have found for SED:

```
1. Symptom:
        100% GOTO goes to wrong row, subsequent file modifying command would
        crash SED (it swells up and dies) (TOPS10 and TOPS20)
   fix:  remove "," at PERCD2:-4 in SED1MV
was:
          AOSA  RW,             ;IF SLIDE, MOVE TO START OF NEXT LINE

should be:
          AOSA  RW              ;IF SLIDE, MOVE TO START OF NEXT LINE


2. Symptom:
        large del line would cause SED to bomb. (TOPS10 only)
   fix: at CLSLST+7 IN SED1CM
was:
        MOVNI   T1,CLSBUF(T1)   ;FIND -NUMBER OF WORDS IN THIS LAST PIECE

should be:
        MOVNI   T1,-CLSBUF+1(T1);FIND -NUMBER OF WORDS IN THIS LAST PIECE
```

Neither of these 2 bugs shows up in the VAX version.

If anyone has any more SED bug fixes, please send them to myself or to this publication.

Sincerely,

Dan Tomandl

Dan Tomandl
Locke Computer Center, SB-50
University of Washington
Seattle, WA 98195
206-543-9275
DAN@UWALOCKE (BITNET)

XD-Ada^{R}

Abstract

This paper describes a joint effort between Digital Equipment Corporation (DIGITAL) and Systems Designers (SD) to develop a family of Ada^{R} cross compilers that will involve a VAX^{TM} host and a series of microprocessor targets. The cross-compiler effort will combine the VAX Ada compiler front end, VAX Ada program library manager, and VAX/VMS^{TM} debugger technology with a series of SD-Ada^{TM} back ends and associated SD-Ada tools. The first three microprocessor targets will be the MIL-STD-1750A, Motorola M68000 family, and Intel 80x86 series. Until XD-Ada is available, users will be able to use the existing DIGITAL and SD software to develop microprocessor applications; this paper also includes a description of how the joint use of the current Ada products will be possible.

^{R} Ada is a registered trademark of the U S Government (Ada Joint Program Office).

^{TM} VAX is a trademark of Digital Equipment Corporation

^{TM} VMS is a trademark of Digital Equipment Corporation

^{TM} SD-Ada is a trademark of Systems Designers plc

TABLE OF CONTENTS

1 Digital Equipment Corporation/Systems Designers—XD-Ada

Digital Equipment Corporation and Systems Designers, both leading centers of Ada^{R} expertise, have been involved in the development of Ada since its inception. Both companies offer validated Ada products, and both have development programs to enhance and extend their product ranges.

Digital Equipment Corporation (DIGITAL) has developed and validated an Ada compiler for the VAX^{TM} family of computers (VAX Ada). Because it was produced to run only on the series of VAX computers, the VAX Ada compiler is extremely well integrated with the VAX/VMS^{TM} operating system environment, and its performance is second to none. The VAX Ada compiler can generate code for VAX machines running either the VAX/VMS operating system or the VAXELN^{TM} kernel executive. Appendix A gives a more detailed description of VAX Ada.

Systems Designers (SD) has concentrated on the production of cross compilers (SD-Ada^{TM}) by producing back ends for the 1750A and M68000 processors. (The Intel 80x86 series will be targeted in a future release.) These back ends are used with a VAX/VMS-hosted front end initially developed at Karlsruhe University and SYSTEAM KG, of West Germany. The back-end design stresses portability, making the SD products readily rehostable and retargetable. Appendix B gives a more detailed description of the SD-Ada products.

DIGITAL and SD have now agreed to a joint program that combines the VAX Ada compiler front end, VAX Ada program library manager, and VAX/VMS debugger technology with the SD-Ada back ends and associated SD-Ada tools. The resultant product line—XD-Ada—will combine the best of the two technologies to provide an integrated Ada microprocessor development system. The excellent resource usage of the VAX Ada compiler and the efficiency of the SD-Ada compiler back ends will result in a family of Ada cross compilers and development tools of the highest quality.

XD-Ada will be marketed worldwide by both DIGITAL and SD through their networks of subsidiaries and distributors.

2 XD-Ada

Use of the XD-Ada family of cross compilers will involve a VAX host and a series of microprocessor targets. The first three targets will be the MIL-STD-1750A, Motorola M68000 family, and Intel 80x86 series.

2.1 User Interface Commonality

The XD-Ada host interface will, in general, provide commands identical to those provided by VAX Ada. Where this is not practical, VAX/VMS command conventions will be followed. The overall effect will be that users will have very little to learn in changing from one compiler to another.

The target-independent parts of XD-Ada will be based on DIGITAL's existing software, ensuring commonality of target-independent Ada language features. In particular, the XD-Ada front end will be the same as the VAX Ada front end, making the error messages and tool interfaces available to VAX Ada users also available to XD-Ada users.

The target-dependent parts of the system will be based on SD's existing software, which includes target-dependent tools necessary for the cross-development of software for embedded systems.

[R] Ada is a registered trademark of the U S. Government (Ada Joint Program Office).

[TM] VAX is a trademark of Digital Equipment Corporation

[TM] VMS is a trademark of Digital Fquipment Corporation.

[TM] VAXELN is a trademark of Digital Equipment Corporation

[TM] SD-Ada is a trademark of Systems Designers plc.

2.2 Object Code Quality

High-quality local code generation will result from the use of SD's pattern-matching code generators. At the global level, code quality will be enhanced by performing global data flow analysis and optimization, with particular attention paid to the elimination of redundant run-time checks. The analysis and optimization will ensure that the object code quality will meet the needs of the embedded systems market.

2.3 Program Library Management

The VAX Ada program library manager (ACS) will be extended to become an integral part of XD-Ada. In particular, the ACS automatic recompilation features will be provided, and will allow users to compile from one program library to another library that may have been created for a different target architecture. An XD-Ada user can thus keep software development for a host and target in step, thereby avoiding the problems which occur when recompilation is performed manually.

2.4 Debugging

The XD-Ada debugger will allow users to debug software on the target computer from the host computer. Also, in order to provide maximum capability, the debugging system will be integrated with proprietary target-emulation systems. The debugger user interface will provide most, if not all, of the debugging facilities offered by the VAX/VMS debugger. In particular, XD-Ada's target-independent debugger commands will be identical to those used with VAX Ada.

Additional commands will be provided for specific target computers, and the style of those commands will be compatible with those used in VAX Ada.

2.5 Summary

XD-Ada will provide an efficient compilation system that uses state-of-the-art code generation technology and results in an effective environment for software development for embedded computers.

XD-Ada will benefit from a continuing program to provide more development tools, such as support of other host-target protocols, and interfaces to specialized test hardware that provides emulation and diagnostic capabilities.

Customer support will be provided at the level expected and required by the customer base, and, at a minimum, will match the level of support traditionally provided by both DIGITAL and SD.

3 Making the Transition to XD-Ada

The transition to XD-Ada is envisioned as a two-step process: first, before XD-Ada is available, users will be able to use VAX Ada with the SD-Ada cross compilers to develop microprocessor software; later, XD-Ada will be substituted for SD-Ada, and will provide a simpler, more integrated user interface.

The following sections describe how the transition will be made from the current products to XD-Ada. Appendix C details any differences between the current products, and explains how those differences will be resolved for XD-Ada.

3.1 Step 1: Using VAX Ada and SD-Ada Together

Using VAX Ada and SD-Ada together should not present difficulties for the user: program development using VAX Ada or SD-Ada involves very similar—and generally compatible—activities.

3.1.1 Dealing with Target Dependences

Strict conformance to the Ada language standard by both VAX Ada and SD-Ada means that implementation-independent source may be compiled using either VAX Ada or SD-Ada. Source incompatibilities will arise from target-specific, implementation-dependent features, but these incompatibilities can be resolved during the transition by isolating target-specific, implementation-dependent features in separate compilation units.

Thus, before XD-Ada is available, it is expected that Ada programs needing to be compiled for an SD-targeted microprocessor would be divided into target-dependent and target-independent units. The VAX Ada capability to distinguish and display the target-independent and target-dependent features in a compilation unit will be highly useful at this stage.

The target-independent units would be developed with VAX Ada and tested using a host-dependent test harness, if necessary. The target-independent compilation units would then be compiled for the target system, along with the target-dependent units, using SD-Ada. The complete program, including the target-dependent units, could then be tested, and any errors corrected in the source. The source would then be recompiled, and the program could be tested on either the host or target, as appropriate.

3.1.2 The Program Development Process

VAX Ada or SD-Ada source files may be created using any VAX/VMS editor or text generation program. The VAX Language-Sensitive Editor is particularly appropriate for this purpose. (It is expected that the Editor will be customized to reflect the target-dependent features supported by XD-Ada.)

In both VAX Ada and SD-Ada, the source is compiled into a user-created program library. Once the source has been compiled, it is linked for a particular target using the appropriate linking commands: ACS LINK for VAX Ada and a VAX/VMS or VAXELN target, and BUILD for SD-Ada and a microprocessor target. Both the linker and the builder check for consistency and completeness of the Ada program, create an object module that will elaborate the program's library packages in the correct order, and link together all of the required program units.

The ACS LINK command results in an .EXE file, which may subsequently be executed by means of the VAX/VMS RUN command. The SD-Ada BUILD command determines the location of the linked program in the target memory using information in the build symbol table file, and then downline loads the program to the target for execution. The build symbol table file is produced by means of the SD-Ada Definition Tool.

High-level symbolic debuggers are available for use with both VAX Ada and SD-Ada programs. The VAX/VMS debugger is used for VAX Ada programs, and provides extensive debugging facilities that take full advantage of the underlying operating system. The SD-Ada debugger is more limited in the facilities that it provides, reflecting the restricted capabilites of the target environment.

3.2 Step 2: Replacing SD-Ada with XD-Ada

Once XD-Ada is available, host-target development will be even simpler than it will be with the combined use of VAX Ada and SD-Ada. The XD-Ada system will be substituted for the SD-Ada system, and this substitution will remove any differences in the user interface and in the facilities provided. For example, the lack of communication between the host and target libraries when using SD-Ada with VAX Ada means that changes to common source files cannot automatically be reflected in both the host and target program libraries. This will not be the case with XD-Ada. The use of ACS by XD-Ada will ensure compatible program library management for both host and target libraries.

The net effect of changing over to XD-Ada will thus be a simpler development cycle, with an ability to alternate easily between host and target development. The XD-Ada host-target development process will be much the same as the VAX Ada and SD-Ada host-target development process, but there will be an even greater compatibility among the program development tools. After changing over to XD-Ada, users will find it a simple matter to produce a new version of an existing Ada program, targeted to a different processor.

Appendix A

VAX Ada

VAX Ada consists of an optimizing compiler, an Ada program library manager (ACS), a run-time library, a library of predefined units, and debugger support. VAX Ada is well integrated into the VAX/VMS environment, and can be used with a variety of DIGITAL program development tools. Like other VAX languages, VAX Ada conforms to the VAX Calling Standard, and thus can call and be called by other VAX languages such as VAX FORTRAN, VAX PASCAL, and VAX PL/I.

A.1 The VAX Ada Compiler

The VAX Ada compiler implements the full ANSI/MIL-STD-1815A-1983 language, and is hosted on the VAX/VMS operating system. It produces highly optimized code that can be targeted to VAX machines running the VAX/VMS operating system or the VAXELN executive.

The VAX Ada compiler was developed wholely within DIGITAL, and had the major design requirements that it should be reliable, robust, efficient, highly integrated with VAX/VMS, and highly maintainable. These requirements have been met.

The VAX Ada compiler uses the VAX Code Generator, which is also used by the VAX PL/I and C compilers. The front end of the compiler is Ada-specific, and generates the low-level intermediate language required by the VAX Code Generator. This front end will be modified to become the front end of the new XD-Ada compiler, and will generate a low-level intermediate language well-suited to the code generator technology used by the current SD-Ada cross compilers. Within this front end, the Ada program is represented in a tree form that is conceptually similar to DIANA, but much more compact.

VAX Ada was first validated as of September 17, 1984 using Version 1.4 of the Ada Compiler Validation Capability (ACVC)

VAX Ada was revalidated as of September 6, 1986 using Version 1.6 of the ACVC. At the same time and using the same version of the ACVC, VAXELN Ada was also validated. (VAXELN Ada allows Ada code to be targeted to VAX machines running the VAXELN executive. It complements and builds on VAX Ada by using both the VAX Ada compiler and program library manager. The major differences betweeen VAX Ada and VAXELN Ada are the VAXELN Ada run-time library, additional VAXELN-related predefined units, and support for the VAXELN Remote Debugger.)

A.1.1 Efficiency

One of the major design requirements for the VAX Ada compiler was that it should make efficient use of computing resources, both at compile-time and at run-time.

A representative compilation rate for VAX Ada, on a VAX 11/780, is about 900 lines per minute. This rate is an average; package specifications, which generate little code, tend to compile much faster than this. Package bodies, especially if they contain many instantiations or inlined procedures, tend to compile slower than this. The VAX Ada compiler uses a software demand loading technique to avoid loading definitions from the compilation library until they are actually referenced; this technique greatly reduces the amount of memory required to compile a unit that depends upon many other units.

The code generated by VAX Ada is of high quality; it is comparable to that of VAX PL/I, VAX C, or VAX FORTRAN. For example, on the VAX 8600, VAX Ada demonstrates a single-precision Whetstone benchmark of 5,362 kilo-Whetstone instructions per second (KWIPS).

A.1.2 Integration with the VAX Language-Sensitive Editor

The VAX Ada compiler is fully integrated with the VAX/VMS Language-Sensitive Editor. The Editor is shipped with templates describing the syntax of the Ada language and some implementation-dependent features of VAX Ada. These templates simplify entering or editing a VAX Ada program. It is possible to invoke the VAX Ada compiler directly from the Editor; in this case, any errors reported by the compiler are displayed by the Editor in a separate window; the Editor has commands to allow easy navigation from one error to the next. In some cases, the VAX Ada compiler will suggest a correction to the program; the user can accept this suggestion with a single keystroke.

A.2 Low-Level Language Features

VAX Ada supports most of the features of Chapter 13 of the Ada LRM (representation clauses and implementation-dependent features). These features include length clauses, enumeration representation clauses, record representation clauses, address clauses, pragma INTERFACE, and the generics for unchecked programming.

Additional pragmas are provided to extend the capabilities of pragma INTERFACE; subprograms, objects, and exceptions may be shared between Ada and non-Ada code.

Package MACHINE_CODE is not provided, since the effect can be achieved by providing the body of a subprogram in any of a wide range of other VAX languages. For VAX/VMS targets, a pragma AST_ENTRY is provided to support AST processing; for VAXELN targets, Ada subprograms are connected to VAX hardware interrupts using the VAXELN CONNECT_TO_INTERRUPT system service.

A.3 Program Library Manager

The VAX Ada program library manager (ACS) performs a wide variety of library maintenance and program development functions.

ACS automatically tracks which units in the library have become obsolete because they depend upon a unit that has been changed, and will automatically recompile all such obsolete units in the correct order. This recompilation is done using a saved copy of the original source.

ACS can also scan the local working directory, looking for files that have been edited since the last time they were compiled. In this case, the changed files will be compiled, along with all units that depend upon them, in the correct order. Thus, fixing a bug in a system using VAX Ada can be very easy.

ACS serves as the interface to the VAX/VMS Linker for an Ada program. It first checks that all units required have been compiled, and are not obsolete, and then it invokes the linker for either a VAX/VMS or VAXELN target. Because the target systems are very similar, it is possible to defer the choice of target until link-time, except for modules which explictly use target-dependent features.

ACS allows modules written in any VAX-supported language (such as FORTRAN, MACRO32, or PASCAL) to be entered into the library as the body of an Ada package or subprogram.

A.4 Debugging VAX Ada Programs

Once a VAX Ada program has been linked, it can be debugged using the VAX/VMS Debugger. If the target system is VAX/VMS, the debugger is run directly; if the target system is VAXELN, the program is debugged from the VAX/VMS host using a remote debugger. (The VAXELN target system contains a small debugging kernel, which communicates over Ethernet with the host-based debugger; the host-based debugger is based heavily on the VAX/VMS Debugger.)

The VAX/VMS Debugger is a full-screen, multiwindow, symbolic debugger. It provides a wide range of commands, including commands to examine and modify variables, set breakpoints, watchpoints, and tracepoints, intercept exceptions, step through a program one statement at a time, and so on.

With the VAX/VMS debugger, the user can efficiently debug Ada programs using a familiar, Ada-like syntax for expressions and names. Indexing, slicing, dereferencing, and selection are supported. If a name is overloaded, the debugger can provide a list of possible meanings to allow the user to determine which particular one was intended.

The debugger also provides support for debugging multitasking programs. From the debugger, the user can intercept any or all rendezvous, determine which events a task is waiting for, schedule a particular task (or prevent it from being scheduled), and so on.

Appendix B

The SD-Ada System

The SD-Ada system is an integrated set of software tools for the cross-development of embedded systems in Ada. The host-target mode of working allows the full support of the host system to be employed during the initial stages of development, and utilizes target-specific cross-development tools as an effective means of performing the subsequent, target-dependent development.

B.1 SD-Ada Family of Cross Compilers

The SD-Ada production-quality cross-compilers implement the full ANSI/MIL-STD-1815A-1983 language, and have been designed to produce compact, efficient, code. They provide representation clauses and an interface to assembler code (essential for target development); they also provide comprehensive diagnostics.

The SD-Ada back ends have been designed to be readily retargetable, enabling cross-compilation for a variety of microprocessor targets to be provided. This ease of retargeting comes from the table-driven pattern-matching approach used for code generation. This approach allows high-quality local code selection to be performed efficiently and reliably, and simplifies modification for different target architectures: an architectural description of the new target is used to automatically generate the code generator's code selection phase. Currently, the Motorola M68000 and MIL-STD-1750A processors are supported, and back ends for other microprocessor targets (such as the Intel 80x86 series) are under development.

A number of compilation tools are supplied with the SD-Ada cross compilers: a cross-reference tool, a syntax checker, a pretty printer, and the Library User System (LUSY). LUSY is an interactive tool for Ada program library maintenance, and provides access to compilation unit information: status, dependences, and relationships.

The VAX/VMS x MIL-STD-1750A and VAX/VMS x MC68000/MC68010 cross compilers entered the formal validation process in May 1986 using Version 1.7 of the ACVC.

B.2 Low-Level Language Features

The SD-Ada cross compilers support most of the features of Chapter 13 of the Ada LRM (representation clauses and implementation-dependent features). These features include length clauses, enumeration representation clauses, record representation clauses, pragma INTERFACE, and the generics for unchecked conversion.

The MIL-STD-1750A cross compiler implements the package LOW_LEVEL_IO, which provides the procedures SEND_CONTROL and RECEIVE_CONTROL, using the XIO instruction.

B.3 SD-Assembler

The SD-Assembler is a powerful macro-assembler that allows host development of assembler code. Once developed, this code can then be included in an Ada program by making use of pragma INTERFACE and the foreign code acquisition feature of the program library manager (LUSY).

B.4 Builder System

The builder system links compiled Ada units to produce an image file. For testing purposes, the image file is loaded into the target using an RS232 serial interface; for production use, the memory image file is processed by a programmable read-only memory (PROM) formatter to produce output for a PROM programming device.

In addition to linking, the builder system provides the user with a means of specifying the configurable characteristics of the target, placing the code and various data segments of Ada units in specified target memory locations, and incorporating object code produced by the SD-Assembler.

B.5 Host-Target Debugging

The SD-Ada debugger provides Ada source-level symbolic debugging. It acts on the whole program, and introduces a negligible run-time overhead on the target. Breakpoint, break-in, exception monitoring and control, data examination and modification, statement execution profiles, tracing, logging, and command sequence facilities are all provided. For debugging time-critical code, diagnostic output can be collected on the target for subsequent transmission to the host. This collection process avoids transmission overheads during target program execution.

B.6 Target Run-Time System

The target run-time system provides the support needed to run Ada programs on a bare target processor. The system includes Ada tasking, storage management and exception handling, target library software, and the debug system support (loader and protocol handler). The system is supplied in source form as well as binary form, and can be modified to suit individual user's needs.

Appendix C

Differences in VAX Ada, SD-Ada, and XD-Ada

There are currently a number of user-visible language differences between VAX Ada and SD-Ada, and there will eventually be differences among VAX Ada, SD-Ada, and XD-Ada. A major goal of the development of XD-Ada is to limit the differences between VAX Ada and XD-Ada to those that can be directly attributed to differences in the targets.

This appendix discusses the language-related differences between VAX Ada and SD-Ada, and indicates their expected resolution in XD-Ada. Note that because the detailed specification of XD-Ada is still underway, some of the currently expected resolutions may change. Also, some differences are still unresolved; resolution of those differences will be made as the detailed specification develops.

For convenience, the appendix is organized to follow the applicable chapters of the *Reference Manual for the Ada Programming Language*. Similarities are mentioned only where necessary; it can be assumed that any language features not mentioned here are supported by both VAX Ada and SD-Ada, and will be supported by XD-Ada. For complete summaries of each of the current implementations, see the following documents:

- *VAX Ada and VAXELN Ada Technical Summary*
- *The SD-Ada Compiler Technical Specification and Answers to the Ada-Europe Guidelines VAX/VMS x M68000 Family*
- *The SD-Ada Compiler Technical Specification and Answers to the Ada-Europe Guidelines VAX/VMS x MIL-STD-1750A Variant*

The VAX Ada compiler has been validated, the SD-Ada cross compilers are in process of being validated, and XD-Ada will be validated before it is released. Thus, all variations in the source language accepted by VAX Ada, SD-Ada, and XD-Ada will conform to the Ada Standard (ANSI /MIL-STD-1815A-1983).

C.1 Lexical Elements

The maximum number of characters on a line is 80 for SD-Ada and 120 for VAX Ada.

The line length restricts only the maximum character length of identifiers and numeric literals.

C.2 Declarations and Types

In both VAX Ada and SD-Ada, the representation of language-defined types (INTEGER, FLOAT, and so on) is chosen with the properties of the target machine (such as word size) in mind. When an application requires a type to have specific properties (such as range, precision, or amount of storage used), the recommended procedure is to define a type with the desired properties, and let the compilers map the user-defined type onto one of the available hardware types.

C.2.1 Enumeration Types

For SD-Ada, the maximum number of elements in an enumeration type is 2**31-1; in VAX Ada—and in XD-Ada—the maximum number is 2**16-1. It is expected that, in practice, users will not be needing extremely large enumeration types; however, representation clauses can be used if large component values in enumeration types are necessary.

C.2.2 Integer Types

In VAX Ada, the language-defined type INTEGER has a 32-bit signed representation. VAX Ada also provides two additional signed integer types: SHORT_INTEGER (a 16-bit signed representation), and SHORT_SHORT_INTEGER (an 8-bit signed representation). VAX Ada provides unsigned 8-, 16-, and 32-bit integer types in package SYSTEM.

In SD-Ada, the language-defined type INTEGER has a 16-bit signed representation on the 1750A, and a 32-bit signed representation on the M68000. SD-Ada also provides LONG_INTEGER on the 1750A (a 32-bit signed representation). Subsequent releases will provide a type SHORT_INTEGER on the 1750A (an 8-bit signed representation) and the M68000 (a 16-bit signed representation), as well as a type SHORT_SHORT_INTEGER on the M68000 (an 8-bit signed representation).

It is expected that XD-Ada will provide integer signed types using the same names as SD-Ada. It is not yet determined whether support for unsigned arithmetic will be provided in the initial version of XD-Ada.

C.2.3 Floating Point Types

The number of floating-point types provided by an implementation depends on the target (as long as a number of language rules are followed).

VAX Ada provides the three predefined floating point types required in package STANDARD: FLOAT, LONG_FLOAT and LONG_LONG_FLOAT. In addition, VAX Ada provides four predefined floating point types in package SYSTEM: F_FLOAT, D_FLOAT, G_FLOAT, and H_FLOAT. All VAX Ada floating-point types are implemented using one of the four VAX hardware floating-point representations: F_floating (32 bits), D_floating (64 bits), G_floating (64 bits) and H_floating (128 bits). Type LONG_FLOAT uses the G_floating representation by default, but this can be changed to D_floating using pragma LONG_FLOAT.

The SD-Ada cross compilers support floating-point types that map naturally onto the target computer. For the 1750A and M68000 cross compilers, only type FLOAT is currently supported. Subsequent releases are expected to support additional floating-point types.

The XD-Ada cross compilers will support types FLOAT, LONG_FLOAT, and LONG_LONG FLOAT as appropriate for the target architecture.

	VAX Ada	SD-Ada/XD-Ada	
		M68000	1750A
Type	T'DIGITS	T'DIGITS	T'DIGITS
FLOAT	6	6	6
LONG_FLOAT		NI	NI
using D_floating	9		
using G_floating (default)	15		
LONG_LONG_FLOAT	33	NI	NI

Note: NI - Not Currently Implemented

Type SHORT_FLOAT will not be supported for any of the cross compilers.

C.2.4 Fixed Point Types

In addition to type DURATION, VAX Ada supports 63 anonymous predefined fixed-point types. XD-Ada will also support these fixed-point types on those target machines that provide the ability to generate code for 32-bit operations.

C.3 Names and Expressions

In VAX Ada, static expressions of type *universal_integer* have no limit on the implemented range. SD-Ada imposes a limit of 120 bits, but this limit can be changed to a larger value when the particular cross compiler is generated. XD-Ada will have no limit on ranges of static *universal_integer* expressions.

Static expressions of type *universal_real* will have no limit on the implemented accuracy or range on any of the XD-Ada cross compilers. Evaluation of such expressions during compilation will use a general universal arithmetic package.

In VAX Ada and SD-Ada, nonstatic expressions of type *universal_integer* are evaluated during execution using 32-bit signed binary representation; the same will be true for XD-Ada.

In VAX Ada, nonstatic expressions of type *universal_real* are evaluated during execution using the H_floating representation, and hence are limited to an accuracy of approximately 33 digits and a range of approximately -5.9*(10**4931) .. 5.9*(10**4931). The SD-Ada cross compilers choose the representation of type FLOAT on the target hardware to evaluate nonstatic *universal_real* expressions. Subsequent releases of the cross compilers—as well as XD-Ada—will choose the floating point type that offers the most precision and the greatest range.

C.4 Subprograms

Currently, VAX Ada supports pragma INLINE, with some restrictions. VAX Ada will also automatically inline some subprograms, if the estimated size of the inlined subprogram is approximately the size of the call. Automatic inlining can be suppressed by the user if desired.

SD-Ada does not currently support pragma INLINE.

XD-Ada will support pragma INLINE with the same restrictions as VAX Ada, will also perform automatic inlining, and will allow automatic inlining to be suppressed by the user. The details of the cost-estimation algorithm used to determine whether or not automatic inlining will be done may be different for different target processors.

C.5 Tasks

In VAX Ada—and eventually in XD-Ada—tasks initiated in library packages follow the same rules for task termination as other tasks; tasks are not terminated simply because the main program terminates. (Terminate alternatives in selective wait statements in library tasks are recommended.) This rule has been recently clarified by the Ada language maintenance committee; SD-Ada will conform to it in a future release.

The properties of type DURATION for VAX Ada and SD-Ada are as follows:

	VAX Ada	SD-Ada/XD-Ada	
		M68000	1750A
DURATION'DELTA	1.0e-04	1.0e-14	1.0e-14
DURATION'SMALL	2.0**(-14)	1.0e-14	1.0e-14
DURATION'FIRST	-131072.0000	-86400	-86400
DURATION'LAST	131071.9999	86400	86400

The properties of subtype PRIORITY for VAX Ada and SD-Ada are as follows:

	VAX Ada	SD-Ada	
		M68000	1750A
PRIORITY'FIRST	0	0	0
PRIORITY'LAST	15	15	255
Default for an individual task	7	0	0
Default for a main subprogram	7	1	0

For XD-Ada, these properties have not yet been specified.

VAX Ada does not support pragma SHARED. Instead, it supports the implementation-defined pragma VOLATILE, which guarantees that a variable will be allocated in a main memory location from which the value is fetched and to which the value is updated on each use. Pragma VOLATILE does not force synchronization; it can be used with variables of any type, including variables of composite types.

SD-Ada also does not support pragma SHARED. Support for this pragma is expected in future releases of VAX Ada and SD-Ada, and is planned for XD-Ada. Pragma VOLATILE is also expected to be implemented on the XD-Ada cross compilers.

VAX Ada provides the implementation-defined pragmas TIME_SLICE, to enable round-robin task scheduling, and AST_ENTRY, to allow the handling of VAX/VMS asynchronous system traps. Neither are provided by SD-Ada, nor will they be provided by XD-Ada.

C.6 Program Structure and Compilation Issues

In VAX Ada and SD-Ada, any library procedure can be a main subprogram, provided that it has no parameters. This will also be true for XD-Ada. (Note that for VAX Ada, a function can also be a main program, provided that it returns a value of a discrete type.)

C.7 Exceptions

For VAX Ada and SD-Ada, the predefined exceptions are raised as defined in the *Reference Manual for the Ada Programming Language*. In addition, VAX Ada provides the exception NON_ADA_ERROR (in package SYSTEM), to allow an Ada exception handler to handle VAX conditions raised in imported code. This is very VAX specific, is not supported by SD-Ada, and will not be supported by XD-Ada.

C.8 Representation Clauses and Implementation Dependent Features

VAX Ada supports all of the implementation-dependent features of Chapter 13 that have a useful interpretation in the VAX/VMS or VAXELN environments.

The SD-Ada cross compilers are currently more restrictive, but the restrictions are being relaxed with each new release. It is expected that XD-Ada will implement most of the implementation-dependent features that have a useful interpretation on each target.

C.8.1 Representation Clauses

VAX Ada implements pragma PACK. The pragma is currently not implemented in SD-Ada, but it will be implemented in a future release. Pragma PACK is scheduled for implementation in XD-Ada.

Note that because the behavior of pragma PACK is implementation dependent, users will be advised to use representation clauses to guarantee a particular representation across targets.

C.8.2 Length Clauses

In VAX Ada and SD-Ada, the value given in a size specification for a discrete type must not exceed 32 bits; the specified size becomes the default allocation for all objects and components of that type. The same will be true for XD-Ada. In the SD-Ada cross compilers, there is currently a further restriction that does not permit the size specification to be smaller than the size that would be chosen by the compiler in the absence of a length clause. The XD-Ada compiler is not expected to have this further restriction.

For all other types on VAX Ada and all of the cross compilers, the given size must equal the size that would apply in the absence of a size specification.

The *Reference Manual for the Ada Programming Language* states that for a collection size specification (T'STORAGE_SIZE), the given size becomes the initial and maximum size of the collection. In VAX Ada, in the absence of a collection size specification, or for a size specification of zero, no storage is initially allocated for a collection, and the collection is extended as needed (until all virtual memory for the process is exhausted).

In SD-Ada, target computers that do not have virtual memory will not, in general, have dynamic collections. Currently, the SD-Ada cross-compilers do not support dynamic collections, and if no STORAGE_SIZE attribute is given, the size of a collection assumes a default value, which is not extended. This strategy will also be used in the XD-Ada cross compilers for which no virtual memory is available. XD-Ada may provide dynamic collections for targets with large address space, but the use of this will always be under the control of the user.

For nondynamic collections in VAX Ada, SD-Ada, and eventually in XD-Ada, STORAGE_ERROR will be raised if the collection does not fit into the available stack space.

For a task storage specification (T'STORAGE_SIZE), in VAX Ada, SD-Ada, and eventually in XD-Ada, the given size becomes the initial and maximum size for the task activation (the task stack size). This excludes the space for the task control block, which belongs to the parent. In the absence of a task storage specification (or for a specification of zero in VAX Ada), a default size is used. In any case, the task stack size is fixed at activation and cannot be extended. If the value is less than zero, CONSTRAINT_ERROR is raised.

VAX Ada also provides the pragmas TASK_STORAGE and MAIN_STORAGE to offer the user more control over the management of task and main task storage. It is expected that XD-Ada will provide these pragmas.

In VAX Ada and SD-Ada, the given value for the specification of *small* for a fixed point type (T'SMALL) must be a power of 2.0 (2.0**N, where -31 < = N < = 31) that is less than or equal to the delta of the type. This restriction may be lifted for XD-Ada.

C.8.3 Enumeration Representation Clauses

There are no specific restrictions on enumeration representation clauses for VAX Ada or SD-Ada, nor will there be any restrictions for XD-Ada.

C.8.4 Record Representation Clauses

Record representation clauses are supported by both VAX Ada and SD-Ada.

In VAX Ada, the value in an alignment clause must be a power of 2 (2**N, where 0 < = N < = 9). For stack objects, the alignment must not exceed 4 (longword alignment). For statically and collection-allocated (heap-allocated) objects, alignments up to 512 are supported. Alignment clauses that specify very small storage places are implemented by biasing the representation of the affected component (COMPONENT_SUBTYPE'FIRST is subtracted from the original value).

SD-Ada permits only **mod** 1 for an alignment clause for the 1750A cross compiler, and **mod** 1 or 2 for the M68000 cross compiler. If a record component requires alignment on a word boundary, then the alignment clause has no effect.

The XD-Ada compiler will implement alignment clauses as VAX Ada does.

VAX Ada distinguishes types that are bit-alignable and byte-alignable. Components of bit-alignable types can be allocated beginning at arbitrary bit offsets in component clauses, while components of byte-alignable types must be allocated at byte (addressable storage) boundaries. In general, discrete types, arrays of discrete types, and record types whose size is 32 bits or less are bit-alignable, while other types are not. (The exact rules are a little more complicated; see the *VAX Ada Programmer's Run-Time Reference Manual* for full details.)

Currently, SD-Ada only allows representation clauses which lay out record components on byte boundaries for the M68000 and word boundaries for the 1750A. A subsequent release of the cross compilers will allow bit-alignable representation clauses.

XD-Ada will provide bit-alignable representation clauses, and the restrictions will depend on the support offered by the target hardware for addressing and manipulating those component values.

VAX Ada and SD-Ada do not generate implementation-dependent components or names for record types. XD-Ada will not generate them either.

C.8.5 Address Clauses

VAX Ada supports address representation clauses for variables, but does not support address representation clauses for constants, subprogram, package or task units, or single entries. (In VAX Ada, the implementation-defined attribute AST_ENTRY and pragma AST_ENTRY can be used to map a VAX/VMS asynchronous system trap (AST) to an entry call.)

The SD-Ada cross compilers implement address clauses for objects using a level of indirection. An object declaration with an address clause is treated by each cross compiler as an access object (pointer) whose designated type is the same as the type of the object declaration. Storage for the pointer is allocated at the point of elaboration and initialized to the specified address. Accesses to the object are made indirectly through the pointer. This access object is initialized with the given address at the point of elaboration of the corresponding address clause.

In SD-Ada, address clauses may be given only for objects. Address clauses are not supported for other entities. Unchecked storage deallocation will not work for objects with address clauses, and objects with address clauses may not be initialized.

Further control is provided over the layout of compilation units by the SD-Ada builder. The user can place static areas of compilation units at nominated addresses during the build process.

The XD-Ada cross compilers will implement address clauses as SD-Ada does, except that XD-Ada will not use a level of indirection to access varibles at user specified locations unless the address is not known statically. The compiler will generate code to reference absolute addresses.

The builder facilities for placing units at specified addresses will be available in the XD-Ada cross compilers.

C.8.6 Interrupts

Embedded applications that are tightly coupled with external events often require the ability to map task entries to interrupt vectors. This facility is not provided by VAX Ada; it will be provided by SD-Ada and XD-Ada.

C.8.7 The Package SYSTEM

Package SYSTEM is highly implementation dependent. The constituents of package SYSTEM for VAX Ada and SD-Ada are detailed in the appropriate product documentation.

C.8.8 Representation Attributes

In VAX Ada and SD-Ada, the representation attributes ADDRESS, SIZE, POSITION, FIRST_BIT, LAST_BIT, and STORAGE_SIZE are supported. VAX Ada also provides the implementation-defined attribute BIT, which yields the bit offset within a storage unit of the first bit allocated to an object. XD-Ada will provide this attribute as well. In VAX Ada a value of 0 to 7 is allowed for VAX/VMS and VAXELN targets; In XD-Ada, a value of 0 to 7 will be allowed for M68000 targets, and a value of 0 to 15 will be allowed for the 1750A target.

The implementation-defined attribute MACHINE_SIZE is provided by VAX Ada. It yields the actual size that is allocated for a variable of a type or subtype, taking into account the storage alignment and padding conventions of the VAX Ada implementation. It is expected that this attribute will also be provided by the XD-Ada cross compilers and will have similar semantics to the VAX Ada implementation.

The values of the representation attributes for the VAX Ada and SD-Ada floating point types are summarized as follows:

	VAX Ada Floating Point Representation				SD-Ada/XD-Ada	
Attribute	F	D	G	H	M68000	1750A
MACHINE_RADIX	2	2	2	2	2	2
MACHINE_MANTISSA	24	56	53	113	24	23
MACHINE_EMAX	127	127	1023	16383	128	127
MACHINE_EMIN	-127	-127	-1023	-16383	-125	-128
MACHINE_ROUNDS	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE
MACHINE_OVERFLOWS	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE

C.8.9 Machine Code Insertions

Machine code insertions are not supported by VAX Ada or SD-ADA. However, XD-Ada will support this package.

C.8.10 Interface to Other Languages

In VAX Ada, calls to subprograms written in any VAX language, as well as to system routines, can be made using the language-defined pragma INTERFACE and a series of implementation-defined import-export pragmas. The pragmas IMPORT_PROCEDURE, IMPORT_FUNCTION, and IMPORT_VALUED_PROCEDURE are provided for use in combination with pragma INTERFACE to allow the internal and external names of the subprogram to be different, or to force particular parameter-passing mechanisms.

Corresponding export pragmas are provided to allow VAX Ada subprograms to be called by other VAX languages. Unlike the import pragmas, these pragmas are not used with pragma INTERFACE, and do not allow a choice of parameter-passing mechanisms.

VAX Ada also provides the pragmas IMPORT_OBJECT and EXPORT_OBJECT to allow objects to be imported and exported; the VAX Ada pragma PSECT_OBJECT allows VAX Ada programs to share data with programs written in other VAX/VMS languages.

The SD-Ada pragma EXPORT has the same functionality as the VAX Ada pragma EXPORT_OBJECT. In order to import an SD-Ada subprogram, which must be written in assembler, it is necessary to specify pragma INTERFACE. The assembly language program must have the same name as the Ada subprogram, and must conform to the calling sequence expected by the compiler.

It is expected that XD-Ada will provide pragmas IMPORT_OBJECT, EXPORT_OBJECT, IMPORT_PROCEDURE, EXPORT_PROCEDURE, IMPORT_FUNCTION, and EXPORT_FUNCTION in a manner generally compatible with VAX Ada. External names for items, if specified, must agree with the SD-Ada assembler conventions, and support for the VAX/VMS descriptor passing mechanisms will not be provided.

Pragmas IMPORT_EXCEPTION, EXPORT_EXCEPTION, AST_ENTRY, as well as a variety of predefined types and operations in package SYSTEM are also provided by VAX Ada to allow programs to take full advantage of the VAX/VMS and VAXELN environments (for example the VAX Ada function SYSTEM.IMPORT_LITERAL allows global symbols to be obtained). In SD-Ada and in XD-Ada these pragmas and additional system-related features are and will not be provided.

C.8.11 Unchecked Programming

Unchecked deallocation is supported by both VAX Ada and SD-Ada. (Note that pragma IMPORT_PROCEDURE cannot be used to substitute a user-defined deallocation procedure in VAX Ada.)

Unchecked conversion is also supported by both VAX Ada and SD-Ada, subject to the following: The actual subtype corresponding to the formal type TARGET must not be an unconstrained array type or an unconstrained type with discriminants. Further, in VAX Ada, when the target type is a type with discriminants, the value resulting from a call of the conversion function resulting from an instantiation of UNCHECKED_CONVERSION is checked to assure that the discriminants satisfy the constraints of the actual subtype. In SD-Ada, UNCHECKED_CONVERSION currently generates no checks, and the user is encouraged to write additional code to perform explicit checking for types with discriminants.

For an unchecked conversion in VAX Ada, when the size of the source value is not the same as the target subtype, the value is truncated (high-order bits are ignored) or zero-filled (high-order bits). Unchecked conversions in SD-Ada require that the size of the target type does not exceed the size of the source type.

XD-Ada will support unchecked deallocation and unchecked conversion as VAX Ada does.

C.9 Input-Output

VAX Ada supports the language-defined packages SEQUENTIAL_IO, DIRECT_IO, TEXT_IO and IO_EXCEPTIONS. In addition, VAX Ada provides packages RELATIVE_IO, INDEXED_IO, SEQUENTIAL_MIXED_IO, DIRECT_MIXED_IO, RELATIVE_MIXED_IO, INDEXED_MIXED_IO and AUX_IO_EXCEPTIONS. (VAXELN Ada provides only SEQUENTIAL_MIXED_IO and DIRECT_MIXED_IO; the VAXELN file services do not support relative and indexed files.)

In general, target computers do not have powerful input-output capabilities, so support for the Ada input-output packages is not provided in SD-Ada, and the packages TEXT_IO, SEQUENTIAL_IO and DIRECT_IO are implemented as "null" packages. When any of the input-output procedures in these packages are called, the exception STATUS_ERROR or USE_ERROR is raised.

SD-Ada does provide a simple TARGET_IO package for both the M68000 and the 1750A targets to perform character output to the RS232 port. The 1750A target also offers the LOW_LEVEL_IO package. The two procedures SEND_CONTROL and RECEIVE_CONTROL use the MIL-STD-1750A XIO instruction to implement low level input-output.

The XD-Ada compiler is likely to have the same support for input-output as SD-Ada.

C.10 Predefined Language Attributes

The VAX Ada implementation-defined attributes are:

AST_ENTRY
BIT
MACHINE_SIZE
NULL_PARAMETER
TYPE_CLASS

SD-Ada provides no implementation-defined attributes. XD-Ada will provide the VAX Ada attributes BIT, MACHINE_SIZE, and TYPE_CLASS.

C.11 Predefined Language Pragmas

An Ada implementation has considerable freedom in the way it handles both language-defined and implementation-defined pragmas. However, an implementation-defined pragma cannot change the legality of an Ada program, and an implementation must gracefully ignore any pragmas that it does not recognize.

There are no pragmas that are accepted by both VAX Ada and SD-Ada and that have a significantly different effect. However, there are a number of pragmas that are accepted only by VAX Ada or only by SD-Ada. In most cases, these pragmas have a target-dependent function, and they would normally be used in code that is explicitly target-dependent. A list of the pragmas supported by VAX Ada and SD-Ada follows, with an indication of what will be supported in XD-Ada. An asterisk following the pragma name indicates an implementation-defined pragma.

- AST_ENTRY*—provided by VAX Ada to allow handling of VAX/VMS asynchronous system traps; will not be provided by XD-Ada.
- CONTROLLED—has no effect in VAX Ada or in SD-Ada (the behavior to be gained by the pragma is the default behavior in both VAX Ada and SD-Ada). It will also have no effect in XD-Ada.
- DEBUG*—Provided by SD-Ada. It may be used in conjunction with the debugging system to cause the compiler to allow a specified scalar variable to be monitored during a debugger session at the user's request. Values of the specified variable are stored in a buffer area for later inspection. Thus, specification of this pragma gives simple debugging capability for time-critical applications, without adding overhead to the normal thread of execution. Pragma DEBUG will be provided by XD-Ada.
- ELABORATE—provided by VAX Ada, SD-Ada, and XD-Ada.
- EXPORT*—provided by SD-Ada to externally name data objects held in static areas; equivalent to the VAX Ada pragma EXPORT_OBJECT. Will not be provided by XD-Ada (see EXPORT_OBJECT, below).
- EXPORT_EXCEPTION*—provided by VAX Ada to allow non-Ada code to raise Ada exceptions. Will not be provided by XD-Ada.
- EXPORT_FUNCTION*—provided by VAX Ada to allow non-Ada code to call Ada functions. Will be provided by XD-Ada.

- EXPORT_OBJECT*—provided by VAX Ada to allow non-Ada code to address Ada objects. Will be provided by XD-Ada.
- EXPORT_PROCEDURE*—provided by VAX Ada to allow non-Ada code to call Ada procedures. Will be provided by XD-Ada.
- EXPORT_VALUED_PROCEDURE*—provided by VAX Ada to allow non-Ada code to call an Ada procedure that returns a result and may update its parameters. Will not be provided by XD-Ada.
- IMPORT_EXCEPTION*—provided by VAX Ada to allow Ada programs to raise non-Ada exceptions. Will not be provided by XD-Ada.
- IMPORT_FUNCTION*—provided by VAX Ada to allow Ada programs to call non-Ada functions. Will be provided by XD-Ada.
- IMPORT_OBJECT*—provided by VAX Ada to allow Ada programs to use non-Ada variables. Will be provided by XD-Ada.
- IMPORT_PROCEDURE*—provided by VAX Ada to allow Ada programs to call non-Ada procedures. Will be provided by XD-Ada.
- IMPORT_VALUED_PROCEDURE*—provided by VAX Ada to allow Ada programs to call non-Ada routines that return a result and update their parameters. Will not be provided by XD-Ada.
- INLINE—provided by VAX Ada, but not currently provided by SD-Ada (scheduled for a future release). Will be provided by XD-Ada.
- INTERFACE—Provided by VAX Ada, SD-Ada, and XD-Ada.
- LIST—Provided by VAX Ada, SD-Ada, and XD-Ada.
- LONG_FLOAT*—Provided by VAX Ada to change the default representation of implementation-defined floating point types. Will have a null effect in XD-Ada.
- MAIN_STORAGE*—Provided by VAX Ada to allow users to adjust the size of the task stack used for a main program. Will be provided by XD-Ada.
- MEMORY_SIZE—Provided by VAX Ada, but not by SD-Ada. Will be provided by XD-Ada.
- OPTIMIZE—Provided by VAX Ada, but not provided by SD-Ada. Will be provided by XD-Ada.
- PACK—Provided by VAX Ada, but not currently provided by SD-Ada (scheduled for a future release). Will be provided by XD-Ada.
- PAGE—Provided by VAX Ada, SD-Ada, and XD-Ada.
- PRIORITY—Provided by VAX Ada, SD-Ada, and XD-Ada.
- PSECT_OBJECT*—Provided by VAX Ada to allow sharing of object storage among Ada and non-Ada programs. Will not be provided by XD-Ada.
- SHARED—Not currently provided by VAX Ada or SD-Ada. Is planned for a future release of VAX Ada, and will be provided by XD-Ada.
- STORAGE_UNIT—Provided by VAX Ada, but not by SD-Ada. Will be provided by XD-Ada.
- SUPPRESS—Not provided by VAX Ada or SD-Ada. It is expected that the pragma will be fully supported by VAX Ada in the future. Will be provided by XD-Ada.
- SUPPRESS_ALL*—Provided by both VAX Ada and SD-Ada to cause all checks for NUMERIC_ERROR and CONSTRAINT_ERROR to be suppressed in the compilation unit to which the pragma applies. Will be provided by XD-Ada.
- SYSTEM_NAME—Provided by VAX Ada, but has no effect in SD-Ada. Will be provided by XD-Ada.

- TASK_STORAGE*—Provided by VAX Ada to allow users to adjust the stack storage allocated for tasks. Will be provided by XD-Ada.
- TIME_SLICE*—Provided by VAX Ada to enable round-robin scheduling of tasks. Will not be provided by XD-Ada.
- TITLE*—Provided by VAX Ada to determine the default title and/or subtitle portions of a compilation listing. Will be provided by XD-Ada.
- VOLATILE*—Provided by VAX Ada to specify that a variable may be modified asynchronously. Will be provided by XD-Ada.

Using VAX PCA to Find Performance Problems

Note: This is a transcription of a talk given at the Fall 1985 Symposium in Anaheim; it was the most highly rated L&T session, according to our survey forms. The talk was given by Bert Beander, the supervisor of PCA. A copy of the slides that accompanied the talk appear at the end, so you can see the various tables and histograms the talk references. Special thanks to Danielle Doptis, of Lear Siegler, who took several Saturday afternoons to do the transcription.

I gave a talk two days ago where I described PCA; that was a talk that described features and functionality in PCA. What I thought I would do in this talk is to describe briefly "What is PCA?"; and "What is its functionality?"

I will give a thumbnail sketch, then I will talk about how PCA has been used internally in Digital to improve the performance of several of our products. What I'm hoping that this will do is illustrate how PCA can be used to improve performance.

What is PCA? PCA is a tool to measure and analyze the performance of application programs (in other words user mode programs.)

It is also a tool that can measure test coverage. What that means is it can tell you what parts of your program are either executed or not executed by a given set of test data.

It is a symbolic tool: it uses the debug symbol table to get the addresses and names of all the routines in your program, and to get the line number information. And also to be able to display source lines. Its command interface is in fact very similar to debug in many ways.

I should mention that PCA does not measure *system performance;* it's not that kind of tool. There's a tool called SPM you can use for that purpose. The purpose of PCA is simply to tell the programmer where he is spending time or other resources in his program - so that he can then tune it to improve its performance.

So, what is the structure of PCA? There are two components to PCA, called the *collector* and the *analyzer.*

The *collector* is a part that runs like a debugger, with the user image that you are measuring. It gathers performance data on the image as it runs to completion, and writes that data out to a performance data file.

Then after that is done, you use the *anlayzer* which is a separate post-processor to analyze the performance data file and produce various kinds of reports – histograms and tables and other kinds of reports.

PCA has a rich set of data collection capabilities. There are basically six kinds of data that you can collect with PCA

First of all you can collect *program counter sampling data.* The idea here is that PCA will sample the program counter every ten milliseconds (or you can specify different intervals if you want a slower sampling rate). It will sample the program counter and write all the samples out to the data file. When you reduce that data it tells you where your program is spending its time. This includes not only the CPU time but page faulting time and I/O time as well. It's a good overall measure of where your program is spending its time.

PCA can also collect *page fault data.* For every page fault that your program generates, PCA can get the faulting program counter value: in other words the code address, as well as faulting address, and also a time stamp. That gets recorded in the data file: you can plot that and find out where in your code you are getting the most page faults.

PCA can measure *system service calls.* What we do here is intercept system service calls and every time one of them occurs we just record where it was called, which system service was called, and we record a time stamp as well. For RMS systems services we do some special processing to collect I/O data. We intercept the RMS system services, but we also rummage around in the File Access Blocks and the Record Acces Blocks to dig out the information such as which file you were doing I/O to, how many bytes were transferred, what virtual block number on the file was accessed and various other pieces of information.

In addition to these things, PCA can collect *exact execution counts.* The idea here is that, for a given location in your program, you may want to know the exact number of times that program location is executed. You can do that in PCA at different levels of granularity. So if you want you can just measure the number of times to execute one specific location; or you can specify whole classes of locations like every routine in the whole program. You can measure how many times every routine in the whole program is called or how many times every line in some routine is executed, for example. You have different levels of granularity. Execution count data is slow to collect. There is a high execution overhead in collecting its data points. Usually one doesn't want to do it for every line in the whole program. You want to be a little more selective.

The final thing you can do is *test coverage.* Test coverage is something like getting execution counts but all you really care about is whether the count is zero or non-zero. In other words, did a given piece of code get executed *at all* or did it not get executed? The implementation is the same as the execution counts in that we put breakpoints on all the program locations you specify. But when we measure test coverage we can throw away the breakpoint the first time it is hit, because we don't care about the exact count. Test coverage data can be gathered quite effeciently.

There are also rich analysis capabilities. The fundamental thing you can get, of course, is performance histograms that show, with different levels of granularity, where you are spending time - so you can find out how much time you are spending in each *module* of your program or you can find out how much time you are spending in each *routine*, or even how much time you are spending on *individual lines*. So you can focus in on hot spots down to the line level.

You can also get performance tables. I'm going to have some examples of what these look like. Tables are the same as histograms except that the formatting is different. You get columns and numbers giving the exact number of counts of different kinds that were collected.

PCA, because it uses the debug symbol table which has source file information in it, can display source files in what are called "annotated source listings." So you can see your performance data displayed next to the source text of your program.

The next bullet I have on the slide here says "user controls granularity and presentation." You can first look at the program broken down by module - to see at the module level where time is being spent, which modules are the bad ones. Then you can focus in on individual routines or the individual lines to see where the time is spent. So you can really focus in on hot spots and identify them exactly.

The analyzer has an interactive interface. The intended useage of the analyzer is that you will first generate one histogram, at a coarse level of granualarity; and then you will focus in on hot spots by looking at finer levels of granularity. For many kinds of data you can also look at the data in a number of different ways. For I/O data, for example, you can see how the I/O's are distributed throughout time, as opposed to how they are distributed throughout the program. Or how they are distributed across the different files you are using.

This is what a typical histogram looks like - in this case I have a small program and I had it broken down by module. And you can then see the histogram bar that is proportional to the amount of time spent in each module. In the second module down there is a module called "*prime*"

and in this case that's the one where 60% of the time is spent. So, when you see a histogram like this you know right away that there is no need to improve the efficiency of a routine like the "read error" routine, which doesn't take any time at all. Even if you could eliminate all of the time it takes it's not going to speed up your program. So you want to look at subroutine "prime".

What source listing looks like - this is the procedure *prime* (or logical function *prime* actually.) Here you see the source listing, and next to the source listing are numbers and little abbreviated histogram bars and also numeric percentages. So you can see that in this case line eight takes 60% of the time in this program execution. So if you want to speed up the program that's where you ought to be looking.

What a table looks like - the same data as shown in previous plot. The program was broken down by module, here I show the table form of that display so you can see that exact data counts, and what percentage that is of the total counts. For PC sampling data, which is collected by random sampling process, there is a 95% confidence interval.

The typical usage of PCA is that first of all you observe a performance problem, after that you collect some data with the PCA collector. Then you analyze the data with the PCA anlayzer to try to identify the cause of the perceived performance problem. Once you think you have identified that, the idea is to go back and modify the program to make it more efficient by whatever means you can think of. And then after that you remeasure the performance and repeat this whole cycle. You keep going until you either feel that the performance is good enough or you can't think of any way to improve it any further.

That's my thumbnail sketch of PCA. What I wanted to do next is to start talking about some examples of how it's been used internally in Digital.

The first example has to do with the Ada compiler; when the Ada compiler was in development, and in field test. A couple of problems were observed. First of all, the initialization time seemed unacceptably slow. It seemed to take 3-4 seconds to compile the null program and that seemed kind of strange.

Another concern was the overall compilation rate. This wasn't really perceived as a problem; in fact the VAX Ada compiler is probably the fastest Ada compiler there is but of course the Ada group wanted to make it as fast as they could. So, they wanted to see if they could find some places where they could do some tuning, and get some performance gain.

Another perceived concern was that there seemed to be a large number of page faults when compiling certain kinds of programs.

Those were the problems. So what did they do about it? The approach with PCA was to collect PC sampling data and generate the appropriate histograms, for a variety of Ada program compilations. They also collected page fault data, since that was a concern. They also collected routine execution counts: in other words, the count of the number of executions for every routine in the whole compiler.

And they found some surprises.

What did they find? Well, I mentioned that the initialization time was very slow. What they found is that even a tiny Ada program or even a null Ada program *faults in the entire compiler*.

This was caused by a number of really trivial initialization routines in the compiler. The problem was that these initilization routines were with the routines that they initialized and that meant that they were scattered through the entire virtual address space of the compiler. As a result, if you call one of those routines, of course VMS is going to page in that routine and in fact will page in a bunch of pages nearby figuring you are going to touch those other nearby pages pretty soon.

But that's not going to happen because the next thing to happen is some other initialization routine somewhere else was called and the net effect of all of this is that by calling these initialization routines the entire compiler was paged through during initialization. And that of course was rather unnecessary, because the Ada compiler is really big.

Another finding was about something called the standard environment in Ada. It's sort of a pre-existing package. The way it was implemented in the Ada compiler was that a parse tree for it existed, but that was then run through the compiler. So the standard environment was compiled everytime you compiled any kind of program and that turned out to take a fair amount of time.

What do they do about it? Well, the problem here with initialization routine was a complete surprise to the Ada developers but once they understood what was happening the solution was obvious. They put all of the initialization routines in a separate PSECT; that meant that all those routines were contiguous in memory and so when the first one was called all of them were paged in and after that they were in memory and nothing else in the compiler was paged in. That prevented the entire compiler from being paged during initialization. So, that got a big speed up there.

Then the internal compilation of the standard environment was eliminated by simply doing it when the compiler is built and taking the resulting output and sticking that in the compiler.

There were some additional findings doing general PC sampling on the compiler. One was that there were some common hot spots that were discovered: the memory management routines for example, turned out to be called a lot and to take a fair amount of time.

Also, there were frequent calls to a general purpose routine, whose result usually was already known. Let me explain it this way: in Ada you have types and these types can have public components and private components. Inside a package all of these components are visible but outside of the package only the public components are visible. So there are two versions of each type. So they kept two type descriptions around for each such type. One called a "base type" - that is the publically known type. And there is the "full type" which includes the private components. Now if the two versions are the same, in other words if there are no private components, then the base type and the full type are identical and so the pointer is the same for both of those two types. But there was a routine in the compiler that translated the base type into the full type (or actually the base type pointer into the full type pointer.) And 90% of the time or more it always returned the base type pointer which was the input parameter into the routine. So in effect what this routine almost always returned was its input arguments. So that's what I mean that the result was already known, and yet this routine was called.

There was another problem too, and that has to do with generic instantiation. A generic routine is a routine that is sort of a template for a routine and then you can create different instances of that routine, specifying different types for the various variables within the routine. What has to happen inside a compiler when you use a generic or create an instance of a generic routine is that the parse tree for that routine has to be copied. That's done with a recursive routine and as it turned out because it was done with a recursive routine there was a large number of calls that were caused and that was taking a fair amount of time. So it was very call intensive.

In addition, a lot of code paging was observed, (I will say more about that later.)

So what was the corrective action that was taken?

First of all, memory management: The code was tightened in various places, and some alternate entry points were added. For example, in the standard memory allocation memory entry point, where you return a block of zeroed memory, in a lot of places the memory block was immediately filled in after it was returned by the allocation routine so there really wasn't any point to zeroing the memory. So they provided an alternate entry point, to return an unzeroed block of memory for some performance gain.

In the case where a routine was called repeatedly that 90% of the time returned only its input argument as the output argument: in that particular case it turned out to be easier to determine if the routine to do the symbol table pointer translation was really necessary. So they carried along a flag that said that, then you could eliminate the routine called.

In the generic instanciation case which was really call intensive, there wasn't really any good way to get rid of all of the calls. But you could make it more efficient In Bliss it is possible to specify that the JSB instruction should be used instead of CALLS or CALLG instructions. That is considerably faster. It is also possible to use global registers for parameters, especially parameters that are not going to be changed across all of the recursive calls in these routines. By changing the linkage in this way we were able to speed up the calls even though the number of calls were the same.

Also, I mentioned there was a lot of code paging. To explain that I have to explain a little bit about how the Ada compiler compiles a set of nested routines. It compiles them "inside out", as it's called. The innermost routine is compiled first. And then after that has been completely compiled and logic code generated, then all the data structures for that routine are thrown away.

The thinking here was that you cannot compile the outermost routine first, then throw away the data structures, because you have to keep the data structures to compile the innermost routines. The idea was to reduce the amount of memory kept around for data storage for these compilations. However, it turns out that in Ada it's very common to have lots of small procedures and packages. And the result of that was, when each one of these small procedures was compiled, the little bit of code generated was sent thru the entire backend of the compiler - sent thru optimization, code generation, peephole optimization, and object language generation. That meant that you paged in all of the code pages, in other words all of the parts of the compiler that handle all of those different phases of the compilation process. That was being done a lot of times because there were lots of small little routines being passed through the compiler. That was the cause of the code paging. So the philospy now (I think it has been implemented by now in the development versions of the compiler) is that it's better to save up a bunch of these little routines and then push them all through the backend of the compiler so that you minimize the number of passes through the backend of the compiler and thus the number of times that you page in the whole backend of the compiler. That was kind of an interesting observation on paging.

What were the gains achieved? Initialization originally took about 3 1/2 CPU seconds. By precompiling the standard environment and by changing all of the initialization routines to stay all in one PSECT, that was reduced to a quarter of a CPU second. So that was a quite good gain.

Overall (for all the changes) the Ada compile speed didn't increase all that much but it was still 5-10% faster. That is nothing to sneeze at; it was worthwhile How much you gain depends a lot on the program you have. A program originating a generic instanciation, for example, will see a bigger gain than some other programs, in the compile speed.

The time expended for these improvements was about one person month, except for the last thing I discussed. The code paging in the backend of the compiler being fixed up - I couldn't get any data on that because that has been done too recently.

The next example I wanted to talk about is the TPU editor. The perceived performance problem here was that section file compilation was very slow on the MicroVAX I.

I should explain what a section file is, if you don't know. A section file is a file that defines a user interface to the TPU editor, such as the EDT interface or the EVE interface. It defines all the different keypad definitions you might set up for command definitions and what their semantics are.

The EDT interface, when implemented on TPU, is described by a section file. The problem was that the section file compilation was very slow.

The approach using PCA was to get histograms of PC sampling data and look at that. They also got some histograms on system services usage so they could see how many times each systems service was called.

What did they find? First of all they found that there was some "MATCHC" ("match character") instruction used in the symbol table lookup routine in the section file compiler. (The section file is compiled into an internal format.) That MATCHC instruction took over *half of the total compilation time!*

The thing to remember about MATCHC is that this instruction is emulated in software on the MicroVAX I so it is going to be a very slow instruction on that particular processor. It's also emulated on the MicroVAX II except the MicroVAX II is a much faster machine. So the compilation speed really wasn't perceived as a problem on that machine, but it was on the MicroVAX I. So that was one thing we found

Another thing we found was that the GETJPI system service was called *on every line compiled!* When we looked into it turned out to be just a bug: it wasn't supposed to be called at all. It was measurement code left in, from early development, that they forgot to pull. *[laugnter]*

So the corrective action here was, first of all, to remove the GETJPI call. The other thing they did: when they looked at that match character instruction problem, they realized that the whole symbol table lookup could be done much better with a hash function. So they removed the MATCHC instruction and used the hash function. In fact, they made sure they avoided some of the other emulated instructions.

The improvements achieved are shown on this slide: this is the EDT section file compilation CPU time. On the VAX 750 it went from 36 to 19 seconds so the compilation time was reduced by half approximately. That's not too bad. On the MicroVAX I it went from 6 minutes to 35 seconds! *[laughter]* That was a 90% improvement in the compilation speed. The time expended was one person week to achieve this gain. And so this goes to show if you have the right tool to see what your program is really doing and where it's spending it's time, sometimes you find that it's not at all hard to get a large performance improvement.

My next example concerns MMS, the Module Management System. If you're not familiar with MMS, this is the tool to automate project builds - to determine what modules need to be recompiled based on dependency data which is stored, and file time stamps. This dependency data is stored in something called MMS description file. So this description file will essentially say that Module A depends upon Module B because B is an "include" file that A uses. And so forth. That's my thumbnail description of that tool.

Now the perceived performance problems: There were two of them; one was that the developers felt that there was high overhead. Users felt it even more, I think. The high overhead had to do with determining module status even when no action was needed. In other words, even if a given module didn't have to be re-compiled, just figuring that fact out took quite a bit of time.

Another problem was that MMS was very slow in processing a large number of modules in a description file. If you had a large description file

with hundreds of modules you would be very slow in determining what modules had to be recompiled, and then to generate a command file that would recompile them all.

So, what did the MMS developers do? Well, they used PCA to measure MMS when building itself. I guess MMS is a medium-sized tool: it has about 40 modules. They are almost all in C. Then they collected PC sampling data and used performance histograms to pinpoint the problem. It was a pretty straightforward application of PCA

So what was found? First of all there was heavy use of a C runtime library routine for doing string compares. The other thing that is more significant, is that this developer discovered that there was an "*order* n^2" algorithm for parsing of MMS in the description file. (Meaning that the time the algorithm takes is proportional to n^2 where n is the number of modules in the description file.

Its really bad for you when n gets big.

The next slide shows the corrective action that was taken. First of all the string comparison that had been done in C - that routine was replaced by a more specialized Bliss routine where you can get compare character instructions inline. We happened to use Bliss in this particular case. If you don't have Bliss, the common thing you might do is to recode something in assembly language. To get better performance usually what you lose is generality, but what you gain is performance, so that was what was done here.

In addition, the internal data structures that MMS uses were reorganized so that this order n^2 algorithm could be turned into an order $n \log n$ algorithm. What it meant is that a linear searching algorithm that was used repeatedly in MMS was turned into a binary searching algorithm which of course is much faster when n is large.

So what kind of improvements were achieved? There was a 10% overall time reduction from just using the Bliss routine, instead of using C's string comparison.

There was a much bigger time reduction from reorganizing the internal data structures and using a binary search algorithm. The improvement varies from 30% to 90% depending upon the size of the description file. Since you went from n^2 to $n \log n$, the savings depends upon the size.

The time expended for the first 10% was about one person week. For the rest of the time it turned out to be more complicated to reorganize the internal data structures so that took about another 2 man months approximately. So that was the total time expended.

The next slide shows what the improvements were. This is comparing MMS version 2.0 to MMS version 2.1: it's CPU time of several different programs. MMS building itself went from 30 seconds to 20 seconds. This is for the MMS parsing: it doesn't include the actual compile of all the things you're building: it's just the MMS part of figuring out what should be rebuilt. It went from 30 seconds to 20 seconds so it speeded it up by 33%.

It was also tried in the Basic compiler, which is a bigger program. The Basic compiler MMS description file processing time went from somewhere over 4 minutes to 46 seconds: that's almost a 5 to 1 improvement in speed.

And then there was a customer who had sent in a description file which took 35 minutes and that went down to 1 minute and 50 seconds for a 95% speed up: by a factor of 20 in other words. So this was again, quite worthwhile for that kind of situation.

There is one more example I want to discuss, and this is using PCA on itself. The problem here was with a field test version of PCA. When we were testing, this version was very slow to collect test coverage data for large programs. One group that really brought that home was the Fortran compiler group in which they tried to measure the test coverage on their test system.

The Fortran compiler group has a test system that compiles a whole string of Fortran programs, then executes the programs, and then sees if the results are correct. If you just run the compile part of that process (in other words you don't *run* the actual test programs, just compile them) the Fortran test system requires an hour and 40 minutes of CPU time, on a 785, with 331,000 page faults. That's how long the Fortran compiler test system would normally run.

So, about a year ago, (actually when I was at DECUS announcing PCA) the Fortran compiler group decided to measure test coverage and so they linked in PCA to measure test coverage in every line of the whole program and they ran it.

And they ran it.

And they ran it.

And they ran it....and the final numbers were as follows:

It took 96 hours of CPU time! *[laughter]* That's *four days* on a 785: 188 hours in elapsed time so that's *8 days! [more laughter]*

And that's the whole week I was at DECUS, and a little more!

It racked up 133 million page faults in a thousand page working set in that time.

So the Fortran compiler group thought that this was a bit slow. *[even more laughter]* These numbers were my welcome back present when I returned of course.

What we wanted to do, of course, was to figure out why it was slow. And so for that we needed a performance analysis tool, and of course PCA is a performance analysis tool.

So what we did is that we dummied things up so we could use one copy of the PCA collector to look at the peformance of another copy of the PCA collector measuring test coverage on a small test program. We didn't do it on the Fortran compiler, because we didn't want to wait that long! *[laughter again]*

So what we did was collect and analyze PC sampling data and page fault data. What we found then was that there were two places in the code were all the time went. What we also found was that those two places traverse the same data structure. What we did then was go in with a debugger and just single stepped through to see what that code was doing.

Let me explain what we do in the collector when we measure test coverage. The basic idea is that we find every line in the debug symbol table for the program (in this case the program was the Fortran compiler) and we put a break point in every such line.

When we hit the breakpoint we have to first of all recognize which line it was, so we have a big breakpoint table which is indexed by hashing so the access is very fast. And then we have to get the original opcode that was at the instruction where we put the breakpoint and store that back into the instruction. Then we have to single step over the instruction and then catch the trace-bit trap and then restore the breakpoint – that is what we do for execution counts. So the procedure is: we get a breakpoint fault, we restore the original instruction opcode, we single step over the instruction with a trace bit, we catch a trace bit exception after the instruction is executed, and then we can restore the breakpoint. Now in the case of test coverage, we don't want to restore the breakpoint. We just want to throw it away. But we use the same code in these cases.

Now, the other thing that is significant about this processing is that while we are doing this, it is possible that we will hit another breakpoint. It can happen that we hit the breakpoint, are about to single step over the instruction, and then an AST comes in. That AST may happen to hit *another* breakpoint, and so we have to be prepared to handle the second breakpoint before we have finished handling the first one.

Also, you can get an exception on the instruction you are single stepping over. And then the user may have an exception handler that has breakpoints in it. And so again we have to be able to handle multiple breakpoint operations at the same time.

The way we do that is we maintain a linked list of breakpoint entries that keeps track of the current procedure of stepping over a breakpoint. And normally there should be only one entry on that list; or maybe two or theoretically it could be maybe three or four but. normally only one or two entries on that linked list.

What we found is that the link list kept growing. There were 25,000 coverage breakpoints set in the Fortran compiler when we did it by line. (Of course, by line means only the lines that actually have generated code.)

The bug in the breakpoint processing caused this linked list I was describing to just keep going up to 25,000 entries instead of maintaining its normal length of one or two!

An explanation of this: the way we got rid of the entries in that linked list of breakpoints is that we set a delete bit in the entry and then there was a cleanup routine that would remove deleted entries. Well, there was a bug in the code and this delete bit entry was not set when we were measuring test coverage, and so the linked list just kept building up.

The performance consequences of this bug were: first of all the breakpoint processing, instead of becoming a linear operation where the processing time was proportional to the number of test coverage breakpoints, now became an order n^2 operation because for every breakpoint that occurred we had to scan this list of *all previous breakpoints* that we had hit (whose entries were not deleted like they should have been.) So, we have an order n^2 algorithm and in this case n was *very* large. It was 25.000!

What's more we went through this link list *three times* for every breakpoint: that just made it worse. Once when you first get the breakpoint fault. Once when you return to the user program (there's a routine that is supposed to scan the list for entries to delete). Then when we finally got the trace bit fault when we returned, there was a third scan to look for entries that had been deleted. And of course nothing had been deleted, but anyway there where three scans to the list. The list got very large eventually; it built up to around one megabyte and that meant that every pass through the list caused us to page through a megabyte of memory and that's why we got so many page faults.

So the corrective action then, was to first of all fix this bug where the delete bit was not set. It could have been fixed in a single line, but when we looked at it we realized that if you're going to throw away the breakpoint anyway there was no need to single step over the instruction only to put the breakpoint back in and then remove it right away – we could just start up the program immediately after getting the breakpoint fault. So we did the fix in six lines of code. But it was still a trivial fix. And that caused the breakpoint algorithm to become linear.

So the improvements that we achieved are listed up here. The CPU time went from the 96 hours down to less than 5 minutes, *[laughter]* and the elapsed time went from the 8 days down to less than 5 minutes. Actually, to measure this what we did was we ran the Fortran compiler test system in two successive nights, one collecting test coverage data and one not. And it actually ran *faster* collecting test coverage data! And so I told the Fortran people this just goes to show that PCA makes your programs run faster. *([laughter])* (The real reason for it, of course, is that there is always some noise when you run these things, so it's not exact to the minute.) One was an hour and 39 minutes and another was an hour and 38 minutes: that kind of thing is just on the noise level. All we can say is that we really can't measure how much the overhead is. And of course the page faults then went from 133 million down to about 5,000. And the time expended to get this gain was three person days.

To complete the talk I want to mention some general conclusions you can draw from these examples and others that I have seen.

One is that there are some common problems you tend to see when you're measuring performance that cause bad performance. One fairly common situation is that you have code that s more general than it really needs to be – for example in the Ada compiler we have this routine that translated one kind of symbol table pointer to another symbol table pointer. This was a general purpose routine and it was always correct, but the work it did was in many cases unnecessary. By special casing" the times where it wasn't necessary to do the translation, there was a speedup.

Similarily in the case of MMS where they wrote a routine in Bliss instead of using the C routine. This was case of writing a more specialized piece of code, but one that performs better.

Another common problem that I've seen quite a few times is what I like to call the "hidden n^2 algorithm". In other words you have an algorithm that runs in order n^2 time where n is the number of data items that your are processing or some such thing. But you were not aware of that, or when you wrote it you didn't realize that n would ever be large, or you may simply may have written the algorithm is such a way that you didn't realize that there would be this n^2 type of behavior. This turns out to be a fairly common problem and it's usually best solved by changing the algorithm used. It often comes up in searching cases so you might change it to binary search or a hashing method or some such method to speed things up.

Another conclusion I think one can draw, that I've seen many times, is that programmer intuition is amazing poor when it comes to performance problems. I think programmers have a very good intuition about the *static* structure of their program: in other words what things are done in what parts of the program. But because, (and I guess that's because they look at the code and that shows static structure of the program) they don't usually look at peformamnce data they don't have good intuition for what's really taking the time. And in these examples, in some cases the developers had some intuition about what might have been the problem but in all of the cases they also found surprises that they never could of guessed in a million years. So in general, I think programmer intuition is not a good guide to performance issues.

Another point is that you can often get big performance gains with relatively little effort, and that is one thing I wanted to illustrate through these examples. That's very common and that's really a consequence to the fact that programmers really don't know where the time is going.

And so the final conclusion that I think falls out of the previous conclusions is that you really do need some sort of tool like PCA. Naturally I like to recommend PCA, but you do need some sort of tool that will give you performance information in order to get optimal performance from the application programs.

USING VAX PCA
TO FIND
PERFORMANCE PROBLEMS

VAX PERFORMANCE AND
COVERAGE ANALYZER (PCA)

- o Tool to measure and analyze performance of application programs
- o Tool to measure test coverage
- o Symbolic: uses Debug Symbol Table
- o Command interface similar to DEBUG

STRUCTURE OF PCA

- o Collector gathers performance data into a performance data file
- o Analyzer processes data to produce performance histograms and tables

RICH COLLECTION CAPABILITIES

- Program counter sampling data
- Page fault data
- System service calls
- I/O data (RMS services)
- Exact execution counts
- Test coverage data

RICH ANALYSIS CAPABILITIES

- Performance histograms
- Performance tables
- Annotated source listings
- User-controlled granularity of presentation
- Interactive interface

```
                  VAX Performance and Coverage Analyzer                  Page 1

          Program Counter Sampling Data (386 data points total)

Bucket Name           +----+----+----+----+----+----+----+----+----+----+
PROGRAM_ADDRESS\      |
 PRIMES . . . . . . . |*                                                     0.8%
 PRIME  . . . . . . . |**************************************************   61.9%
 READ_RANGE . . . . . |****                                                  4.7%
 READ_END_OF_FILE . . |                                                      0.0%
 READ_ERROR . . . . . |                                                      0.0%
 OUTPUT_TO_DATAFILE . |                                                      0.3%
 SHARE$FORRTL . . . . |****                                                  5.4%
 SHARE$LIBRTL . . . . |**                                                    2.6%
 SHARE$PCA$COLLECTOR  |                                                      0.0%
 SHARE$DBGSSISHR  . . |                                                      0.0%
 SHARE$PCA$PRVSHR . . |                                                      0.0%
 SHARE$LBRSHR . . . . |                                                      0.0%
 SHARE$SMGSHR . . . . |                                                      0.0%
                      |
                      |
                      |
                      +----+----+----+----+----+----+----+----+----+----+
PCAA>
```

```
                    VAX Performance and Coverage Analyzer                  Page 1

            Program Counter Sampling Data (386 data points total)

Percent     Count  Line
PRIME\
                     1:
                     2: C      Function to identify whether a given number is p
                     3: C      If it is prime, the returned function value is T
                     4: C
  0.0%               5:        LOGICAL FUNCTION PRIME(NUMBER)
  0.0%               6:        PRIME = .TRUE.
  0.3%               7:        DO 10 I = 2, NUMBER/2
 60.4%  ********     8:        IF ((NUMBER - ((NUMBER / I) * I)) .EQ. 0) THEN
                     9:            PRIME = .FALSE.
  0.0%              10:            RETURN
  0.0%              11:        ENDIF
  1.3%              12:    10  CONTINUE
  0.0%              13:        RETURN
                    14:        END

PCAA>
```

```
                    VAX Performance and Coverage Analyzer                  Page 1

            Program Counter Sampling Data (386 data points total)

                                 Data                 95% Conf
Bucket Name                      Count    Percent     Interval
PROGRAM_ADDRESS\
 PRIMES . . . . . . . . . . .      3        0.8%
 PRIME  . . . . . . . . . . .    239       61.9% +/-   4.8%
 READ_RANGE . . . . . . . . .     18        4.7% +/-   2.1%
 READ_END_OF_FILE . . . . . .      0        0.0%
 READ_ERROR . . . . . . . . .      0        0.0%
 OUTPUT_TO_DATAFILE . . . . .      1        0.3%
 SHARE$FORRTL . . . . . . . .     21        5.4% +/-   2.3%
 SHARE$LIBRTL . . . . . . . .     10        2.6% +/-   1.6%
 SHARE$PCA$COLLECTOR  . . . .      0        0.0%
 SHARE$DBGSSISHR  . . . . . .      0        0.0%
 SHARE$PCA$PRVSHR . . . . . .      0        0.0%
 SHARE$LBRSHR . . . . . . . .      0        0.0%
 SHARE$SMGSHR . . . . . . . .      0        0.0%

PCAA>
```

TYPICAL USAGE OF PCA

- Performance problem observed
- Collect data with PCA Collector
- Analyze data with PCA Analyzer to identify cause of problem
- Modify program based on analysis
- Remeasure performance, repeat cycle

VAX PCA EXAMPLE: ADA COMPILER

Performance concerns:

- o Initialization unacceptably slow (3-4 seconds for null program)
- o Overall compilation rate
- o Large number of page faults

VAX PCA EXAMPLE: ADA COMPILER

Approach with PCA:

- o PC sampling histograms for a variety of program compilations
- o Page fault histograms
- o Routine execution counts

VAX PCA EXAMPLE: ADA COMPILER

Findings:

- o Tiny Ada program faults in whole compiler! Caused by trivial initialization routines
- o Internal compilation of standard environment

VAX PCA EXAMPLE: ADA COMPILER

Corrective actions:

- o Initialization routines moved to a separate PSECT
- o Internal compilation eliminated; done when compiler is built

VAX PCA EXAMPLE: ADA COMPILER

Additional findings:

- o Several common hot-spots:
  - Memory management calls
  - Frequent calls to general routine whose result was already known
  - Generic instantiation is call-intensive
- o Lots of code paging

VAX PCA EXAMPLE: ADA COMPILER

Corrective actions:

- o Memory management: Code tightened, added alternate entry points
- o Eliminate unnecessary routine calls when result known ahead of time
- o JSBs and global registers for generic instantiation
- o Modify algorithm to reduce paging

VAX PCA EXAMPLE: ADA COMPILER

Improvements achieved:

- o Initialization: 3 1/2 CPU seconds down to 1/4 CPU second
- o Overall: Ada compile speed 5 - 10% faster
- o Time expended: 1 person-month

(No data yet on reduced code paging)

VAX PCA EXAMPLE: TPU EDITOR

Performance problem:

- o Section file compilation very slow on MicroVAX I

Section file defines a user interface to TPU editor such as EDT or EVE interface

VAX PCA EXAMPLE: TPU EDITOR

Approach with PCA:

- o Histograms of PC sampling data
- o Histograms of system service usage

VAX PCA EXAMPLE: TPU EDITOR

Findings:

- o MATCHC instruction used in symbol table lookup code took over half of total compilation time (emulated in software on MicroVAX I)
- o $GETJPI system service called on every line compiled. Bug!

VAX PCA EXAMPLE: TPU EDITOR

Corrective action:

- o $GETJPI call removed
- o Symbol lookup code rewritten to eliminate use of MATCHC; now uses a hash function

VAX PCA EXAMPLE: TPU EDITOR

Improvements achieved:

- o EDT section file compilation (CPU time):

Processor	Before	After	Reduced
VAX 750	36 sec	19 sec	47%
MicroVAX I	6 min	35 sec	90%

- o Time expended: 1 person-week

VAX PCA EXAMPLE: MMS

Module Management System

- o Tool to automate project builds
- o Determines what modules need to be recompiled based on dependency data and file time stamps
- o Module dependency data given with an "MMS description file"

VAX PCA EXAMPLE: MMS

Performance problems:

- o High overhead to determine module status even with no action needed
- o Slow when processing a large number of modules in description file

VAX PCA EXAMPLE: MMS

Approach with PCA:

- o Measured MMS when building itself (40 modules, mostly in C)
- o Collected PC sampling data
- o Used performance histograms to pinpoint problems

VAX PCA EXAMPLE: MMS

Findings:

- o Heavy use of C Run-Time Library routines for string compares
- o Order(N**2) algorithm for parsing MMS description file, where N = number of modules in file

VAX PCA EXAMPLE: MMS

Corrective actions:

- o Replace string compares in C with specialized BLISS routine using inline CMPC instruction
- o Reorganized internal data structures to make O(N**2) algorithm O(N log N)

VAX PCA EXAMPLE: MMS

Improvements achieved:

- o 10% overall time reduction from using BLISS routine
- o 30% - 90% time reduction from reorganizing data structures (depends on description file size)

Time expended: 10 person-weeks

VAX PCA EXAMPLE: MMS

Improvements in MMS parse phase (CPU time)

Description file	V2.0 time	V2.1 time	Reduced
MMS build	30sec	20sec	33%
BASIC compiler	4min 8sec	46sec	81%
Customer build	35min	1min 50sec	95%

VAX PCA EXAMPLE: PCA COLLECTOR

Performance problem:

- Field test PCA Collector gathered test coverage data extremely slowly
- FORTRAN compiler test system normally takes 1 hour 40 minutes CPU time for compile phase with 331,000 page faults on a VAX-11/785

VAX PCA EXAMPLE: PCA COLLECTOR

With field test version of PCA, gathering test coverage data by line consumed:

- 96 hours (4 days) CPU time
- 188 hours (8 days) elapsed time
- 133,000,000 page faults in 1000-page working set

A bit slow!!!

VAX PCA EXAMPLE: PCA COLLECTOR

Approach with PCA:

- Use one copy of PCA Collector to look at performance of another copy of PCA Collector measuring coverage on a small test program
- Collect and analyze PC sampling and page fault data
- Then use DEBUG to step through code

VAX PCA EXAMPLE: PCA COLLECTOR

Findings:

- 25,000 coverage breakpoints set
- Bug in breakpoint processing caused a linked list to grow to 25,000 entries; normal length is 1 or 2 entries
- List entry delete-bit not set!

VAX PCA EXAMPLE: PCA COLLECTOR

Performance consequences:

- Bug made breakpoint processing run in O(N**2) time, where N = number of breakpoints (25,000)
- Bug caused paging through 1 Mbyte of memory several times for every breakpoint hit

VAX PCA EXAMPLE: PCA COLLECTOR

Corrective action:

- Bug fixed: 6 lines of code modified (could have been fixed in one line)
- Bug fix caused breakpoint algorithm to become linear, O(N)

VAX PCA EXAMPLE: PCA COLLECTOR

Improvement achieved:

- CPU time for FORTRAN compiler coverage: 96 hours down to less than 5 minutes (noise level)
- Elapsed time: 8 days down to less than 5 minutes (noise)
- Page faults: 133,000,000 down to 5,000

Time expended: 3 person-days

CONCLUSIONS

- Some common problems:
  - Code that is too general
  - Hidden O(N**2) algorithm
- Programmer intuition is poor guide to performance problems
- Often can get big performance gains with little effort

FINAL CONCLUSION

You need a tool like PCA to get optimal performance from your application programs

QUESTIONS?

Response to Letter Ballot on submission of FORTRAN 8X
to X3 for processing as an American National Standard
Digital Equipment Corporation

Digital Equipment Corporation votes NO on this ballot.

Technical and Editorial Comments:

Digital's concerns with the current form of the draft FORTRAN revision (FORTRAN 8X) are in these areas:

1. Compatibility with FORTRAN 77

2. Implementability of the language

3. Performance of implementations of the language

We believe that these problems will prevent this form of the language from ever gaining wide acceptance in the computer industry. Thus, we recommend that this document NOT be forwarded to X3 for processing as a draft American National Standard. Based on these problems, we also question the conformance of S8 to its X3/SD-3 program of work. Given below is a discussion of the specific problems.

1 COMPATIBILITY WITH FORTRAN 77

While S8 claims that FORTRAN 8X is an upward compatible extension to FORTRAN 77, it is, nevertheless, a new and very different language. In fact, they are different in most of the ways that languages can be different, for example:

1. Source Form. S8 specifies an entirely new source form, which cannot coexist with the FORTRAN 77 source form. See section 3.3.

2. Compilation model. FORTRAN 77 is one of the only widely used languages that allows a separate, independent compilation model. S8 forces a dependent compilation model because of its strict type compatibility rules.

3. Memory Allocation. FORTRAN 77 requires only a simple static run time storage allocation model. S8 requires a more complex, heap based dynamic run time storage allocation model to support the ALLOCATE and FREE statements and rules.

4. Subset. FORTRAN 77 defines a useful subset for a modest size language. FORTRAN 8X defines no official subset for a very large language.

5. Declarations. FORTRAN 77 defines a simply scoped attribute oriented declaration syntax. S8 defines a more complex syntax, with nested scopes and most aspects of an entity oriented approach.

6. Control Flow Constructs. FORTRAN 77 defines a small number of simple, powerful control constructs, FORTRAN 8X adds several more complex constructs which allow only restricted use.

7. Document format. S8 is a complete rewrite of X3.9-1978, which does not retain the organization, style, metalanguage, or terminology.

Historical precedent would normally give two very different names to languages with so many differences. In addition, there is much evidence for the attempt to abandon the FORTRAN 77 compatibility that does exist:

1. Section 1.6 states an explicit intent to remove deprecated FORTRAN 77 features in a future revision of the standard.

2. Deprecated FORTRAN 77 features are identified by font in the document.

3. Section 1.6 defines the "core" language which excludes these features.

4. Section 1.6 also defines "core conforming" programs and program units. This defines a sanctioned subset that excludes the deprecated features.

5. Section 1.7 defines a "standard module" that must be core conforming.

Moreover, due to the lack of document compatibility, it is unlikely that all FORTRAN 77 conforming programs do indeed conform to the language described by S8 and don't contradict new rules and restrictions of the S8 language.

Even if S8 has managed to retain complete compatibility with FORTRAN 77, the intent is clearly to encourage users to rewrite their programs. The number of changes between FORTRAN 77 and S8 is much greater than for any previous revision of FORTRAN or any other language. This leads to the perception of two incompatible languages glued together. The motivation appears to be an attempt to actually redefine FORTRAN as a new, all-purpose programming language. This is neither needed, nor wanted, by Digital and its user community.

Experience with Digital's user community indicates heavy dependence on all the features of the FORTRAN 77 language, including the features marked as deprecated in S8. While many worthwhile new features are included in the new language, we believe these should be judged by the user community on their own merits. Urging the implementors and users of FORTRAN 77 to convert their programs to use these new features instead is neither productive nor warranted.

We believe that the extent of these compatibility problems are such as to place this document in clear jeopardy of failing to meet the approval of the SPARC review of compliance with the X3/SD-3 (charter) of X3J3. In particular, section 3.0 of the SD-3 discusses the expected benefits of a strong upward compatibility. It is apparent that the problems described above place all these expected benefits in question.

2 IMPLEMENTABILITY OF THE LANGUAGE

The success of the FORTRAN language is due in large part to its wide availability. The small size and simplicity of the FORTRAN 77 language and availability of a standard subset contributed significantly to this wide availability by allowing implementations to be quickly developed even by small groups and in restricted programming environments. Another contributing factor was the degree to which FORTRAN 77 mirrored the language extensions previously available on existing FORTRAN IV implementations.

These conditions for success are not evidenced in the language proposed in the S8 document. Many of the additional features specified by this document are drawn from other languages, not from extended FORTRAN 77 implementations. Examples are the MODULE/USE feature, strong type checking for data structures, user defined operators including operator overloading, user defined generic function definition, and condition handling procedures. Thus, conforming implementations for most widely used processors will be a long time coming.

To illustrate this problem, we present a partial summary of the major work items required to implement the FORTRAN 8X language, starting from a conforming FORTRAN 77 compiler and run time system. Many of these tasks are straight forward in a large operating system environment; small systems vendors will find implementation much more difficult.

1. New source form related:

   1. New parsing method for free form source and significant blanks (in the same compiler with the old form.)
   2. Insignificant separator to replace insignificant blanks
   3. 31 character variable names
   4. End of line comments with "!"
   5. New, column independent, continuation line rules
   6. Multiple statements per line, with ;, introducing problems with line oriented editors and FORTRAN sensitive debuggers

2. New control flow constructs:

   1. Complex new condition handling mechanism at run time
   2. CASE construct
   3. New DO constructs
   4. FORALL construct
   5. Masked array assignments (WHERE construct)

3. Memory Allocation

   1. Heap memory allocation for ALLOCATE and FREE statements, interaction with procedure invocation and return
   2. Stack based variable and array allocation and initialization at run time for recursive functions
   3. Arbitrary size memory temp allocation for array temps

4. New parsing and symbol table manipulation tasks

   1. User defined operator overloading
   2. Variable renaming in USE
   3. Ranged array name and aliased array name remapping
   4. Array constructors
   5. User defined type constant constructors
   6. User selectable exponent letter
   7. New syntax for name attributes in declarations
   8. Declaration checking for IMPLICIT NONE support

5. New Data types

   1. Selectable precision and range floating point types, user defined functions based on these types
   2. User defined structure types with tagged variant structures
   3. Strong type checking for user defined types at compile time and run time

4. BIT data type

5. Parameterized types

6. New dependent compilation scheme

   1. Designing precompiled file format for modules

   2. Method of mapping MODULE names to vendor specific file or library names

   3. MODULE declaration processing, writing module files or library

   4. USE declaration processing, MODULE reading operation

   5. Enforcing name visibility rules

7. Other new array feature support

   1. Assumed shape arrays requiring extensive descriptor processing

   2. Vector subscripts and assignment

   3. Discontiguous array temps and structure temps

   4. New library routines for array transformations

8. New subprogram features

   1. User written generic functions, with combinatorial explosion in the number of different versions required to support user defined precision and range.

   2. Keyword argument mapping to positional arguments

   3. Optional argument support

   4. Internal routines

   5. Recursive functions

   6. INTENT checking

   7. Procedure interface block processing

   8. User defined elemental functions and assignment

9. New I/O statement requirements

   1. Namelist I/O (with very different syntax from existing methods)

   2. New OPEN/CLOSE keywords

   3. New INQUIRE rules

   4. New I/O statement control list keywords

   5. New format codes

10. New intrinsic functions and rules for argument passing

It should be clear from this list that the change from FORTRAN IV to FORTRAN 77 was a small step compared to the giant leap required to go from FORTRAN 77 to FORTRAN 8X.

As was argued above, we believe that the extent of these implementability problems are such as to place this document in clear jeopardy of failing to meet the approval of the SPARC review of compliance with the X3/SD-3. In particular, section 4.0 of the SD-3 discusses the feasibility of development of conforming implementations. It is apparent that the problems described above place all these assumptions in question.

3 PERFORMANCE OF IMPLEMENTATIONS OF THE LANGUAGE

Another major contributor to FORTRAN's success has been the high performance of its implementations. On any vendor's computer system both compile speed and the speed of the generated code are often better for FORTRAN than for any other language. This is partly due to the implementor's paying specific attention to the FORTRAN performance when designing the system, and partly due to the experience of the user community in exploiting these designs with FORTRAN.

In the area of compile performance, the language proposed by S8 contains features with gross performance problems, such as user defined generic functions and user specified precision and range. Even in the absence of such problems, the very size of the language dictates that compilers will grow and slow down by a considerable amount. This will degrade the acceptability of FORTRAN in all development-intensive environments, especially in educational institutions.

A similar case can be argued for run time performance. No one maintains that use of FORTRAN 8X will allow significant performance improvements in user programs. In fact, a significant run time performance degradation is likely.

Had FORTRAN 8X kept to its original timetable, it might have been able to significantly assist in the ability to exploit the emerging market for computer systems capable of processing entire vectors with a single operation. However, the technology of vectorizing FORTRAN 77 compilers has largely displaced this potential, so FORTRAN 8X has no strong positive performance message at all.

While it is theoretically possible that all the new features of FORTRAN 8X can be implemented without slowing down any FORTRAN 77 features, rational design makes it unlikely. The sheer complexity of the language will force the implementor to use design principles for reliability that emphasize simplicity and unity of design. There are many areas of the language where this is likely to happen, but the one that stands out is the array processing features. Here, a number of language features that give considerable power, such as passing aliased arrays, assumed shape arrays, and allocatable arrays, which may force the implementor to use slower methods which will degrade the speed of existing code.

Again, this argument speaks directly to one of the requirements for this standard specified in the X3/SD-3. Section 5.3, in particular, points out the importance of allowing an efficient implementation at a reasonable cost. These efficiency considerations place in doubt the conformance of S8 to this section of the SD-3.

4 DIGITAL'S RECOMMENDATIONS

To express the above concerns in a more constructive way, we offer the following suggestions on how S8 could be made more acceptable.

1. Remove the deprecated features list and accept the fact that all FORTRAN 77 features must remain in all major vendor implementations, so they must remain in the FORTRAN standard. Remove all language concerning the intention of removing these features from future versions of the standard.

2. Remove many of the new features given in S8 whose justification is to "improve" FORTRAN 77 by replacing these deprecated features. An example is the use of shared storage in modules as opposed to COMMON.

3. Remove the incompatible new source form and replace it with one more compatible with FORTRAN 77, that can be intermixed with existing FORTRAN 77 programs. Several such models exist on major vendors' systems. Specifically objectionable are significant blanks and the insignificant separator to replace them.

4. Remove name attributes from declarations. Include only widely implemented extensions such as dimension and data initialization attributes associated with the name. These give most of the benefit of the S8 method, without perturbing the simple keyword processing of FORTRAN 77 style declarations.

5. Remove the MODULE and USE facilities and replace them with the one simple source manipulation feature that is already widely implemented by major vendors' FORTRAN systems: the INCLUDE statement.

6. Simplify the new features to support user-selectable precision and range. For precision, there is a de facto standard practice already in place: the use of size specifiers (e.g. REAL*4) on the REAL and COMPLEX declaration statements.

7. Change the namelist support to use a syntax like the existing industry practice, with a NAMELIST statement in the declaration section, and the reference to a namelist name in the I/O statement. This is a clear example of FORTRAN 8X introducing unnecessary incompatibility with industry practice.

8. Simplify the mechanisms for specifying subarrays. There are 3 overlapping methods specified by S8, triplets, alias arrays, and ranged arrays.

9. Relax the strong type checking aspect of the TYPE facility. This is incompatible with the weak checking used in the rest of the language and not necessary to exploit the power of data structures.

10. Remove user written generic functions, keyword arguments, and user defined operators. These do not contribute to the power of the language to express algorithms and will not benefit most users, while contributing greatly to implementation complexity.

11. Insure that a strictly separate compilation model is allowed as a valid implementation of FORTRAN 8X. This will insure that simple separate compilation implementations are still possible.

A NEWLY ENHANCED DEBUGGER FOR THE PDP-11S

PDP-11 SYMBOLIC DEBUGGER V2.0
(formerly FORTRAN-77 DEBUG)

by
Marilyn Finch
PDP-11 Languages
Software Product Manager

PDP-11 Symbolic Debugger V2.0 is a major release providing I & D space support and an interface to COBOL-81. The name of this product has been changed to emphasize the additional language support. Now not only can the debugger assist programmers in debugging their FORTRAN-77 and MACRO-11 programs, but COBOL-81 programmers can also utilize this timesaving utility. A SET LANGUAGE command is available to allow debugging of multi-language programs.

This version of the debugger has been designed to occupy less than 4k bytes of user task space. Unlike the debugger which is bundled in the COBOL-81 distribution kit, the PDP-11 Symbolic Debugger is much smaller and faster. The PDP-11 Symbolic Debugger allows the user to refer to program locations by symbols or line numbers rather than by addresses, thus, saving valuable programmer time. This also makes the PDP-11 Symbolic Debugger an execellent replacement for ODT. Breakpoints and tracepoints may be set in overlay segments that are not currently resident. A programmer may step through a program by source line, which facilitates source debugging, or by PDP-11 instructions.

The documentation has been totally rewritten for this release. One installation guide contains the information needed to install the PDP-11 Symbolic Debugger on any of the supported operating systems. A language specific manual has also been added to assist the user to debug each specific language. A quick reference guide is provided for experienced users of the debugger highlighting proper syntax for this release. Release notes have been placed on the distribution medium to allow us the opportunity to provide you the customers with the latest possible information. The following lists the contents of the documentation set.

- PDP-11 Symbolic Debugger User's Guide
- PDP-11 Symbolic Debugger Quick Reference Guide
- PDP-11 Symbolic Debugger Installation Guide
- PDP-11 Symbolic Debugger Information for COBOL-81 Users
- PDP-11 Symbolic Debugger Information for FORTRAN-77 Users
- PDP-11 Symbolic Debugger Information for MACRO-11 Users

The PDP-11 Symbolic Debugger is available on RSX-11M, RSX-11M-PLUS, Micro/RSX, RSTS/E, Micro/RSTS, VMS via VAX-11 RSX, and P/OS via the Pro/Tool Kit or the Professional Host Tool Kit.

For further information on this product, contact your Digital Equipment Corporation local sales office.

VAX LANGUAGES AND TOOLS RELEASE STATUS

VAX (tm) Ada (r) - Q*056

- Current Version: V1.2
- V1.2 Started Shipments: February, 1986
- Major Features of V1.0:
  - -Full ANSI Language
  - -Production quality
  - -Highly integrated into VAX/VMS Environment
  - -Multi-language capabilities
  - -Comprehensive diagnostics
  - -U.S. Government validated
  - -Full symbolic debugging support
  - -VAX Language-Sensitive Editor support

(tm) VAX is a trademark of Digital Equipment Corporation
(r) Ada is a registered trademark of the U.S. Government (Ada Joint Program Office)

VAXELN Ada - Q*A97

- Current Version: V1.0
- To Start Shipments: April, 1986
- Major Features:
  - -Compatible with VAX Ada
  - -Retargetable to real-time/embedded environment
  - -Remote debugger
  - -Tailorable run-time environment
  - -Run-time library retargetted from VAX/VMS to VAXELN
  - -Package of interfaces to VAXELN services

VAX APL - Q*020

- Current Version: V2.1
- V2.1 to Start Shipments: May, 1986
- Major Features:
  - -Performance Improvements
  - -APL can call other VAX languages which adhere to the VMS calling standard
  - -Multi-key ISAM

VAX Basic - Q*095

- Current Version: V2.4
- V2.4 Started Shipments: September, 1985
- Major Features:
  - -VAX Language-Sensitive Editor support
  - -CDD support
  - -Contains compile-time directives
  - -Provides structured programming constructs
  - -Conforms to ANSI Minimal Basic

VAX Bliss - Q*106

- Current Version: V4.2
- V4.2 Start Shipments: February, 1986
- Major Features:
  - -Ease of use
  - -/Check qualifiers
  - -/Cross reference switch
  - -VAX Language-Sensitive Editor support

VAX C - Q*015

- Current Version: V2.2
- V2.2 To Start Shipments: May, 1986
- Major Features:
  - -Full Debug support
  - -CDD support
  - -VAX Language-Sensitive Editor support
  - -Improved run-time routines for UN*X compatibility
  - -Shareable run-time library (which is now distributed as part of VMS, not longer on the VAX C binary kit)

VAX Cobol - Q*099

Current Version: V3.3

V3.3 Start Shipments: April, 1986

Major Features:
- -VAX Language-Sensitive Editor support
- -Screen handling extensions
- -Extended DML

VAX Fortran - Q*100

Current Version: V4.4

V4.4 Start Shipments: February, 1986

Major Features:
- -VAX Language-Sensitive Editor support
- -Global optimizations
- -CDD support
- -Records

(VAX Fortran on Ultrix - Q*A99, To start shipments:May, 1986)

VAX Pascal - Q*126

Current Version: V3.3

V3.3 Start Shipments: April, 1986

Major Features:
- -Performance/Runtime Optimizations
- -CDD Support
- -VAX Language-Sensitive Editor Support
- -Compatibility support for VAELN Pascal
- -Source Line Debugging

VAX PL/I - Q*114

Current Version: V2.4

V2.4 Start Shipments: April, 1986

Major Features:
- -VAX Language-Sensitive Editor support
- -CDD support
- -Compile-time pre-processor

VAX RPG II - Q*631

Current Version: V2.0

V2.0 Start shipping: December, 1985

Major Features:
- -Conforms and is an extended implementation of the IBM RPGII defacto standard
- -Fast compile and runtime performance
- -Full screen editor
- -Compatible with IBM implementations on Systems 3, 34, and 38
- -CDD support

VAX Performance and Coverage Analyzer - Q*119

Current Version: V1.1

V1.1 began shipping: December, 1985

Major features:
- -Helps to find execution bottlenecks in application programs
- -Provides test coverage analysis to determine which lines of an application are executed by a given set of test programs
- -Has an interface to the VAX DEC/Test Manager

VAX DEC/CMS - Q*007

Current version: V2.2

V2.2 start shipping: May, 1986

Major Features of V2.0 are:
- -a callable interface
- -new security features
- -significantly improved performance
- -groups for the easy organization of related files

VAX DEC/MMS - Q*500

Current version: V2.1

V2.1 began shipping: December, 1985

Major features of V2.0 are:
- -support for CDD
- -support for TDMS
- -support for FMS

VAX DEC/Shell - Q*143

Current Version: V1.2

V1.2 to begin shipping: May, 1986

Major features of the V1.0 DEC/Shell include:
- -an alternate command line interpreter
- -the script language
- -a set of commonly used UN*X utilities

o DEC/Shell is based on the UN*X V7 Bourne Shell

VAX DEC/Test Manager - Q*927

Current Version: V2.0

V2.0 began shipping: March, 1986

Major features:
- -Ability to test interactive applications on a character cell terminal
- -Incıeased integration with VAX DEC/CMS (can store tests in a CMS library for Test Manager retrieval)
- -Performance Improvements

VAX Language-Sensitive Editor: Q*057

Current Version: V1.3

V1.3 to begin shipping: May, 1986

Major Features:
- -Supports Ada(r), Basic, Bliss, C, Cobol, Fortran, Pascal, Pl/I
- -Edit, compile, review, and correct compilation errors withing a single editing session
- -Speeds up source code entry using formatted language-specific source code templates
- -Provides for interactive editing capabilities during a debugging (VAX Debug) or performance analysis session (VAX Performance and Coverage Analyzer)
- -User tailorable and user extensible
- -Extensive on-line help for supported VAX languages

VAX Scan - Q*495

Current Version: V1.0

V1.0 began shipping: November, 1985

Major features:
- -A complete VAX programming language used to create programs that deal with pattern matching and text transformation
- -Used for creating translators, preprocessors, filters, and parsers
- -To build tools for converting data from other vendor's computing equipment
- -Finds and replaces text in files

VAX NOTES: Q*960

V1 Begins shipping: May 1986

Computer-based conferencing software

- Supports online discussions between groups of people
  - Project team communication
- Maintains a permanent record of discussions
  - Easily accessible by variety of search criteria
- Unique distributed architecture
  - Server based
  - Leadership performance in DECnet networks
  - Easy discussion between geographically dispersed groups
- Easy to learn and use
  - Choice of DEC standard editing styles

VAX Cobol Generator: Q*365

V1 to begin shipping: August, 1986

- 4GL approach to Cobol programming
- Graphical interface
- Generates error-free VAX Cobol code
- direct access to RMS and Rdb
- promotes structured design

VMS - current version is 4.2

PDP-11 FORTRAN-77

Q-number = Q*668, Q*234, Q*100, Q*453

Current Version: V5.0
V5.0 Started Shipments: January 1986
Major Features:

-Added TK50 distribution support on RSX, RSTS/E, Micro/RSX, and Micro/RSTS

FORTRAN-77/VAX to RSX

Q-number =Q*138

Current Version: V5.1
V5.1 Started Shipments: January 1985
Major Features:

-Supports VMS V4.0 and VAX-11 RSX
-Includes patches on previous update kit
-Supports 8600 and 8650 VAX processors
-1600 BPI Magtape distribution

PDP-11 PASCAL

Q-number = Q*128

Current Version: V1.2
V1.2 Started Shipments: January 1986
Major Features:

-Fixes for bugs
-TK50 distribution
-Much more reliable
-Supports RSX-11M V4.1 (or later), RSX-11M-PLUS V2.1 (or later), Micro/RSX V1.1 (or later)
-FIS support eliminated. Supports EIS and FPP math options

PDP-11 Symbolic Debugger
(formerly PDP-11 FORTRAN-77 DEBUG)

Q-number = Q*811, Q*804, Q*232, Q*233, Q*139, Q*421, QBA21

Current Version: V2.0
V2.0 Start Shipments: April/May 1986
Major Features:

-I & D Space support
-Auto-installation
-Now supports debugging of COBOL-81 programs
-TK50 distribution
-Supports 8600 and 8650 VAX processors
-1600 BPI distribution

LaTeX LSE Templates Submitted to DECUS Library

LaTeX is a powerful, easy to use, public domain text formatting package based on TeX. This submission includes a VAX Language Sensitive Editor (LSEDIT) language definition for LaTeX. Using LSEDIT and the LaTeX language definition, a user, regardless of his/her level of experience, can quickly and easily learn to format complex documents using LaTeX. Use of LSEDIT reduces the amount of typing necessary by automatically supplying the user with a set of templates that define the basic structure of a given LaTeX style. These templates can be selected and filled in or deleted as appropriate. The novice user will use the templates extensively, while the more experienced user will use the templates as an aid in remembering infrequently used commands or formats.

The default LaTeX styles supported by the LSEDIT language definition are: article, report, letter and slides (SLiTeX). This submission also includes three new styles for LaTeX: memo, MIL-STD-490 documents, and book form documents. These new styles are also supported by the LSEDIT language definition for LaTeX.

VMS format HELP library entries are included for most of the features within version 2.08 of LaTeX and SLiTeX. The LaTeX source for the ”VAX Language Sensitive Editor (LSEDIT) Quick Reference Guide for use with the LaTeX Environment” is included on the magnetic media.

This package requires LSEDIT to be installed.

This submission was made by Kent McPherson, from Lear Siegler/Instrument Divison.

WHAT'S IN A FORMATTER?

Philosophies in Conflict

Panel Discussion
Languages & Tools SIG

Tuesday, October 7, 1986, 2:00 p.m.

Ballroom H

DECUS Symposium, Moscone Center

San Francisco, California

WYSIWYG or markup language? "What You See Is What You Get" or redefineable embedded commands? Controversy is growing over the "right" approach to document formatting. WYSIWYG formatters, often part of word processing systems, permit the user to arrange a document to his liking on the screen, using a mouse or a light pen or cursor commands, printing the results only when the user is satisfied with the visible image.

Markup languages, sometimes associated with typesetting systems, require commands to be entered along with text, to be processed by the formatter. In some cases, the compiler's output can be previewed before printing, providing a chance to change the commands.

The Languages & Tools SIG has gathered a panel of four well-known and eminently qualifed experts to examine this issue. Dr. Leslie Lamport is the author of LaTeX, a system of document-formatting macros built on Knuth's TeX typesetting language. Dr. Brian Reid is the author of Scribe, a powerful and widely-used markup language. Mark Bramhall is a Consulting Software Engineer at Digital and VAX/TPU project leader. Lawrence Bohn, an experienced software developer, is Director of Product Management at Interleaf Corporation.

The SIG has formulated a series of questions on this topic. In 10-minute presentations, followed by 5-minute "rebuttals", panelists will address these and other issues. After the presentations, the discussion will be opened to audience participation. Immediately following the session, informal discussions with the panelists will continue in the Languages & Tools Campground.

Questions:

- Why are there two (or more?) different philosophies of document formatting? What are the strengths and weaknesses of the two approaches?
- What would an ideal system of each type incorporate? Which, if any, existing systems meet this ideal? Where do they fall short?
- Is the choice of approach likely to be affected by the established style of a target publication?
- How does a document's content—mathematical, scientific, or graphic—affect the choice?
- Is one of these approaches better or worse under certain circumstances, or for certain kinds of users (an author, for example, vs. a secretary)? If so, will some users be forced to learn two systems?
- Is a markup language required at some level of document complexity, while WYSIWYG is suitable (easier to use) for single-use, possibly smaller or simpler, documents? If so, does this imply that markup languages, in order to be powerful and versatile, are likely to be too hard for non-programmers to use? And if that's so, should non-programmers stay out of the kitchen?
- How do requirements for transportability or document re-use affect the choice? Is there nowadays, in fact, an economic imperative that forces almost anything publishable to be also reuseable—that is, resalable in some other format (for electronic submission or on-line retrieval, for example, or for targeted anthologies)? If so, does that circumstance force the use of markup languages, whose embedded commands can be globally redefined, instead of WYSIWYG?
- Is a viable international standard for document formatters possible?
- Is there reason to hope that the best elements of these two philosophies can be combined into a "perfect" formatter?

L&T Seminars for Fall Symposium

The Fall Symposium in San Francisco is coming early this year! The Pre-Symposium seminars will be held on Sunday October 5. As usual, Languages and Tools has lined up an outstanding roster of seminars. You'll be receiving full abstracts and registration details in the mail, but we wanted to give you a quick preview, so that you can start planning for them...

Introduction to LaTeX is an overview of the LaTeX text formatting system. Based on the the TeX typesetting software from Donald Knuth, LaTeX gives the user the power of TeX with a much more user-friendly interface. LaTeX handles math Greek, tables, indices, and other complex document structures with ease. All this software is in the public domain, and is becoming widely used around the world. It runs on VAX/VMS and UNIX systems, as well as IBM PC's and dozens of other computers. (This article was formatted using LaTeX.) We are extremely fortunate to have Leslie Lamport, the author of the LaTeX environment and of the book *LaTeX: A Document Preparation System* as the instructor for this seminar.

An Introduction to Formal Techniques for Software Verification and Specification is a being offered for the first time this fall. Formal verification of software is a technique for proving that a given piece of software is correct, using mathematical logic and related methods. What was once a highly theoretical branch of computer science has now taken on a more practical role in the implementation of actual computer programs and systems where correctness is of primary importance. This seminar will give an overview of the field of formal specification and verification, to give the flavor of what the techniques are, and what they can and cannot do. The US government has a strong interest in this subject, as it relates to computer security. Current government standards and DoD regulations (such as the National Computer Security Trusted Computer System Evaluation Criteria) will be covered. Are these techniques actually applicable to real-world work? The seminar concludes with a review of actual uses of these techniques on several projects – what worked, what didn't, what will be done differently next time. This seminar will serve as an excellent introduction to the topic of formal verification as it handles the subject from multiple viewpoints – from highly theoretical to eminently practical. It will be of value to people working on systems where correctness in both specifications and implementation are critical: to anyone interested in software testing techniques: and as a general introduction to this unique area of computer science.

Software Development Tools: A Practioner's Guide to What They Are; How to Acquire Them: How to Use Them shows how tools provide some of the best opportunities available for increasing programmer productivity. This seminar will provide a practical guide to the several dozen different types of software tools now commercially available, will explain the advantages and disadvantages of each type, and list specific vendors for these tools in the DEC environment. Tools covering all aspects of the software lifecycle, including requirements, design, coding, testing and documentation, will be discussed. In depth coverage will be given on tools available from DEC. The seminar explores the difficult task of justifying tool purchases to software-inexperienced managers, and convincing traditional programmers to try these new techniques. Included are case histories of tool introduction and use, detailing for each tool, initial fears and how they were resolved: user reaction: things we wish we would have known: and how it helped productivity. This is a nuts-and-bolts seminar, of interest to anyone looking for ways to increase software development productivity at their shop.

LT001 CHOOSING AUTOMATED STRUCTURED ANALYSIS TOOLS

June Baker Computer Sciences Corporation

In pursuit of increased productivity and quality, organizations are seeking automated tools to assist in the design and development process. A plethora of automated structured analysis tools currently is available, and organizations must determine which one(s) fulfill their needs.

Our research shows that there is a vast difference in the capabilities and usefulness of commercially available products. We have evaluated several during the past six months in order to recommend sets of tools to ongoing and startup projects at CSC. This session discusses the criteria we used to study a subset of available tools and our results.

LT002 WHAT'S IN A FORMATTER? WYSIWYG VS MARKUP LANGUAGES A PANEL DISCUSSION

Leslie Lamport Digital Equipment Corporation
Brian Reid Digital Equipment Corporation
Mark Bramhall Digital Equipment Corporation
Lawrence Bohm Interleaf Corporation

WYSIWYG formatters permit the user to arrange a document to his liking on the screen, using a mouse or a light pen or cursor commands, printing the results only when the user is satisfied with the visible image. Markup languages require commands to be entered along with text, to be processed by the formatter. In some cases, the compiler's output can be previewed before printing, providing a chance to change the commands.

Which approach to employ is currently a hotly-debated question. A panel of four well-qualified experts examines the relative merits of the two approaches. Dr. Leslie Lamport is the author of LaTeX, a system of document-formatting macros built on Knuth's TeX typesetting language. Dr. Brian Read is the author of Scribe, a powerful and widely-used markup language. Mark Bramhall is a Consulting Software Engineer at Digital and VAX/TPU project leader. Larry Bohn, an experienced software developer, is Director of Product Planning at Interleaf Corporation. The audience will participate in the discussion following panelists' presentations of their views on the critical issues separating these two philosophies. An important and timely debate.

Immediately following this session, informal discussions with the panelists will continue in the Language and Tools campground.

LT003 CHOOSING A DOCUMENT FORMATTING SYSTEM

Richard K Wallace Los Alamos National Laboratory

After surveying available tools for formatting large computer code manuals, we chose the TeX/LaTeX system to be initially implemented on VAX computers. This talk will describe the process, criteria, and alternatives that were considered in arriving at the decision to standardize on TeX.

LT005 USE OF THE VAX DEC/TEST MANAGER IN AN ANSI STANDARD MAINTENANCE PHASE TEST STRATEGY

James W Tibbetts Hughes Aircraft Co.
Robert L Lanphar Hughes Aircraft Co.

The ANSI committee has published standards for test strategy development and documentation. Although primarily designed for use in new products, many of the precepts can be retro-fitted into existing software with the adoption of an aggressive testing policy. One of the necessary functions of such a testing policy is that of configuration control of the tests used; that is, a tracking system for version-to-version modifications and the specific tests used to validate their updated functionality. The DEC/Test Manager provides a strong basis for this type of configuration control, as well as providing an ongoing baseline for product improvement. it is also adaptable to test tracking for products which require secondary analysis to validate their functions; for example, a tool which produces terminal-specific output and requires the substitution of data files into the Test Manager library. This paper will describe some practical testing applications which have been developed using the DEC/Test Manager.

LT006 ISSUES IN LARGE SYSTEM MAINTENANCE, THE TURNOVER

Joseph A Pollizzi, 3rd Space Telescope Science Inst.

The Space Telescope Science Institute (STScI) is currently accepting a large satellite ground system from a major government contractor. The ongoing maintenance and planned enhancements to this system will be performed by an in-house group. This paper will discuss the formation of this group, its experiences in planning for the delivery of this system, the acceptance of it, and the first 6 months afterwards.

Special emphasis will be placed on our choices and usage of various "off the shelf" tools for configuration control, program analysis, and build processing. The Digital Equipment Corporation products of CMS and MMS were chosen as the configuration/build tools for this application. Many of the special build construction tools were developed using VAX Scan. Other than build and configuration control, various other tools have been investigated for automating the testing (VAX DEC/Test Manager), and to provide for some automatic analysis of the software construction and operation (MAT and VAX/PCA).

LT009 AI/LANGUAGES AND TOOLS/UNISIG JOINT RECEPTION

Katherine Hornback Lear Siegler, Inc.

One of the biggest benefits of attending a DECUS symposium is getting to know so many interesting people with common interests -- a chance meeting with someone who has already solved a problem you have been wrestling with for two weeks; or an opportunity to ask a Digital developer an intricate technical question. It is with this in mind that the Artificial Intelligence, UNISIG, and Languages and Tools SIG steering committees will be hosting an informal reception for symposium attendees. The steering committees of all three SIGs will be there, as will the Digital developers from the AI, ULTRIX, Technical Languages and Tools groups. It will be a chance to relax during a hectic day, and meet people interested in the same things you are. It is rumored that some of the Digital developers will be performing an encore -- something that you surely will not want to miss. There will be a cash bar and munches; please come join us.

LT010 LANGUAGES AND TOOLS ROADMAP

Katherine Hornback	Lear Siegler, Inc.
Joel Clinkenbeard	Digitial Equipment Corporation

This session will be a roadmap to sessions and events during the week that are an interest to Language and Tool users. It will provide pointers to detailed sessions on many topics later in the week. You will also be introduced to the Digital developers, and the L&T steering committee members.

The information-packed Languages and Tools folder will be distributed at this session -- it's a collection of over a dozen useful documents, questionnaires, and other goodies that are essential for anyone interested in L&T.

LT011 LANGUAGES AND TOOLS WRAPUP

Katherine Hornback Lear Siegler, Inc.

At this session, we will evaluate the successes and problems encountered at this symposium, and will ask for feedback from the user community. We will also solicit preferences for sessions to be presented at the next symposium. In addition, we try to interest the attendees in participation in the activities of the SIG as volunteers. The current activities of the SIG will be summarized, and you can get to know some of the SIG leadership. If your interests lead you to sessions on languages and tools, come see if you can find some way in which to take a more active role in the Languages and Tools SIG.

LT012 WRITING A TPU SECTION FROM SCRATCH

Ken Coar General Dynamics

In this session, I intend to share my experiences in writing a TPU initialization section from scratch - that is, one not based upon either EVE or the EDT emulator. I intend to discuss my approach and the specific methods I used to implement various features. Since I wrote the section in a 'Self-booting' manner (i.e. I used it to edit itself), I would like to share the order of implementation I used and the functions I developed to, at least partially, emulate Gosling's EMACS.

LT013 CALLING VMS SYSTEM SERVICES FROM VAX C QUESTIONS AND ANSWERS

James Maves	Eaton Corporation
Art Bjork	Digitial Equipment Corporation

This session will be an informal question and answer period for people who desire to have specific question answered about the use of VMS System Services, RMS, and the Run Time Library from VAX C. If you have any questions, answers, techniques, helpful hints, as well as any kinks to be aware of, come and share them.

LT014 VAX FORTRAN AND SYSTEM SERVICES

Earl S. Cory EATON Information Management

New features in version 4 of VAX FORTRAN allow the use of VMS System Services in a much easier manner. The STRUCTURE, MAP, UNION, and RECORD statements eliminate the need of named COMMON areas in calling system services. These new features will be discussed and examples of their use in calling VMS System Services and Run-Time Library functions will be shown. In particular $GETJPI, LIB$GETJPI, and $SYSQIO will be discussed.

LT015 LANGUAGES AND TOOLS QUESTIONS AND ANSWERS

Joel Clinkenbeard Digital Equipment Corporation

This session will provide a means for users to get their questions on languages and tools answered. Representatives from Digital will be available to answer questions on languages such as Fortran, Pascal, C, Ada, Bliss, APL, and Scan, as well as tools such as NOTES, LSE, TPU, PCA, MMS, CMS, DEBUG and DEC/Test Manager. This session will also be a resource for locating more specialized languages and tools not currently supplied by Digital, with knowledgeable users sitting on the panel to answer questions about things like TeX, LaTeX, cross-targeting compilers, requirements and design tools, etc.

LT017 BOUQUETS OR BRICKBATS? USER'S FORUM ON DEC L & T PRODUCTS

Katherine Hornbach Lear Siegler, Inc.

Ever wanted to tell Digital developers how they should be doing their job? What new products they should be working on but seem to be ignoring? Or maybe some feature you really like and would like to see replicated in other products? This is your chance to toss bouquets or brickbats in Digital's direction. We'll have all the Languages and Tools developers and managers that we can find, lined up at the front of the room. Tell them what you like and don't like about how they're doing their job. To get things started, we'll do some general audience surveys about some of the more prominent issues in the L&T area at present, such as text formatting tools and software design tools. Then we'll open it up to comments from the floor -- any Digital L&T product or potential product is fair game.

Part of the session will be devoted entirely to documentation issues. How well are the L&T products documented? Is the online help adequate? Do you have any suggestions on how things could be improved?

(This session will try to stick to more technical aspects of Languages and Tools -- go to the "Digital Asks the L&T User" session to tell Digital that your software costs too much or isn't delivered on time).

LT019 FPAINT - A FORTRAN DATA ENTRY MANAGER

John Sinclair Inland steel Research

FPAINT permits a programmer to interactively design a data entry screen for use in a Fortran application. This provides an easy way to generate complex screen entry applications and maintains a standard user and program interface across applications. With a minimum of system requirements. FPAINT can be ported to any system with Fortran-77 and video display terminals capable of cursor addressing. FPAINT provides integer, real, character, date, and label field types, control of the display attributes, input range checking and type validation, and optional help text for each field. The programmer has complete control over the run-time screen dynamics.

LT020 AUTOMATIC REPORT GENERATION USING TEX/LATEX THROUGH USER PROGRAM

Bipin Junnarkar Dowell Schlumberger Inc.
Jean-Claude Jujeux Dowell Schlumberger Inc.

Use of TeX/LaTeX to produce high quality reports dynamically from user programs is presented. Ability to modify the output format without necessity of re-compiling and re-linking the user program gives additional flexibility. the method dynamically assigns numerical values to TeX/LaTeX variables from within the user program and hence effectively combines user program with TeX/LaTeX typesetting software. This combination allows user to exploit the vast resources of TeX/LaTeX (e.g. font type and size selection etc.) to produce high quality reports. Output could be produced on any output device which supports TeX/LaTeX.

LT022 BUT THAT'S IMPOSSIBLE IN PASCAL OR SYSTEMS PROGRAMMING IN A HIGH-LEVEL LANGUAGE

E.W. Sewell E-Systems, Garland Division

Popular wisdom has always held that systems programming can be done only in MACRO or BLISS. While this is true for code running in kernel mode and/or accessing the system memory space, it does not automatically follow that it is not possible to use high level languages at all.

This session illustrates techniques for systems programming in Pascal, including user-written system services and accessing the system memory space. Examples include a simple program to display various system control blocks and portions of code from an Ancillary Control Process (ACP) written in Pascal. While the examples are in Pascal, the techniques illustrated can be used for any high-level language which supports a record structure, pointers, and the VMS standard calling mechanism.

LT023 IN SEARCH OF THE BEST USER INTERFACE FOR VAX/VMS DEVELOPMENT

Shava Nerad MIT

Available user interface software packages can be expedient, but may not produce the most flexible or the most usable interface design. What are the layered products we could use? What methods are good for language-based development? What can we learn from computer-based instruction interfaces?

LT024 DESIGN CONSIDERATIONS FOR CONFIGURATION MANAGEMENT SYSTEMS

Mark Kidwell Texas Instruments
Steve Ellison Bell Helicopter - Textron

The disparity between Configuration Management Systems affects not only management, but designers. speakers from various companies discuss "general" CM Requirements. Case studies of CM Systems are looked at detailing how to make a CM system friendly to both management and the development community. Requirements for both general commercial and government contracts are discussed.

LT025 SOFTWARE CONFIGURATION MANAGEMENT PANEL

George L Scott Computer Sciences Corporation

A panel of users will discuss configuration management throughout the software life cycle. Both government and non-government environments are included. Problem reporting, source change control and tracking, and relating executable to source are all topics of discussion.

LT026 CONFIGURATION MANAGEMENT CLINIC

G Del Merritt Computer Sciences Corporation

A group of knowledgeable VAX users and Digital Equipment Corporation representatives will be available to answer Configuration management related questions.

LT027 VALIDATING SOFTWARE SYSTEMS

Kenneth G Roller Digital Equipment Corporation

Interactive application systems pose unique problems in the area of validation. This presentation will address these unique problems and show procedures and tools available today to help the software engineer identify and resolve problems. LIMS/SM will be used as an example for these techniques emphasis will be placed on the interactions of the designer and tester throughout the life cycle of the product.

LT029 SOFTWARE PROJECT MANAGEMENT OVERVIEW

Bob Abramson Digital Equipment Corporation

This session will provide an overview of the application of project management techniques to software development.

Software Project Management is the process of facilitating the estimating, planning, and controlling of software development projects. The estimation activity provides a projection of the total effort, development time, and staff levels required to do a software project. planning provides task-level schedules to which projects "commit". Both estimation and planning provide the project manager and project leader with information to do "what if" analysis at differing levels of detail and increasing levels of confidence. Control is a process of comparing, measuring and reporting progress against plan at the project and individual contributor levels. it provides the information to track project status, determine progress trends, and to take corrective action if necessary.

LT030 DEC ASKS THE USERS

Lawrence S Pearl Digital Equipment Corporation

Are willing to be questioned rather than ask questions? If so, attend this session in which Digital Equipment Corporation's personnel will turn the tables on DECUS ATTENDEES. The ground rules (strictly enforced) are that only Digital Equipment Corporation's personnel may ask questions and members of the audience must agree to respond -- individually decisions and software development methodologies in use within your corporations (no proprietary information, please!). By helping us understand our buying decisions and work modes, we'll be better able in the future to provide the products and processes you need.

LT031 WRITING APPLICATIONS IN VAX SCAN

Jim Totton Digital Equipment Corporation

This session will discuss techniques for writing more sophisticated VAX SCAN programs. The talk will assume the ATTENDEES have an understanding of the basic SCAN programming constructs. It will build on these to show how to build language translators. Topics covered will include the design of tokens and macro pictures, error recover, and the use of trees.

LT033 USING TOOLS IN THE MAINTENANCE PHASE OF THE SOFTWARE LIFE-CYCLE

Software Development Tech Group Digital Equipment Corporation

Maintaining complex software system is often costly in terms of time and development resources. Often resources for new development must be balanced against resources needed to maintain existing software. up from planning for the maintenance phase as well as the smart use of available tools and methods to automate the maintenance process will result in large gains productivity.

Representatives of the Software Development Technologies Group will discuss methods used within Digital Equipment Corporation to maintain software. The role of tools such as the VAX Debugger, the VAX Performance and Coverage Analyzed, the DEC/Test Manager, VAX/SCAN, DEC/CMS and the VAX Language Sensitive Editor in the Maintenance phase of the software development cycle will be

described and numerous examples will be given.

LT034 INTEGRATED DEVELOPMENT ENVIRONMENT CRYSTAL BALL

Software Development Tech Group Digital Equipment Corporation

As larger and more complex software systems are being developed, it is generally recognized that the use of software development tools throughout the project life-cycle help to increase programmer productivity, assist in project management and improve overall quality of the software being produced. Many tools are aimed at specific sections of the life-cycle and are not as a result additional work has to be done manually to relate information through the development process.

Representatives of Digital's Software Development Technologies Group will discuss integrated development environments which link tool functionality and information which can then be accessed throughout the development process. Discussion will center on the needs and requirements of an integrated environment as well as the ties that are needed from the Requirements stage on through Design, Implementation, Testing and maintenance both from the viewpoint of a project manager as well as a developer.

LT035 VAX DEC/TEST MANAGER DEMONSTRATION

Software Development Tech Group Digital Equipment Corporation

A representative from Digital Engineering will give an on-line demonstration of the capabilities of the DEC/Test Manager V2, [one of Digital's VMS programmer productivity tools]. This session will demonstrate how to use the DEC/Test Manager to automate the regression testing process for both interactive and non-interactive tests and how to integrate your testing tools with other tools available.

Features to be demonstrated include test system setup, recording and replaying terminal sessions, as well as comparing terminal screens and reviewing test results. Additional features, including integration with DEC/Code Management System (DEC/CMS) and VAX Performance and coverage Analyzer (VAX/PCA), as well as automate filtering capabilities will also be presented.

LT036 THE TESTING PHASE OF THE SOFTWARE LIFE CYCLE

Software Tool Developer Digital Equipment Corporation

Anyone doing software development, or programming of any type, wants to produce "correct" software. It has been estimated that 50% of the effort and more than 50% of the cost of developing software are the efforts that must be put into testing and validation of the software. This includes the cost of locating errors, correcting errors, verifying the correction, and guarding against further regression of the system with the addition of new functionality throughout the development cycle.

Representatives of the Software Development Technologies Group will discuss the testing process they employ at Digital. A short discussion of the definition of testing and formal strategies for testing will be followed by are used by Digital engineers as well as the methods and strategies used by the engineers in satisfying these needs will be presented. These tools include the VAX Debugger, the VAX Performance and Coverage Analyzer, and the DEC/Test Manager, among others.

LT037 VAX DEC/SHELL AND VNXSET

Susan Azibert Digital Equipment Corporation

VAX DEC/Shell is a port of the UNIX V7 Bourne Shell and many of the most popular UNIX utilities to VMS and Micro/VMS. The DEC/Shell runs as an alternate command line interpreter to DCL while at the same time allowing easy access to VMS functionality. DEC/Shell, along with VAX C, VAX DEC/CMS, and VAX DEC/MMS, make up the layered product set called VNXset. All of the VNXset products will be discussed in this talk.

LT038 INTRODUCTION TO VAXSET

Susan Azibert Digital Equipment Corporation

This session introduces VAXset a powerful collection of proven software engineering tools: VAX Language-Sensitive Editor, VAX Performance and Coverage Analyzer, VAX DEC/Test manager, DEC/CMS (Code Management System), and DEC/MMS (Module Management System).

The focus will be on what the tools do to help solve problems and how they can work together in the VMS environment.

While primarily a presentation of the technical capabilities of the tools when used together, the session will also cover how their usage can simplify and enhance the process of software development and maintenance.

LT039 USING VMS SYSTEM SERVICES FROM VAX C

Art Bjork Digital Equipment Corporation
James Maves EATON Corporation

This session will address the use of VAX System Services and standard Run Time Libraries from C programs. It will include examples of C programs that use VMS system services and will give techniques for making it easier and more efficient to use the VMS utility routines.

LT040 MULTIPROCESS ADA RESEARCH RESULTS

Joel Clinkenbeard Digital Equipment Corporation

The Ada programming language has tasking constructs built into the language. Today, tasks in VAX Ada applications run within a single process. This does not allow the tasks to take advantage of multiprocessor systems.

Digital Equipment Corporation has been researching the solutions to allowing Ada tasks to be run in separate processes and therefore to allow the tasks to run in parallel on multiprocessor systems. The results of this research will be discussed.

(R) Ada is a registered trademark of the Ada Joint Program Office

LT041 HOW MULTIPROCESSOR SISAL IS IMPLEMENTED

Becky Will Digital Equipment Corporation

SISAL is a "data flow" language, designed to expose the parallelism inherent in an application. Its unusual features include a single-assignment rule, handling of errors, and a stream datatype. Unlike many parallel programs, SISAL programs are deterministic.

Digital Equipment Corporation is working on a SISAL compiler in conjunction with research labs and universities doing research on parallel processing. This talk will describe how the compiler can automatically decompose SISAL application programs to take advantage of the 8300 and 8800 dual processors.

LT042 PARAMETER PASSING IN VAX PASCAL

Becky Will Digital Equipment Corporation

The VAX Common language Environment allows programs written in VAX PASCAL to call useful routines which may be supplied or more easily written in another language. Frequently getting the arguments in a form that the calling program expects seems like black magic. This talk will demystify the incantations needed - and provide examples and helpful hints for passing parameters when the defaults don't provide the needed support.

LT043 IMPROVING PERFORMANCE IN VAX PASCAL APPLICATIONS

Becky Will Digital Equipment Corporation

Programs that need to execute with optimal run-time efficiency must be carefully designed and written to require as few instructions as possible; the techniques used to compile them must take advantage of the machine's architecture. This session will describe the optimization techniques available in using the VAX PASCAL compiler such as inline expansion and disabling checking. It will also provide tips on how to take advantage of these optimizations in order to improve the efficiency of your Pascal programs.

LT044 VAX DEBUG TUTORIAL AND UPDATE

Bert Beander Digital Equipment Corporation

This session describes the capabilitities of the VAX/VMS Symbolic Debugger and shows how to use VAX DEBUG effectively to locate bugs to your programs. Breakpoints, examine capabilities, source display, and effective use of screen mode will be covered, among other topics. Various new features in version V4.4 will be discussed.

LT045 USING VAX PCA TO FIND PERFORMANCE PROBLEMS

Bert Beander Digital Equipment Corporation

This talk will demonstrate how you can use the VAX Performance and Coverage Analyzer to pinpoint various common performance problems in user-mode programs. Real examples from Digital's own development groups will illustrate how PCA has been used to achieve substantial performance gains in various Digital products.

LT046 THE VAX PERFORMANCE AND COVERAGE ANALYZER TUTORIAL

Bert Beander Digital Equipment Corporation

The features and capabilities of the VAX Performance and Coverage Analyzer will be described. This program development tool analyzes the performance of application programs to identify execution bottlenecks and other performance problems. It can also measure test coverage application programs.

LT047 THE VAX ADA (R) PROGRAMMING ENVIRONMENT

Joel Clinkenbeard Digital Equipment Corporation

An important part of the international Ada (R) effort is the concept of an Ada Program Support Environment, or APSE. The term "APSE" has acquired a variety of connotations, both technical and philosophical. This session will show how the combination of VAX Ada and VAX productivity tools provides an exceptionally strong Ada support environment. it will discuss how and why the VAX environment differs from the APSE described in the DOD Stoneman document. particular emphasis will be placed on program development using the program library manager component (ACS) of the VAX Ada product.

LT050 LANGUAGES AND TOOLS CLINIC

Katherine Hornbach Lear Siegler, Inc.

The Languages and Tools Clinic is an opportunity for novices and experts alike to ask questions about Languages and Tools, in an informal, one-on-one atmosphere. Digital developers will be on-hand, as will experts from the user community. Fair game are questions about almost any VAX language, plus CMS, MMS, Test Manager, PCA, LSE, TPU, and EDT. TeX and LaTeX experts will also be present. Both specific technical questions as well as general strategy and implementation questions are welcome. This session will be held in the relaxed atmosphere of the SIG Campground; stop by and take some time to go over your questions with the experts.

LT051 LANGUAGES AND TOOLS WIZARDRY

Katherine Hornbach Lear Siegler, Inc.

At this session, users have the opportunity to mystify and impress the audience with stories of their tool prowness and wizardry. Have you done something incredible with CMS, MMS, Test Manager, LSE, TPU, PCA or any of the other DEC tools (language tricks are acceptable also)? This is an opportunity to impress your peers and perhaps win a prize.

Stories of wizardry, magic, and war stories are all welcome. Stories can be humorous, but this is not required. Judging will be by a panel of expert tools users and DEC developers.

LT052 LANGUAGES AND TOOLS WISHLIST AND RESPONSE

Katherine Hornbach Lear Siegler, Inc.

During the first part of this year, a Wishlist ballot was distributed at Symposia and in the L&T newsletter, soliciting customer input on the relative importance of various requested language/tool enhancements and new products. Votes from hundreds of users have been tabulated, and the ``winners'' (top vote getters) will be announced at this session. A representative from Digital will be on hand to provide their responses to the top ten items in the Wishlist.

LT053 LANGUAGES AND TOOLS OPEN STEERING COMMITTEE MEETING AND USER FEEDBACK SESSION

Katherine Hornbach Lear Siegler, Inc.

During the Languages and Tools Open Steering Committee meeting, steering committee members will discuss current and planned SIG activities, and will solicit feedback and questions from the audience. Opinions and suggestions on sessions, seminars, campground activities, newsletters, and any other SIG activity are welcome. This is an opportunity to learn more about the Languages and Tools SIG and how it functions.

Part of the meeting will be devoted to recruiting new volunteers to help in SIG activities; if you are interested in becoming more involved in the L&T SIG, this is the session to go to!

LT054 PORTING VAX FORTRAN PROGRAMS BETWEEN VMS AND ULTRIX-32

Joel Clinkenbeard Digital Equipment Corporation

This session will describe the necessary technology to port VAX FORTRAN programs between VMS systems and ULTRIX-32 systems. Topics will include:

o Moving source code and data between ULTRIX and VMS.

o Language differences between VAX FORTRAN on ULTRIX and VMS.

o How to access system routines on ULTRIX and VMS for common application requirements.

o Where to find additional documentation.

o Examples of porting.

LT055 DEC ADA (R) PROGRAM UPDATE

Joel Clinkenbeard Digital Equipment Corporation

Representatives from Digital will discuss the current status of Ada (R) products at Digital: VAX Ada and VAXELN Ada. The talk will cover recently introduced capabilities and performance enhancements, validation status, and Digital's on-going efforts to address critical Ada customer needs.

(R) Ada is registered trademark of the Ada Joint Program Office

LT056 FORTRAN ENTOMOLOGY: FINDING AND FIXING FORTRAN BUGS

Joel Clinkenbeard Digital Equipment Corporation

Finding the source of problems in FORTRAN programs requires tools, experience, and common sense. A representative from Digital will discuss some common FORTRAN user errors gleaned from SPRs, FORTRAN conversion efforts, and internal experience. The session will cover how to interpret compile-time and run-time error messages and how to use tools like VAX DEBUG and VAX PCA. It will also cover practical ways to attack difficult bugs, such as unreproducible results and bugs in very large programs.

LT057 HOW DEC USES ITS TOOLS FOR SOFTWARE DEVELOPMENT -- A CASE STUDY

Technical Languages and Environments Digital Equipment Corporation

This session will walk users through an example of a development team working on a large system which consists of many integrated parts, where each member has a piece on which to work. The session will be given by a developer on a Digital software project, giving the developer's point of view of how tools were used and where they were beneficial. The benefits of tracking, debugging, etc., which you have while using expected extensions of the VAX Language-Sensitive Editor in software development will be itemized. This session will also show how to use other tools in an integrated environment on VMS as it extends its scenario.

LT058 EXTENDING YOUR ENVIRONMENT WITH THE VAX LANGUAGE-SENSITIVE EDITOR

Bev Schultz Digital Equipment Corporation

The VAX Language-Sensitive Editor is not used in a vacuum. It is most powerful when used in the context of an environment for users, and it has been built to optimize on the functionality for users, and it has been built to optimize on the functionality available on VAX/VMS. The VAX Language-Sensitive Editor integrates well with more than 8 VAX COBOL, VAX FORTRAN, VAX PASCAL, and VAX PL/I. The VAX Language-Sensitive Editor optimizes the user's work in the VAX Symbolic Debugger by allowing the debug session to call the editor to modify code while in the VAX Performance and Coverage Analyzer. This session details the use of the entire environment to make the most of the VAX Language-Sensitive Editor, and details other ways in which the user can make the most of his editing environment.

(R) Ada is a registered trademark of the Ada Joint Program Office

LT059 WHAT'S NEW WITH THE VAX LANGUAGE-SENSITIVE EDITOR

Gary Delong Digital Equipment Corporation

The VAX Language-Sensitive Editor is a powerful multi-language, multi-window, screen-oriented editor specifically designed for program capabilities integrated into your overall program environment. It is "language-sensitive" in that it provides a description of the syntax of all the major VAX languages with its "library" of language-specific templates. The compiler interface of the VAX Language-Sensitive Editor makes it possible for you to compile source code from within the editor in the same editing session in which it was written, and then to easily review your errors while examining the associated source code. This session will provide a tutorial of the VAX Language-Sensitive Editor, with emphasis upon the new extensions which make it even more powerful in the world of VAX/VMS.

LT060 VAX LANGUAGE-SENSITIVE EDITOR'S WIZARDS' NOTES

Gary Delong Digital Equipment Corporation

The VAX Language-Sensitive Editor has been designed to be a highly integrated member of the VAX/VMS software development environment. In addition to its language-sensitive editing features and its ability to assist in error correction, it works directly with the VAX Symbolic Debugger and with the VAX associated language compilers. This concentrate on the ties that the Editor gives you to your VMS environment. It will describe advanced capabilities of the Editor and will show how you can create your own templates in addition to the language templates that exist with the Editor. This can give the user-language support for languages that are specific to your company [Jovial...]. Also covered will be ways of extending and reformatting the VMS language support available with the Editor to handle subroutine packages and common conventions that are unique to the needs of your company.

LT061 ANALYZING SOURCE CODE

Bev Schultz Digital Equipment Corporation

The need for interactive cross-referencing and for source code analysis while doing software development is itemized in this session. The ability to give access to source data for entire systems can be extremely useful during the implementation and maintenance phases of a project. Using such facilities to understand the complexities of a system is detailed. Integration of such facilities with the VAX Language Sensitive Editor will be itemized.

LT062 C LANGUAGE STANDARDS UPDATE

Art Bjork Digital Equipment Corporation

This talk will look at the emerging ANSI C Programming Language Standard which is expected to be available for formal public review in July 1986 and will have a major impact on both users and implementors of the C language. This paper will look at major changes to the C language mandated by the ANSI standard and will look at how VAX C will have to be enhanced or modified to conform to this standard.

LT063 VAX C PRODUCT UPDATE

Art Bjork Digital Equipment Corporation

This will give an overview of the VAX C product including the compiler and run time support library. It will cover the current state of the product as well as any changes and enhancements that have been made in the most recent release of the product. This talk will be suitable for first time users of VAX C and for experience VAX C users.

LT064 FORTRAN 8X AND OTHER ANSI STANDARDS UPDATE

Jay Wiley Bechtel Power Corporation

This session will discuss the current status of the ANSI Standards of interest to Language and Tools SIG members. Special attention will be paid to the FORTRAN 8X standard activities.

LT065 USING DEC/CMS AND DEC/TEST MANAGER - A USERS EXPERIENCE

Jay Wiley Bechtel Power Corporation

This session will discuss the use of DEC/CMS and DEC/DTM in managing a (FORTRAN) development project. How these tools were used will be discussed. Example command procedures and naming conventions will also be presented.

LT066 USING MODULA-2 AS AN EFFECTIVE PROGRAMMING TOOL

Erich Stocker IIT Research Institute

Modula-2 has most of the advantages of Ada but is much smaller and more streamlined. As a language, it supports all major structured design and coding tools. It allows a great deal of tailoring which facilitates the coding process and makes it possible to move easily from design to code. It allows co-routines and low level system accesses. This paper will deal with these issues in the design of a high level file management system on a BSD V4.2 Unix-Cambridge Module-2 compiler.

LT067 SOFTWARE METRICS: AN INTRODUCTION

Charles A Lo Presti Battelle, Pacific N.W. Labs

This session will cover quantitative methods for measuring software characteristics. Metrics have been developed to measure scope, size, cost, risk, reliability, and elapsed time for software projects. The speaker will introduce the audience to some current methodologies and software metrics. Emphasis will be placed on metrics useful for project resource requirements estimation. The audience expected would include individuals charged with project planning, software engineers, and systems analysts.

LT068 A GENERALIZED CODING STANDARD AND SOME ASSOCIATED TOOLS

Eric J Straub Battelle, Pacific N.W. Labs

A general coding standard was developed to provide a neat, uniform, and complete form of documentation at the unit (or routine) level, making each routine self-documenting. The standard is based on some concepts from the Modula-2 programming language and may be applied to any language which allows inline documentation. We have currently implemented the standard on FORTRAN, C, DCL, and Datatrieve. The standard is compatible with CMS, but CMS is not required.

To support the coding standard, several software tools were developed. These tools include code to: 1) put a Standard template on a new or existing file; 2) copy the Definition blocks from a group of files to a single output file. This builds a user's manual for the routines.

In addition, language specific tools: 3) generate a list of subroutines and functions called by a FORTRAN routine, sorted by the package (or library) in which the called routines reside. The list consists of FORTRAN type definition statements describing the subroutines; and 4) generate a list of all variables used in a FORTRAN routine which have not been explicitly typed. The list is in a form which may be included in the file to explicitly type the variables. Approximately 20 people have been using this coding standard in conjunction with peer review for almost a year.

LT069 SMG AND FORTRAN: A CASE STUDY

Patrick Woods Ford Aerospace

This session is a case study of a FORTRAN application that used the screen management facility (SMG$) of the VAX/VMS runtime library. To implement a user interface. The FORTRAN application is the space telescope project database support system. This application used the SMG facility to partition any VT-100 compatible terminal screen into multiple windows and it used the SMG facility to perform all I/O in these areas. The focus of the session is on the advantages and disadvantages of using the SMG facility in a FORTRAN application.

LT071 ARCHITECTURE OF A SUCCESSFUL SOFTWARE ENGINEERING ENVIRONMENT

Robert Lanphar Hughes Aircraft Company

This session deals with an ensemble of techniques which have evolved over time to boost the quality production of software deliverables on large projects. A historical perspective will be provided to allow for hindsight evaluation of our activities. This session will be of particular interest to those future software engineering environment architects.

LT072 USING VAX SCAN TO IMPLEMENT LANGUAGE PROCESSORS

Jeffrey Boes Lear Siegler, Inc.

VAX SCAN is a programming language designed to implement tools which manipulate text strings or patterns. The patterns it can recognize are very similar to that for describing BNF grammars, such as those of a programming language.

Developing a SCAN application which processes a programming language is a task not unlike designing a compiler. SCAN programs were written to parse two different languages (JOVIAL and an internally-developed language). This talk will describe the benefits and drawbacks of using SCAN for large parsing applications.

Recommendations for those who would develop large SCAN applications are presented.

LT073 A TUTORIAL INTRODUCTION TO THE ICON PROGRAMMING LANGUAGE

Ken Harris Unico Inc

Icon is a high-level, general purpose programming language especially suited for text processing applications. It provides a very effective tool for quick solutions to "one-shot" text processing problems. This session will be a tutorial introduction to Icon. The speaker will present an overview of the Icon language followed by an analysis of several example programs.

The goal of the session will be to provide the beginner with enough background to begin using Icon for small tasks. No previous experience with Icon is assumed.

LT074 MIGRATING FROM RUNOFF TO TEX

Portia Bjorndahl Hughes Aircraft Co

This is a presentation on how documents are converted and migrated from using DEC standard RUNOFF to TEX. Problems encountered and solutions. A program which automates most of the tasks will be presented.

LT075 INTRODUCTION TO VAX TPU AND THE EVE EDITOR

TPU Developer Digital Equipment Corporation

VAX TPU is a high performance, programmable text processing utility distributed with the VAX/VMS operating system. VAX TPU contains a high level programming language, a compiler, and two editing interfaces; EVE, the extensible VAX editor, and an EDT keypad emulator - both written in the VAX TPU language.

This session provides an introduction to the concepts and capabilities of VAX TPU and the EVE editing interface. Ways to customize the EVE interface will be discussed.

LT076 PROGRAMMING IN VAX TPU

TPU Developer   Digitial Equipment Corporation

VAX TPU provides a high-level programming language that allows you to create your own text processing utilities. This session will discuss the basics of the VAX TPU language and how you can use it to extend and modify TPU-based editors.

LT077 ADVANCED TPU PROGRAMMING

TPU Developer   Digitial Equipment Corporation

VAX TPU is a high performance, programmable text processing utility distributed with the VMS operating system. It was designed as a tool to aid programmers in the development of text-orientied interfaces. VAX TPU has been used as the basis for sever VMS layered products, including VAX Notes.

This session covers more advanced topics in using the VAX TPU programming language. Topics to be covered include using TPU patterns, suing the CALLUSER built-in and building your own utilities on VAX TPU.

LT078 DIGITAL'S INTERNAL DOCUMENTATION TOOLS

Linda Storm   Digital Equip Corporation

The documentation set for VAX/VMS version 4.0, as well as various VMS layered products and languages, was typeset using an in-house documentation production system. This system was designed and developed at Digital to address and solve problems unique to the development, writing, and production of technical documentation.

This session will summarize the requirements for software documentation for VMS software products, including the documentation development cycle; describe the markup language and the tools developed to produce this documentation; and solicit user input on their requirements for documentation and production tools.

LT079 DOCUMENTATION TOOL CRYSTAL BALL

Linda Storm   Digital Equipment Corporation

This session will take a look at the impact that new technology has had on documentation tools and the document production life-cycle. We will attempt to look into the future regarding document processing and the impact that new technology will have on document production. An overview of tools available today will also be overviewed.

LT080 EXTENDED COMPUTER CONFERENCING SYSTEMS

VAX Notes Developer   Digital Equipment Corporation

An off-the-shelf computer conferencing system is a very powerful organizational tool, but could be even more effective if it were integrated into one or more key application systems. This session describes the callable interfaces to VAX NOTES, from the view of someone needing to integrate conferencing with an application suite.

LT081 PRACTICAL TIPS FOR COMPUTER CONFERENCES

VAX Notes Developer   Digital Equipment Corporation

This session provides practical hints for managing computer conferences, especially in distributed processing environments. We have used VAX Notes within Digital for several years, and have learned a number of do's and don'ts for conference moderators - and for software system managers. This session describes some of those learning experiences.

LT082 MEET THE WYSIWYG VS. MARKUP PANEL

Sam Whidden   American Mathematical Society

Text formatting philosophies: what are the choices? "What-You-See-Is-What-You-Get" or a markup language? An easy-to-manipulate, visual, perhaps rigid-format word processing approach, or a sometimes more difficult to learn but flexible embedded-command language? Both these modern, major text-formatting philosphies have their ardent proponents. Four of the most articulate and well known of these have formed a panel to challenge the issues surrounding these two approaches to document preparation. Session LT002 presents their panel discussion, allowing some time for audience participation. But the discussions evoke high feelings and sharp reactions, and the half hour allotted for questions and answers is not likely to be enough. So at the end of the panel session, Leslie Lamport, author of LaTeX, Brian Reid, author of Scribe, Mark Bramhall, DEC's VAX/TPU Project Leader, and Larry Bohm, Interleaf's Director of Product Management, will move from the session room to the Languages and Tools campground where all can join them. in an informal, open-ended debate on this subject of growing interest.

LT083 GRAPHICAL EDITORS IN AN INTEGRATED SOFTWARE DEVELOPMENT ENVIRONMENT

Harry Hazzard Bechtel Power Corporation

Graphical Editors in an integrated software development environment now provide support for several widely used analysis and design methods including Structured System Analysis, Structured Design, and User Software Engineering with extensive consistency checking built into the tools. Included as support tools are; Data Dictionary system which can be edited directly from the graphical editors that support definition of names, along with types, constraints, aliases and associated text; compact and efficient relational database management system to support a project database and Data Dictionary.

The major benefit is that cost of software development is sharply reduced. You can obtain a big leap in productivity with some of the basic tools and the jump is even greater when you use the graphical tools on a high performance work station.

LT084 INTRODUCING TEX AND LATEX AT NJIT

Bill Cheswick New Jersey Institute of Tech

This session discusses the problems of introducing TeX and LaTeX to a community of users, and the solutions we have found at the New Jersey Institute of Technology. Problems include hardware and software selection and training problems. Samples will be available.

LT086 FROM OUT OF THE WINGS.... PDP-11 FORTRAN-77

PDP-11 Languages Digital Equipment Corporation

The long awaited time has come. PDP-11 FORTRAN-77 V5.2 is here.

o What is it?

o What is included?

o Why were these features chosen?

o What does it do for me?

These and other questions will be addressed in this session.

So this version doesn't meet your needs, don't give up. Peek up our sleeve and see what is possible in the future.

Still not satisfied? Then tell us by answering the questionnaire at the session.

LT087 USING PDP-11 SYMBOLIC DEBUGGER

PDP-11 Languages Digital Equipment Corporation

Debugging a program is an art, not a science. Finding a bug in a program requires a firm knowledge of the program's workings, a little intuition, and a little luck. Some tricks of the trade in finding problems in FORTRAN-77, COBOL-81, and MACRO-11 programs using the PDP-11 Symbolic Debugger include:

o Losing my mind (or how to find infinite loops)

o What's going on here (or uses of smart breakpoints)

o WHEN did you say I should do it? (the use of these clauses in breakpoints)

o I'm not gonna type that AGAIN (or where you should use indirect command files)

o My files are gone (or how to examine FABs, RABs, and other ugly data structures)

o No lo comprendo (or language specific problems like COBOL's STATUS variable)

- What you've got here is a classic hardware problem (or problems with I- and D-space)
- Pretty Pictures on your PRO (how to send debugger output out the printer port)
- Running away from the problem (or debugging across the network)

LT088 FORTRAN BENCHMARKS

Robert Walraven Multiware, Inc.

A number of small scientific FORTRAN benchmark programs were run on a wide variety of Digital processors and some non-Digital systems. Results of these runs will be presented.

LT093 USER INTERFACE MANAGEMENT SYSTEMS: A NEW GENERATION OF PRODUCTIVITY TOOLS FOR SOFTWARE DEVELOPMENT ON THE VAX

Gilbert Cardwell Precision Visuals Inc

Recent industry conferences sponsored by SIGGRAPH, Eurographics, and IFIPS have identified the improvements in productivity and in the quality of user interfaces which can result from development and use of a new generation of software tools call User Interface Management Systems (UIMS). This presentation offers an introduction and guide to these new tools. A general model of User Interface Management Systems is presented, the benefits developers can expect to obtain from this class of tools is reviewed, and suggestions are made for evaluation of UIMS features, performance, and design.

LT094 AUTOMATED AND RELIABLE SOFTWARE DEVELOPMENT AND DELIVERY USING DIGITAL SOFTWARE PRODUCTS

Glenn Cooley Survey Sampling, Inc.
Panel Member Goldman, Sachs and Co.

Presented are the details of a software development and delivery environment which makes innovative use of the Digital software products LSE, CMS, MMS, and DECnet along with a concept called "tiered releasing", to reduce the time demands on software developers through extensive automation and increase software reliability. In this environment, software developers are better able to concentrate on the creative aspects of software development since many of the non-creative steps are automated. Similarly, automating these steps has increased reliability since far more data can be maintained on the software developed and delivered with much less reliance on user input accuracy and completeness.

LT095 TOOLING UP FOR PROJECT DEVELOPMENT - THE USE OF SOFTWARE TOOLS IN A VMS ENGINEERING PROJECT ENVIRONMENT

Greg Clemens Goldman, Sachs and Co.

This session will be a panel discussion of the software tools and software engineering techniques used by the Goldman Sachs Financial Strategies Group to build medium and large scale software systems. Topics will include a history of the group and how the "Tools Team" was put together; productivity requirements and the rational for tool selections; and the success and problems with using CMS, MMS, PCA, Notes, Mail, TeX, and the other tools used by the group.

LT096 MAKING ALL THOSE PRODUCTIVITY TOOLS BE PRODUCTIVE TOGETHER

Cindy McDonnell-Feinberg Digital Equipment Corporation

On VAX/VMS systems, Digital provides quite a number of "productivity tools" -- LSE, CMS, MMS, PCA, DTM. Their very number and apparent complexity can make them seen difficult to use, and some sites shy away from them for that reason.

This session will use real examples of application development to address the issues of integrating the use of these tools to be more productive, including the issue of time management.

Specifically, how does the Module Management System (DEC/MMS) interact with the Code Management System (DEC/CMS)? How can the two of them be used harmoniously with the DEC/Test Manager (DTM)? And where in the iterative cycle of making well-tuned, working software does the Performance and Coverage Analyzer (PCA) fit in?

The tasks of application development and maintenance are enormous, but not insurmountable! The hints and suggestions from this session will help ease your way through those enormous tasks.

LT097 GUIDED TOUR THROUGH AN EMACS EXTENSION: DIRED

Peter Kaiser Digital Equipment Corporation

EMACS is a popular extensible editor for VAXes and other computers; in several versions it may be the single most popular tool that crosses such wide boundaries of architecture and operating systems. Much of its power comes from giving the user the ability to write "extensions" that perform useful functions (which may be quite different from simple editing).

This talk is a guided tour through the code and methodology of my own version of DIRED, an extension that allows the user to manipulate groups of files -- examining, modifying, deleting, and printing them -- in a particularly powerful way. The user sees a list of files on the screen, and with a single keystroke can select an action for one of them, or in some cases, for a group of them at a time. The talk covers the possibilities in extending DIRED. It is applicable in general to questions of how to write extensions to extensible editors, including TPU.

This is an improved version of a similar presentation given at the DECUS Symposium of Fall 1985.

LT100 DEBUGGING COMPILERS AT WATERLOO - THEN AND NOW

David Yach WATCOM Products Inc.

WATFOR, WATFIV and WATFOR-11 were FORTRAN compilers developed at the University of Waterloo in the 1960s and 1970s. The objectives of these compilers included fast compilation speed and effective error diagnostics both at compile time and at execution time. They eliminated the need for a separate linking procedure and as a result, FORTRAN programs which contained no syntax errors were placed into immediate execution. Non-experience programmers could be taught programming at minimal cost in time and computing resources. Experienced programmers quickly saw the benefits of good diagnostic capabilities and fast turn-around. Since these compilers were developed, all aspects of computing have changed tremendously. This talk will discuss how several of the goals of these early compilers are still applicable today and how these goals have been addressed in the development of new compilers at Waterloo for student use.

LT101 USING THE CMS CALLABLE INTERFACE

G Del Merritt Computer Sciences Corporation

A discussion on the advantages of using the DEC/CMS callable interface vs. the CMS Monitor. Working knowledge of VMS system services or RTL assumed.

LT102 PUBLIC DOMAIN LANGUAGES AND TOOLS SOFTWARE

Anthony E. Scandora, Jr. Argonne National Lab.

Public Domain software panel -- abstract coming Real Soon

LT103 PDP-11 C USERS FORUM

Anthony E. Scandora, Jr. Argonne National Lab.

Do you write C programs for the PDP-11? How is your support? Have you written a useful subroutine package that is not in the distribution kit? Have you fixed compiler or library bugs? Do you need subroutine packages that are not in the distribution kit or bug fixes? A panel of long-time C users will discuss some of their problems with, enhancements to, and fixes to both Whitesmiths and DECUS C for the PDP-11. Audience participation will be encouraged, and there will be plenty of time for questions.

LT104 PDP-11 LANGUAGES AND TOOLS QUESTION AND ANSWERS

Anthony E. Scandora, Jr. Argonne National Lab.

This session will provide PDP-11 users on all operating systems the opportunity to get answers to questions on languages and tools. Digital developers of FORTRAN, DEBUG, PASCAL, BASIC, DATATRIEVE and other products hosted on a PDP-11 will be available. Experts on popular DECUS software including C, RUNOFF, TECO, and the Software Tools will also be available on the panel.

LT105 PDP-11 / VAX COEXISTENCE AND MIGRATION ISSUES

Anthony E. Scandora, Jr. Argonne National Lab.

Let's take this opportunity to tell Digital what our concerns, needs, strategies, etc., are in relation to PDP-11 and VAX systems' coexistence and migration. There were initial discussions at the previous symposium and this will be your opportunity to add to it to make sure your input is heard.

In particular, what tools and support do you need in this area? What, if anything, exists that has helped you so far? What can Digital do to extend those things, and what would have helped in your previous efforts?

If this area is important to you, we urge you to attend and participate.

LT106 HOW VMS DEVELOPMENT USES CMS

Dick Mahoney Digital Equipment Corporation

This talk will describe how and why the VMS development Team at Digital uses VAX DEC/CMS for source in their development process. Topics to be covered include: why CMS was chosen, how we use various capabilities in CMS, problems encountered, and future needs and requirements.

LT107 SOFTWARE CONFIGURATION MANAGEMENT

Dick Mahoney Digital Equipment Corporation

VMS Development is pursuing possible tools to assist customers in managing their software configurations. Areas of interest include product installation, installation history, system manager control, product to product interactions, license management, usage statistics, and cluster considerations. The scope of software products covered includes Digital produced software, 3rd party software, and customer software. This talk will discuss directions we are taking, as well as progress we have made to date.

LT108 FILING SOME HOLES IN THE VAX-C RUNTIME LIBRARY - THE "SYSTEM" FUNCTION AND RELATED SUPPORT ROUTINES

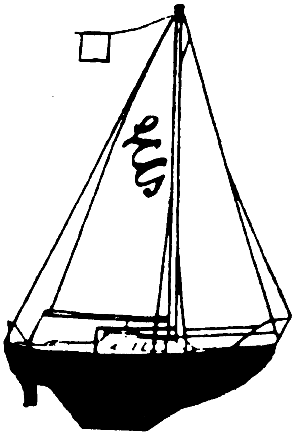
Wayne Baisley Rockwell International

The VAX-C V2 run-time library provides VMS support for a large subset of the UNIX (tm) system support routines. One notable exception is the "system" function, which passes a command line to a shell of command line interpreter (probably the easiest way to make any utility "callable"). This paper describes an implementation of the "system" function using VAX-C. Also described are a number of related and general support routines, and header files.

DECUS

LARGE SYSTEMS SIG
DECUS
U.S. CHAPTER

DIGITAL EQUIPMENT COMPUTER USERS SOCIETY



AT LARGE

LS

The Newsletter of the Large Systems SIG

SIG STEERING COMMITTEE

SIG Chariperson
Leslie Maltz
Stevens Institute of Technology
Computer Center
Hoboken, NJ 07030
(201) 420-5478; BITNET:LMALTZ@SITVXB;
ARPANET:SIT.MALTZ@CU20B.COLUMBIA.EDU

Symposium Coordinator
Robert C. McQueen
Stevens Institute of Technology
Computer Center
Hoboken, NJ 07030
(201) 420-5454; BITNET:RMCQUEEN@SITVXB;
ARPANET:SIT.MCQUEEN@CU20B.COLUMBIA.EDU

Newsletter Editor
Clyde T. Poole
The University of Texas at Austin
Department of Computer Science
Taylor Hall 2.124
Austin, TX 78712-1188
(512) 471-9551
ARPANET:ctp@sally.utexas.edu

Menu Coordinator
Charles R. T. Bacon
National Institutes of Health
Building 12B Room 2N207
Bethesda, MD 20205
(303) 496-4823
BITNET:CRB@NIHCUDEC

Hardware Coordinator
Clive Dawson
Microelectronics & Computer Technology Corp.
9430 Research Blvd.
Echelon Bldg. #1, Suite 200
Austin, TX 78759
(512) 343-0860
ARPANET/CSNET:CLIVE.MCC

Languages Coordinator
David Edwards
SRI International
MS PN349
333 Ravenswood Ave.
Menlo Park, CA 94021
(415) 859-6136

Systems Software Coordinator
Betsy Ramsey
American Mathematical Society
P.O. Box 6248
Providence, RI 02940
(410) 272-9500 ext. 295
ARPANET:EWR@XX.LCS.MIT.EDU

Special Projects Coordinator
E. F. Berkley Shands
Washington University
Department of Computer Science
P.O. Box 1045
St. Louis, MO 63136
(314) 889-6636
BERKLEY@WUCS.UUCP

Networks Coordinator
Don Kassebaum
Computation Center
University of Texas at Austin
Austin, TX 78712
(512) 471-3241
ARPANET:CC.KASSEBAUM@A20.CC.UTEXAS.EDU

Systems Software Coordinator
Carla Rissmeyer
Computation Center
University of Texas at Austin
Austin, TX 78712
(512) 471-3241
ARPANET:CC.RISSMEYER@A20.CC.UTEXAS.EDU

DEC Counterparts
Dave Braithwaite
Digital Equipment Corporation
Marlboro, MA

Rich Whitman
Digital Equipment Corporation
Marlboro, MA

Reed Powell
Digital Equipment Corporation
Marlboto, MA

CHAIRPERSON'S ARTICLE

Leslie Maltz

It is hard to believe that we are about to leave for the next symposium already. Much has happened during recent weeks both in preparation for the symposium as well as in planning for future activities of the SIG. A separate article included in this issue describes many of the activities that will happen in San Francisco, so I won't be redundant. Instead I will simply summarize that we will be sponsoring sessions related to the care, feeding, and management of all of Digital's high-end systems. In addition we are sponsoring sessions of interest to installations that would be considered Information Centers. Our variety of topics covers the traditional types of sessions held at prior symposia, and many sessions related to the management of high-end VAX systems in the multi-vendor environment. We are participating in the support of the theme orientation in view of its appropriateness for our members. The theme is "The Network is the System; the Next Generation of Computing Resources".

For those who are not familiar with our SIG, we have as our focus the issues and concerns associated with the management, support, and use of Digital's high-end computing systems. A recent meeting of the SIG leadership with Digital was held to discuss future directions for all of us. We are in a position to be of service to existing and new customers with Digital's high-end systems, and are making plans to expand those services to meet the needs of the future. I believe that we have reasonable expectations as well as a good understanding of Digital's directions and how they map into our needs as customers. You will start to see some of these new SIG sponsored products and services shortly. Our expanded schedule of sessions in San Francisco is just an indication of this expansion of activities.

This month's issue of the newsletter includes a copy of our SIG MENU. This is our balloting process to identify your needs and priorities in a quantifiable form. Please identify your priorities and return your ballot to us immediately. The results will be tabulated and shared with all, and in particular with Digital as an indication of the needs of the customers. You do not need to be a member of this SIG in order to respond. All we ask is that you have an interest in Digital's high-end computing systems. So get your ballot in to us as quickly as possible to have the maximal impact.

We'll keep you posted via the daily symposium newsletter (Update.daily) of our focus of the day and any special activities that you won't want to miss in San Francisco. See you there.

At San Francisco

The Large Systems SIG will be holding a full five days of sessions at the San Francisco DECUS Symposium. Additionally, the SIG will be presenting two Pre-Symposia Seminars. One of the seminars is devoted to the TCP/IP protocols, which are the basis for several national networks (NSFnet, ARPANET, etc.), and the other is devoted to TOPS-20 System Management. It will be possible at San Francisco to register for the Pre-Symposia Seminars when you arrive. This is the first time that you can register on-site for the seminars, but there will be an additional charge for on-site registration.

The Large System SIG sessions are grouped into five basic areas of interest: VAX Information Center, DECsystem-10/20, High-end VAX systems, Conversions/Migrations and SIG business sessions. The week starts off with the SIG Roadmap and Direction sessions, followed by the a set of Information Center sessions. The end of Monday and all Tuesday is dedicated to the DECsystem-10/20 base, with Digital starting Tuesday off with the Marketing and Engineering Update. The remainder of the week is devoted to the high-end VAX system/cluster sessions and TOPS conversion sessions. The SIG Business Meeting will be held at 5:30pm on Thursday. The following is a complete run down on the sessions that the SIG is offering.

Monday

09:00 - 09:30 LS008 Large Systems SIG Roadmap
09:30 - 10:00 LS027 Large Systems SIG Directions
10:00 - 11:00 LS001 Estimating & Project Management of a Large System's Development Project
11:00 - 12:00 LS014 Architecture for a VAX Information Center
12:00 - 13:00 LS015 Strategies for Information Center Success
13:00 - 14:00 LS017 Information Centers: Managing the needs of Corporate Management
14:00 - 15:00 LS018 Information Centers: Managing the Needs and Requirements of the End Users
15:00 - 16:00 LS016 Information Centers: Implementation and Successes a Panel Discussion
16:00 - 17:00 LS029 TOPS Software Development: Life After DEC

Tuesday

09:00 - 10:30 LS005 DECsystem-10/20 Marketing and Engineering Update
10:30 - 11:30 LS007 TOPS-20 System Update
11:30 - 12:30 LS041 TOPS-20 EMACS Magic
12:30 - 13:30 LS040 TOPS-10/20 FORTRAN v11 Technical Update
13:30 - 14:30 LS006 DECsystem-10 System Update
14:30 - 15:30 LS042 Internet Domain Service
15:30 - 16:30 LS004 Fun with TOPS-20 - Customizing Your EXEC
16:30 - 17:15 LS028 TOPS-10/20 Utility Closet
17:15 - 18:00 LS002 TOPS-10/20 Guru's Get-Together

Wednesday

09:00 - 10:00 LS003 High-End VAXcluster Performance
10:00 - 11:00 LS019 Directions in High-End VAXcluster System Performance Management
11:00 - 12:00 LS025 VAXcluster Systems as an Alternative to Traditional Mainframes
12:00 - 13:00 LS023 High Availability Fault Tolerant Compting Systems "How Do I know if I need one"
13:00 - 14:00 LS024 High Availability Considerations for VAXcluster Systems
14:00 - 15:00 LS044 Disk Space Management at One Large VAXcluster Site
15:00 - 16:00 LS046 Large Disk Farm Management
16:00 - 17:00 LS043 User Account Management at one Large VAXcluster Site
17:00 - 18:00 LS045 Large Tape Farm Management – Myth?

Thursday

09:00 - 10:00 LS037 VAX/VMS Hardware/Software Product Update for High-end Users
10:00 - 11:00 LS026 High-End VAX System Positioning
11:00 - 12:00 LS011 Configuring High-End VAX Systems

12:00 - 12:30 LS013 An Academic Conversion from a DECsystem-10 to a VAX-8650
12:30 - 13:30 LS039 TOPS to UNIX: A Panel Discussion
13:30 - 14:30 LS031 Planning and implementing a Large Network
14:30 - 15:30 LS030 Managing High-End Systems in a Multi-Vendor Environment
15:30 - 16:30 LS038 Digital Systems in Multi-Vendor Environments: Networking Coexistence
16:30 - 17:30 LS010 Large Systems SIG Town Meeting
17:30 - 18:15 LS034 VMS for TOPS: System Administration, Management, and Operations
18:15 - 19:30 LS033 VMS for TOPS: The VMS User Command and Programming Interface

Friday

09:00 - 10:30 LS032 TOPS to VMS Conversions: Tools and Methodology for Converting Higher Level Languages
10:30 - 11:30 LS035 TOPS to VMS Conversions: Database Applications
11:30 - 12:30 LS036 TOPS to VMS Conversions: Tools for Converting Languages, Applications and Users

1986 Large Systems SIG Menu Introduction

This is the official Spring 1986 Menu of the DECUS Large Systems SIG. It is based on issues identified at the Symposium in Dallas. We hope that you will participate by indicating your priorities for issues identified. You need not be a member of the SIG in order to vote. Users of any of Digital's high-end systems (DECSYSTEMS and high-end VAX systems and clusters) are encouraged to participate. Many new members of the SIG interested in high-end VAX systems and are encouraged to vote for those items of concern to their sites.

The results of the Menu will be presented to Digital as an indication of the needs of our installations and the direction we wish Digital to pursue in future development. The results of the balloting will be available to all members of DECUS and will be published in the Large Systems SIG section of the newsletter. The results will also be presented at the Fall Symposium in San Francisco.

The Menu consists of three parts. The first part contains items suggested by TOPS-10 users. The second is a TOPS-20 section. The third section contains VMS issues. The third section is nearly identical to a selection of SIRs submitted by members of the VAX SIG. It is in the format of the VAX SIG's SIR (Software Improvement Request), which traditionally has enjoyed a fuller explanation of each item than has been the case with the Large Systems SIG Menus. To have the fullest potential influence on Digital's development plans, we have coordinated efforts with the VAX SIG on the drafting of items in this section. Sites running or planning to run high-end VAX systems are encouraged to respond to this section.

Each item in this menu has a number, a title, and some text. The text is meant to provide a description of the item, and does not represent a position taken by the Large Systems SIG. It is included for explanatory purposes only. The ballot lists the items in the same order with the same descriptive titles for ease of voting.

We ask that each voter restrict votes to a maximum of five votes per item, and a maximum of twenty-five votes per section. Votes may be marked with a minus sign, indicating opposition to an item rather than support.

Ballots should be completed and mailed as soon as possible. Remember that you need not be a member of the SIG in order to respond. The ballot may be removed from the newsletter, or you may photocopy it.

All ballots must be returned by October 1, 1986. Please mail ballots to:

Charles Bacon
N. I. H.
Bldg. 12B, Rm. 2N207
Bethesda, MD 20892
Attn: L. S. SIG Menu

Category 1
TOPS-10 Menu Items

The following TOPS-10 Menu items were collected at Dallas. With the end drawing near for the active development cycle of TOPS-10, a limited number of Menu items have been submitted. Generally those which have been submitted consist largely of pleas for stability and maintainability, and almost no new functionality. Nevertheless, there are matters of importance to TOPS-20 users here. Items that apply equally well to TOPS-20 are also listed in the TOPS-20 section. Please vote for items which seem the most important. Allocate 25 votes for the whole TOPS-10 section, and no more than 5 votes for a single item. Negative votes may be cast for items actively opposed, but count positively toward the total.

1.1 Command Line Recall with Editing

To be worthwhile, the feature should be capable of holding the last ten or more monitor-level commands from the current context.

1.2 Fix Disk Error Recovery and Reporting Code

The implication is that current code for disk error recovery does not do what is claimed for it, and needs to be fixed. Disk error reporting likewise should be improved.

1.3 Speed up File I-O

I/O on TOPS-10 is currently suffering from too many different generations of hardware to support. This item would require that the the I/O structure be examined to determine what can be done to improve the performance of the I/O system. This includes taking a look at the queued I/O processing on SMP systems, construction of I/O lists, etc.

1.4 Create a SET POLICY Command

This would be preferable to dropping a CPU out of an SMP configuration and bringing it back, when the operator wants a particular CPU to be the Policy CPU.

1.5 Fix RSX20F Ctl-S/Ctl-Q

RSX20F should be able to handle Ctl-S and Ctl-Q properly on the console terminal line. Modern terminals capable of 240 CPS speeds abound, and there is some penalty in operating a system with a slow CTY.

1.6 Provide Support for Extended Addressing in FORTRAN

This is an important item because we realize that TOPS-10 users of the late 1980's will simply not avail themselves of features accessible only through MACRO-10 coding. If user-mode extended addressing is to be of any utility at all, it must be provided in higher level languages such as FORTRAN-10.

1.7 Restrict 7.04 to RAMP Issues

The development in TOPS-10 7.04 should be restricted mainly to providing better performance, reliability and other RAMP issues. New functionality should be kept to a minimum in 7.04, so as to provide for a very stable operating system after Digital development ends.

1.8 Diagnostics for Tri-SMP

There are at present no diagnostics which will check out a tri-SMP system as such. Single-CPU diagnostics don't do the job.

1.9 Session audit trails

This would provide the ability to identify all programs run and all files touched during a session, without being a complete session log.

1.10 Continue development for Corporate Communications Protocols

TOPS-10 7.04 should have support for new Digital corporate communications protocols. This would include Local Area Network print services, DECnet and other protocols that will be available before the end of the TOPS development timeframe. It is important that TOPS-10 support these items to enable users to migrate to other Digital systems.

1.11 Get Rid of Dispatch: do it On Line

The DSIN (Digital Software Information Network) in Colorado Springs can perform the same function faster and cheaper.

1.12 Finish DECnet-10 Implementation of the DDP Device

The DDP device provides communication between a TOPS-10 program and a DN20/DN200 DDCMP line. It was never seriously completed, and could provide a good foundation for DECnet using the older 32K DN20s and DN200s.

1.13 Add AUTHOR Selector to the DIRECT Command

The command DIR/AUTH:[p,pn] should show only those files with the given author field. Similarly, /PROT:<xxx> should be able to select. As a general SCAN feature, many programs would automatically be enhanced.

1.14 Provide TCP/IP on TOPS-10

This would provide support for the basic DoD protocols (IP, ICMP and TCP) for the DECsystem-10 via the NIA-20 (Ethernet) device. This support would provide a means for users to have the DECsystem-10 communicate in a reasonable manner with UNIX based systems.

1.15 TOPS-10 Documentation in Machine-Readable Form

When development ends, Digital should provide these texts, so that users may continue maintenance. The source (.RNO) files should be included.

1.16 Make TOPS-10 sources available to all

When development ends, Digital should place TOPS-10 sources in the public domain so that non-Digital sites may continue development, and distribute their results to any interested party.

1.17 Provide a network-wide Galaxy

Provide a network-wide batch and spooling system. This would help users to migrate applications from the DECsystems to VMS systems.

1.18 Provide a Structured HELP Facility, a la VMS

1.19 Bring all TOPS-10 Languages up to Current ANSI Level

In particular, they should be at the same level as their VAX counterparts.

Category 2
TOPS-20 Menu Items

The following TOPS-20 Menu items were collected at Dallas. They are designated as TOPS-20 partly because of the way they were collected. Items that apply equally well to TOPS-10 are also listed in the TOPS-10 section. Remember, limit any one item to no more than five votes, and no more than 25 votes for the whole section. Negative votes may be cast for items actively opposed, but count positively toward the total.

2.1 Make TOPS-20 sources available to all

When development ends, Digital should place TOPS-20 sources in the public domain so that non-Digital sites may continue development, and distribute their results to any interested party.

2.2 Provide TCP/IP Support in TOPS-20 V6.1 Free of Charge

Digital should adhere to the decision they announced several DECUS's ago. Currently Digital wants the customer to pay a large sum for the TCP/IP monitor modules. Since much of the TCP/IP code was developed in cooperation with the Arpanet user community, it seems only reasonable that Digital bundle this code with TOPS-20.

2.3 Continue development for Corporate Communications Protocols

TOPS-20 7.0 should have support for new Digital corporate communications protocols. This would include Local Area Network print services, DECnet and other protocols that will be available before the end of the TOPS development timeframe. It is important that TOPS-20 support these items to enable users to migrate to other Digital systems.

2.4 TOPS-20 Documentation in Machine-Readable Form

When development ends, Digital should provide these texts, so that users may continue maintenance. The source (.RNO) files should be included.

2.5 Fix RSX20F Ctl-S/Ctl-Q

RSX20F should be able to handle Ctl-S and Ctl-Q properly on the console terminal line. Modern terminals capable of 240 CPS speeds abound, and there is some penalty in operating a system with a slow CTY.

2.6 Get Rid of Dispatch: do it On Line

The DSIN (Digital Software Information Network) in Colorado Springs can perform the same function faster and cheaper.

2.7 GTJFN to Provide Partial Recognition of Filenames

For example, with files FOOBAR and FOOBLY, F<ESC> displays FOOB. Some user sites have implemented this feature.

2.8 Allow VMS-like relative directory specifications

For example, <.MAIL> refers to the MAIL subdirectory of the current directory. Some user sites have implemented this feature.

2.9 Provide a Way to Determine Terminal Inactivity

Something along the lines of an IDLE% jsys. Some user sites have implemented this feature.

2.10 Implement Command Line Editing in the Monitor Itself

So that user applications could employ it. Perhaps this could be made part of the TEXTI% jsys so that user-written programs would have access to the feature.

2.11 Allow a Non-Privileged User to Check the Status

Allow a non-privileged user to check the status of another user's send/receive user (TTMSG%) messages bit.

2.12 Provide a network-wide Galaxy

Provide a network-wide batch and spooling system. This would help users to migrate applications from the DECsystems to VMS systems.

2.13 Make LNMST Reveal Another Job's Logical Names

Currently there is no way to get information on another job's logical names.

2.14 Allow COPY to Write More than One Record per Block

Allow COPY to write more than one record per block to an ANSI-labeled tape. Improve EXEC-level tape handling in general.

2.15 Option to GTJFN for logical names

If ALL: is defined to be A:,B:, ALL:*.* will find all files in both A: and B:.

2.16 Improve date and time handling

In particular, the monitor should default to the correct (current or previous) year if none is specified.

2.17 Implement Access Control List Facility

Implement access control list facility whereby files can have individual protections for each user group, rather than one protection code for all group accesses.

2.18 Make GTJFN Understand AND, OR, NOT and Parentheses

Make the GTJFN JSYS understand AND, OR NOT and parentheses as part of file specifications (like TOPS-10 SCAN and WILD).

2.19 Option to Make Any Structure obey PS: access rights

Provide option to make any structure obey PS: access rights-particularly for user groups.

2.20 Provide a Mechanism (JSYS?) to Allow Default Forms

Forms and/or unit number to be specified for print jobs spooled to LPT: (similar to TOPS-10 SPPRM. UUO).

2.21 Provide recognition input for structure names

2.22 Support a File Comment String in the FDB

It should be accessible through GTFDB% and the DIRECTORY-class commands.

2.23 Standardize Peripheral Interfaces (PHYSIO)

This should be done in such a way as to make it easier to use third party disks/tapes/ controllers.

2.24 Move All Terminal Support Out of the Monitor and EXEC

And move it into a UNIX-like termcaps file that would be used by all TOPS-20 software, and other application software. The user should be able to easily modify this file.

2.25 Implement UNIX-like Pipes in TOPS-20

2.26 Provide a Warm Boot Facility for the DEC-20

When the system goes down, preserve the process context so that the processes can be continued where they left off when the system comes back up.

2.27 Treat One-Page Files Specially

Don't bother maintaining a page table for them. Just construct it when the file is accessed.

2.28 Support 3- and 4-System Clusters

2.29 Support Cluster-wide IPCF

2.30 Support Cluster-wide Galaxy Queues

2.31 Support Cluster-wide ENQ/DEQ

2.32 Support for non-PS: Login

Allow login directories to be on a CFS structure.

2.33 Support for HSC-based Tape Drives (TA78)

2.34 Provide Domain Support for Arpanet

2.35 Provide Arpanet EGP support

2.36 Update the Arpanet TCP/IP utilities FTP, FTPSRT and IPHOST in particular.

2.37 Implement Arpanet TCP/IP Over Asynchronous Lines

2.38 Integrate DECnet Support into the EXEC

That is, support DECnet file access through EXEC commands (COPY, DELETE, PRINT, SUBMIT, DIRECTORY, etc.) rather than through the NFT program.

2.39 Support Intersystem IPCF Between DEC-20s Running DECnet

2.40 PRINT/DEST, OPR's ROUTE/NODE and Network Node Names In the QUEUE% jsys, these should work for DECnet-20 sites.

2.41 Improve Data Interchange Library (DIL) Performance

2.42 Allow DIRECTORY-like selection for the DELETE command

Especially BEFORE, SINCE, LARGER, SMALLER. Some VMS-like options would be nice, too, particularly EXCLUDE and CONFIRM.

2.43 Provide a LIST Subcommand to DIRECTORY-class Commands

So that the user can see what parameters he has specified thus far.

2.44 Make EXEC Features Assembled with XTND Available to all

These features include the extended DIRECTORY-class commands (WDIR, RDIR, QDIR). Source sites have been using these features for years, so there is little question about their reliability.

2.45 Allow ATTACH.CMD and DETACH.CMD

Some user sites have implemented this feature.

2.46 Prohibit Control-C during the taking of LOGIN.CMD

2.47 Provide for Forcing Periodic Password Changes

2.48 Provide an option to enforce minimum password lengths

Providing an option that allows a site to specify the minimum password length allows the site the ability to have greater security.

2.49 Make TOPS-20 MIC handle all of TOPS-10 MIC's Commands

2.50 Provide Official Support for PC

2.51 Provide a Structured HELP Facility, a la VMS

2.52 Provide an ACCOUNT Subcommand to SYSTAT

This would display the user's current account string.

2.53 Have the DISMOUNT Command Report Tape Errors

Let it also perform an automatic INFORMATION VOLUMES display.

2.54 Improve Integration of History and Command Line Editor

In particular, the RECALL command should be able to recall as many commands as can be edited by the command line editor, and the user should be able to edit a RECALLed line.

2.55 If GET% Fails Because the File Has Invalid .EXE Format

If GET% fails because the file has an invalid .EXE file format assume the file is a script of EXEC commands, and load an EXEC with .PRIIN pointing to this file.

2.56 Provide Support for TTY: Devices in LPTSPL

Currently, LPTSPL supports only LPT and MTA devices. Extend this support to TTY devices. Some user sites have implemented this change.

2.57 Implement a Lesser OPERATOR-type Privilege

One that will allow users to issue printer-related commands in OPR, but nothing else. Users with this privilege should be able to START, STOP, CONTINUE, SET PRINTER and SHUTDOWN printers.

2.58 Implement an INFORMATION QUEUES command

This command should list all Galaxy queues. INFO QUEUE /USER would list that user's jobs in all queues.

2.59 Wildcarding on Job and User Names in CANCEL and MODIFY

2.60 Bring all TOPS-20 Languages up to Current ANSI Level

In particular, they should be at the same level as their VAX counterparts.

2.61 Improve DUMPER performance

2.62 Add some TOPS-10 BACKUP-like functionality to DUMPER

The ability to position to a saveset based on name, not number; ability to compare files in a saveset to files on a different structure than the one from which they were saved.

2.63 Add a STATUS Command to DUMPER

One that would list the parameters the user has selected so far.

2.64 Let SYSDPY Display Programs Running in Non-Zero Sections

2.65 Support 132-column display in DPY/SYSDPY

2.66 Teach WATCH to be Careful about its Data File

Enhance it so that it allows its data file to be copied out from under it. If the file is renamed, WATCH should open a new file with the same name.

2.67 Support long filenames in FILCOM

2.68 The Autopatch Process is Still Too Complicated

Aim at making it more like VMSINSTAL. In particular, PEP/PEPB is too picky about checksums and batch completion flags and doesn't work well if it is run by more than one user.

2.69 Autopatch Should Provide .REL File Replacement

Autopatch should provide for .REL file replacement for all products for which the customer does not have a source license.

Category 3

VAX-related Menu Items

These items are transcriptions of VAX SIG SIRs, provided with one-line descriptions ior balloting purposes, and for consistency with the preceding TOPS-10 and TOPS-20 items.

Voting for or against these items should be undertaken as seriously as for the TOPS-related items. Digital is interested in learning the biases of the TOPS community regarding the VAX and VMS, relative to the original VAX SIG membership.

3.1 Implement a MOUNT queue in VMS

Enhance the ALLOCATE command to enable a user to optionally queue the allocation request when all qualifying devices are busy. Device allocation should be handled by a queue manager similar to the VMS V4.0 print queue manager, and the allocation request queues should be made cluster wide to support cluster-visible devices.

User functions should include the ability to specify characteristics required of a generic device, the automatic notification of allocation, the ability to delete an allocation request, the ability to examine the allocation request queue, and the ability to do other interactive processing while waiting for an allocation request to be granted.

Operator functions should include the ability to mark failing devices as unavailable and the ability to force a deallocate. Manager functions should include the ability to define device characteristics and specify physical devices as possessing those characteristics.

Device allocation and deallocation should place records in the accounting file so that charge back accounting can be done for allocated devices.

A mechanism for avoiding deadlocks when multiple devices are allocated should be provided.

```
$ ALLOCATE/QUEUED/WAIT TAPE$CLASS:-
  $_ /CHARACTERISTICS=(DENSITY:6250) LOGICAL_TAPE
```

(Queue an allocation request for a tape drive with 6250 bpi capability and wait until the allocation has completed.)

```
$ ALLOCATE/QUEUED/NOWAIT/NOTIFY DISK$CLASS:-
  $_ /CHARACTERISTICS=(RA60) MY_DISK_PACK
```

(Queue an allocation request for an RA60 disk drive and return control to my terminal. Notify me when the allocation has completed.)

```
$ ALLOCATE/NOQUEUED TERMINAL$CLASS:-
  $_ /CHAR=(AUTODIAL,BAUD:1200 DIAL_OUT_MODEM)
```

(Allocate a terminal device with a 1200 baud autodial modem but don't queue the request. Give an error if all such devices are allocated.)

3.2 Implement AVR in VMS

VMS should provide a complete implementation of automatic volume recognition for tapes, that may be enabled/disabled by the operator on a per drive basis. This means that (with AVR enabled), when a tape is mounted, the system checks possible labels and honors mount requests without operator intervention, if possible. If a job needs 4 tapes, the operator can mount them all if enough drives are available and then forget about them until somebody else needs the tape drives. It should also be possible for a user to request a tape mount based solely on the tape's label and density. The user should not be required to know what physical devices implement a particular tape density on a particular system. VMS should also support a "visual id" or "slot number" which is displayed in all operator messages related to the mount.

It should be possible to operate a VMS system in a mode where all tapes are under system/operator control. This means that they are pre-initialized and users are not allowed to change the labeling on the tape without special privilege. The BACKUP utility must also conform to such labeling restrictions, thereby insuring that the BACKUP data is written onto the proper reels. VMS should require explicit operator intervention for unlabeled tapes. It is not acceptable that an unlabeled tape which happens to be on a drive be automatically assigned.

3.3 Provide a BACKUP/OPERATOR capability

Currently, when the BACKUP command is run interactively, request for additional tape volumes are directed to the interactive terminal. In a large systems environment, users are typically not encouraged to enter the machine room. All tape requests are handled by system operators. It should be possible to specify that all BACKUP tape requests be directed to the system operator.

3.4 Enhance VMS MAIL

There are many enhancements to VMS MAIL that would increase its functionality and ease of use.

Digital should consider incorporating the features of the TOPS-10/20 MS mail utility into VMS MAIL, particularly the user interface commands:

- Prompt for a list of users to be cc'd on the message. It is often useful to distinguish these users from the primary recipients of the message.
- Allow the message to be manipulated before it is sent. In particular, allow the text, "to" and "cc" lists and subject to be edited or changed.
- Allow the user to perform a directory of messages based on search criteria, including
- Before a specified date
- since a specified date
- from a certain user
- subject (a subset of the subject string)
- forward and reverse chronological order
- Make the DELETE command accept a range or list of messages.
- Allow one or more VMS files to be inserted into the text of the message at the place the user is typing.

When a distribution file is specified, the expanded names should optionally be listed so that all recipients can see to whom the message was sent.

3.5 End-to-end encryption of logical links in DECnet-VAX

The VAX/VMS system should support end-to-end DES encryption within DECnet-VAX with a separate DES key being used for each DECnet logical connection. This should be implemented at the NSP level, so that it is transparent to the user. The system manager should be able to activate or deactivate DES encryption between his VAX and any other VAX (that supports that feature). Privileged users on intermediate nodes can read and/or modify data being routed through their nodes or observe data in transit across an Ethernet, and not all nodes in a large DECnet network are equally trustworthy. End-to-end DES encryption would serve to protect this data in transit. Also, DES might be used to protect need-to-know access to classified data in a network.

3.6 Make VMS prompt for unsupplied DECnet passwords

There should be an automatic mechanism by which password prompting would occur when a DECnet access control string is specified with only a username. At the minimum, the COPY command should provide this prompting capability. This is an important security feature for heterogeneous DECnet's where some nodes do not support proxy logins.

3.7 Digital should support TCP/IP on VMS

The TCP/IP protocol is one of the most widely supported network protocols for interconnection of workstations and mid-sized computers. It is frequently necessary to incorporate VAXs into such a mixed-vendor network. The TCP/IP protocol should be supported by Digital under VMS to allow such connections.

3.8 VMS utilities should use a standard output format

Printable output generated by VAX utilities and compilers comes in a great variety of record formats and carriage control conventions. A particularly awkward convention is the use of embedded ASCII control characters to generate multiple print lines from a single record. There appears to be no standard for this or any other mechanism. As a result it is very difficult to print "printable" output on non-Digital printers or transmit it through heterogeneous networks. Digital should document a standard record format and carriage control convention and modify all facilities to conform to this convention. As an alternative, Digital should provide a utility which converts all currently used formats into a standard format. It seems that this functionality currently exists, distributed between the print symbiont, device driver, and "Digital standard" printers.

3.9 Provide in-line help in VMS

Help is available in VMS, but it would be very useful to have this information available as the command is being typed, where it could be used to get a list options for the next portion of the command. Frequently users cannot remember the name of the particular qualifier they want, so they must abort the command, and run the HELP program to find the qualifier. VMS would be much easier to use if the user merely had to type a special character (such as a question mark) and a RETURN to have DCL display a list of what it was expecting to see and retype the command line up the point where the user requested help. (This is similar to the in-line help provided by the TOPS-20 operating system.)

3.10 Provide filename completion in VMS

With the advent of long filenames, it has become a chore to type in the entire file specification. VMS should provide a facility similar to TOPS-20 filename recognition in which the operating system will fill out the rest of the filename when the user enters the ESCape key.

For example, if a directory contained files MARCHDATA.INPUT and MYMESSAGES.LIS, and the user typed "MA<ESC>", VMS should respond by filling out "RCHDATA.INPUT".

3.11 VMS should provide an UNDELETE facility

Users frequently delete files by mistake. VMS should provide an UNDELETE command which allows them to be retrieved. VMS should also provide an EXPUNGE command, which causes deleted files to be removed and their space reclaimed.

When the user DELETEs a file, the file should be marked for deletion, but not actually removed from the file system. The DIRECTORY command should not display the file, except when a /DELETED qualifier is used. When the user (or system operator) issues an EXPUNGE command, all files matching the file specification that are marked for deletion should be removed.

The /VERSION_LIMIT qualifier would indicate how many versions of a file should be kept (in both a deleted and undeleted state) before automatic EXPUNGing occurs.

A new qualifier (perhaps /KEEP_LIMIT) would indicate how many versions should remain undeleted. /KEEP_LIMIT=1 would mean that as new versions of the files are generated, all previous versions would be marked as deleted. Only as many versions as specified by /VERSION_LIMIT would be allowed before the oldest version was expunged.

The system should perform an automatic expunge of the connected directory and login directory complex when the LOGOUT command is issued.

3.12 Support file date last read

VMS provides the ability to maintain a pseudo "date of last access" for files by using a volume wide file retention period to update an expiration date. It would be desirable to have the ability to maintain the date a file was last read, as well as maintain an explicit expiration date for a file. Knowing for certain the date and time a file was last read can be an important security tool. The date the file was last read should be separate from the date the file was last created and the date the file was last modified.

3.13 Allow a privileged user to 'advise' another terminal.

VMS should support a facility which would allow a privileged user to link his terminal to another terminal. This link would at minimum allow the privileged user to issue commands as if they were typed from the other keyboard. This capability would be useful to "cleanup" whatever was running on the remote terminal before its process was deleted. Ideally, the facility should also allow the privileged user to see all output directed to the target terminal. This would allow for fully interactive "hand-holding" or consulting for user problems.

3.14 Provide COMPILE-class commands in VMS

VMS should implement the TOPS-10/20 COMPILE, LOAD and EXECUTE commands.

The COMPILE command invokes the appropriate compiler based on filetype (e.g., .FOR for FORTRAN, .MAR for MACRO). The specific compiler can also be explicitly specified by a qualifier (/FORTRAN, /MACRO). The COMPILE command examines the date on the .OBJ file with the same name as the source file to determine if a recompilation is necessary.

The LOAD command adds a LINK invocation to the COMPILE. The EXECUTE does the COMPILE and LOAD, and then runs the resulting executable image.

These three commands are a great aid to users doing program development, and help to eliminate unnecessary recompiles of modules in large programs.

3.15 Maximum length of account string should be increased

The VMS account string field associated with a process should be expanded to 39 characters to provide increased flexibility.

3.16 Project accounting is urgently needed in VMS

The Spring 1985 VAX SIR Ballot contained a request for project accounting in VMS. Digital's response was "We also feel that project accounting is very important...We feel that this is a reasonably complex area and, as such, some of the enhancements that we intend to make in this area will appear over time."

Project accounting is something that is desperately needed at large sites. In its simplest form, project accounting should provide a SET PROJECT command that would write a process accounting record, and start recording a new record with a new account string specified by the user. The account string should be verified before these actions take place. The system manager should be allowed to set up a file which specifies which UIC's are permitted to use individual account strings.

Digital should provide this form of project accounting until their full-blown system is available.

3.17 VMS should provide mechanism for disk space accounting

VMS should record the user's account string in the file header when the file is created. The string could then be used at user sites to do some form of disk space accounting.

3.18 Increase reliability of accounting records

When the system crashes, no accounting records for logged in processes are written. VMS should provide a mechanism for periodic checkpointing of the records it is collecting so that when the system comes up after a crash, partial records may be written to the accounting file that reflect the information up to the last checkpoint before the crash.

running. Presently only users with WORLD privilege can get this information. The SHOW USERS and SHOW SYSTEM commands should have a qualifier which would make them return this information. Output might look like this:

```
        VAX/VMS V4.3   20-MAY-1986 23:21:54.00   Uptime    3 08:27:11

      Pid    Process Name    Image       State  Pri       CPU
    00000080 NULL            (DCL)        COM     0 3 06:35:37.11
    00000081 SWAPPER         (DCL)        HIB    16 0 00:00:23.34
    00000084 ERRFMT          ERRFMT       HIB     8 0 00:00:08.99
    00000085 OPCOM           OPCOM        LEF     8 0 00:00:02.24
    0000008E OPNS_OPAO       (DCL)        LEF     7 0 00:00:05.51
    000001CB SMITH_VTA58     (DCL)        CUR     4 0 00:00:22.31
    000001CC TPU_SMITH       TPU          LEF     7 0 00:00:13.03
    000001CD JONES_VTA67     KERMIT       LEF     4 0 00:00:11.52
```

Availability of this option might be controlled by a system logical name for the benefit of sites which consider the image name to be sensitive information.

MUMPS

MMP

MUMPS SIG STEERING COMMITTEE

Chairman
Mark Berryman
Plessey Peripheral Systems
Irvine, CA

Symposium Coordinator
Chris Richardson
Computer Sciences Corp.
Ridgecrest, CA

Communications Rep.
Mark Hyde
Advanced Computing Services
DeWitt, NY

Newsletter Editor
Janet Berryman
Plessey Systems, Inc.
Irvine, Ca

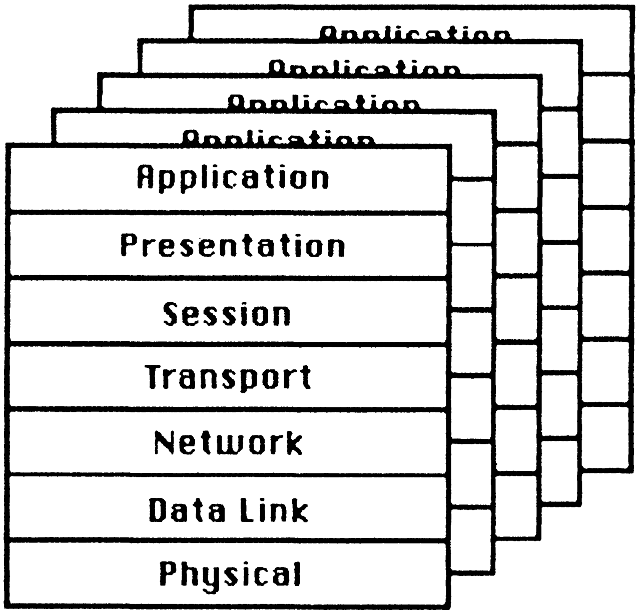
VAX Liaison
Coyett A.J. Dese
VA DM&S Verification & Dev. Ctr.
San Francisco, CA

Digital Counterparts
Beatrice Walther
Digital Equipment Corp.
Marlboro, MA

Diane Brown
Digital Equipment Corporation
Marlboro, MA

NETwords

The NETWORKS SIG Newsletter



NTW

Networks SIG Steering Committee

Bill Brindley
Chairman
Naval Security Group Command
(202) 282-0527

Sandy Traylor
Symposia Coordinator
Target Systems, Inc.
(714) 921-0112

Jim Ebright
Communications Coordinator
Software Results Corporation
(614) 421-2094

Bill Hancock
Technology/
Standards Coordinator
(214) 495-7353

Vickie Hancock
Newsletter Editor
(214) 495-7353

Carole Greenfield
DEC Counterpart
Digital Equipment Corporation

The Networks Special Interest Group (SIG) is one of 25 SIG's within in Digital Equipment Computer User's Society (DECUS). The main purpose of the Networks SIG is to promulgate information concerning the use, development, and standardization of network products that function or involve Digital Equipment Corporation systems. Additional functions of the SIG include the coordination and scheduling of symposia sessions, providing methods for free-flow communications, publication of the Networks SIG newsletter NETWords, participation in domestic and international standards committees, input to Digital for new products and corrections to existing products, promotion of working groups for special network needs and topics, and many, many other functions.

The Networks SIG Steering Committee invites you to participate in the Networks SIG. There are many ways that you can help the Networks SIG. Some of those include chairing sessions at symposium, participation in the various Networks SIG working groups, participation in special research projects, and others. If you are interested in devoting your time and expertise, contact any of the steering committee members.

DECUS is run entirely by volunteer leadership. Help us make DECUS and the Networks SIG better - take an active part in **your** SIG!

OFFICE

AUTOMATION

OA

OFFICE AUTOMATION SIG STEERING COMMITTEE

Chairman
Katherine 'Kit' Trimm
Pivotal, Inc.
Tucson, AZ
602-886-5563

Vice Chairman
Ralph Bradshaw
Johnson and Johnson
Raritan, NJ
201-685-3434

Communications Committee Representative
E. Catherine Ditamore
ARA Services
Philadelphia, PA
215-238-3638

Symposium Coordinator
Mitch Brown
Gen Rad, Inc.
Waltham, MA
617-890-4900

Special Projects
Gene LeClair
HQ Dept. of Army
Washington, DC
202-697-3234

BOF Coordinator
Ray Kaplan
University of Arizona
Tucson, AZ
602-886-5563

Newsletter Editor
Therese LeBlanc
T.M. LeBlanc & Assoc.
Wheeling, IL
312-459-1784

Library
Bob Hassinger
Liberty Mutual Research Center
Hopkington, MA
617-435-9061

Tape Copy Coordinator
Randall Buck
Columbia Savings
Irvine, CA
714-863-3030

ALL-IN-1 Working Group
Leon E. Ottley
Evans and Sutherland
Salt Lake City, UT

Symposia Assistant
Sal Gianni
Northeast Utilities
Hartford, CT
213-665-5652

Store Coordinator
Mike Jackson
Air Force Operational
Test and Evaluation Center
Kirtland AFB, NM
505-256-0267

Personal Computer SIG Liaison
Cheryl Johnson
Grinnell College
Grinnell, IA
515-236-2570

Networks SIG Liaison
Gene LeClair
HQ Dept. of Army
Washington, DC
202-697-3234

DECUS Europe OA SIG
Andreas Verbay
Telinco AG
Spiegelstrasse 20

Digital Counterparts
Les Apigian
Digital Equipment Corporation
Merrimack, NH

Geoff Bock
Digital Equipment Corporation
Merrimack, NH

Session Notes
Martha Rudkin
GMF Robotics
Troy, MI
313-641-4332

Table of Contents

From The Editor

Hello everyone, it's Fall again...here in Chicago that means cooler days and the turning of leaves, my favorite time of the year. This Fall is especially exciting for me because I will be going to San Francisco next month for the Symposium. I hope that many of you will also be able to attend, and to help you on your way, we've included a "Symposia Justification Package". We will also be sponsoring a "First Timer's" evening for everyone to get together and meet each other, plus help introduce our first timers to Symposium.

Our OA Tape Swap is off and running, we have an article about the Tape Swap and Contest, plus a listing of tapes received to date. There is also an excellent article on Password Expiration from a user, and an article about All-In-1 V.2.1 from Digital.

I hope all of you will find the information about Symposia helpful, and I'll see you in San Francisco!

Regards,

P.S. Keep the GREAT articles coming!

Therese LeBlanc
275 London Place
Wheeling, IL 60090

First Timer's Evening

Rallying around the slogan "WE MADE IT FLY!", the Office Automation SIG will be actively seeking new members and increased participation in SIG and Symposia activities in San Francisco. The OA SIG will host a Newcomers (and not-so-newcomers!) evening in their Joint SIG Suite in the Hilton Hotel. It will be a special night for everyone to find out more about the Office Automation Special Interest Group. It will also be a chance for symposia attendees to meet the OA Steering Committee and DEC Developers in a relaxed atmosphere.

It will also be an opportunity for first-time OA SIG participants to find out how to make the most of a very informative, but sometimes hectic week of sessions. Watch for details for this even and others in UPDATE. DAILY and find out how the OA SIG serves the DEC user community!

Symposium Justification Package

During the Dallas Symposium the topic of "justifying" attendance at a symposium to your manager was brought up. Everyone said that they would like to see something in the newsletter prior to Symposia that would help them with the justification process. Well here it is...

There are many different ways to justify attending a symposium, cost effectiveness, training, professional contacts, etc. We have included several of these, plus, a preview of the Office Automation sessions which will be presented.

I. BENEFITS:

1. Sessions range from beginner to very technical so that the individual can target each session to his or her skill level.
2. There will be over 1,000 sessions offered on a wide variety of products and topics, you choose only those which you are interested in attending.
3. You will meet other users with similar interests and systems, and have the opportunity to make professional contacts.
4. The DEC representatives who develop and support the products you use will be accessible to you for discussion of the products, one-on-one problem solving and input for future enhancements.
5. You will get previews of new versions of software for current products, and see new or different products demonstrated.
6. There will be over 50 hours of sessions related to Office Automation including technical sessions for All-In-1 users.
7. During sessions presented by Users, you will be able to see how other companies have used and developed their systems.

The list could go on and on. The key to Symposia is that you can gain information and knowledge on a wide variety of products and systems in a short period of time, and in a cost effective manner. Since the bottom line on most training is cost, the next section shows how you can have all of these benefits for a very reasonalble amount of money.

II. COST:

The fee for attending a full week (5 days) of Symposia is inexpensive compared to most other forms of training. For an average registration fee of $400.00 (5 days) a person could attend in excess of 40 sessions. Here is a sample breakdown for one person to attend Symposia for one full week (based on past trips and fees):

Registration Fee................	$400.00*	
Hotel...........................	$375.00	(@$75.00/day)
Meals...........................	$125.00	(@$25.00/day)
Transportation..................	$**	
Misc............................	$100.00	
TOTAL	$1000.00	

* registration fee includes lunch each day
** air fares will vary according to local cost/free bus services is available from all hotels for DECUS attendees and is used by most people

Based on this breakdown, one week of intensive training would cost about $1,000.00 (not including air fare)...that's about $25.00 per hour for a 40 hour week.

III. SPEAKING:

Many people attend their first (or future) Symposiums by volunteering to present a session. The session may be as short as one-half hour or as long as two hours. Presenting a session is good exposure for both you and the company you work for. Many managers feel that this is a good way to justify letting an employee attend the rest of the Symposia.

These are a few ideas about how you might approach justifying your attendance at the upcoming Symposium. If you need additional information or documentation, please contact a SIG member. Last, but not least, here are some of the benefits past attendees feel that they have gotten from Symposium:

"The DECUS Symposium is a great forum for information exchange. Not only are the sessions extreemely beneficial, but the professional contacts made will invaluable in the future for both finding solutions and solving problems. I'm already looking forward to San Francisco!"
Walt Zelasko, Ameritech

"...you'll meet people you would never get a chance to meet anywhere else, the contacts are invaluable."
Roseanne Flounders, ARA

The Office Automation SIG is proud to present a schedule of excellent sessions for you! Due to the continued growth in our SIG and the number of truly outstanding sessions submitted again, we have scheduled five full days. Plan on staying through Friday afternoon to attend all the sessions of importance to you and then join us at the OASIG Wrapup to let us know how you feel about the SIG's activities. We look forward to welcoming all interested DECUS members to the Symposium and the OA sessions!

The schedule for the week is as follows:

- Our first session Monday morning will open the Symposium with a short roadmap of planned activities for the week, followed by an OASIG Keynote presenting some ideas on the future of information management. The morning continues with a session that has remained one of the most interesting in our schedule -- Digitals's views on the strategic directions of OA products. We then have a user presentation on the challenge of cost-justifying OA systems.

  Monday afternoon's sessions provide updates on the current state of pertinent Digital products. Monday evening's sessions offer the first of several technical sessions about ALL-IN-1.

- Tuesday's sessions present insights into some of the new products and emerging technologies that promise to add variety and flexibility to our OA systems. These topics include videotex, WPS-PLUS External Application Links (XALs), computer conferencing, and TEAMDATA/RALLY. Also included is a session on a calendaring system built by one user organization, and an ALL-IN-1 document archiver built by another.

  The sessions late in the day offer you a choice of topics -- training and support, or advanced technical insights into ALL-IN-1. In the training and support stream, Digital's customer support services are discussed first, followed by sessions on providing training and support to our end-user communities. In the technical stream, we repeat two of the more popular sessions from the last Symposium -- ALL-IN-1 Application Development Tidbits and the Advanced ALL-IN-1 Hack-A-Rama. These sessions begin at 4:00 PM in Section J, and will provide LIVE demonstrations of ALL-IN-1 customizing.

* Wednesday's stream provides some managerial insights into the current state of the OA art. Departmental computing in office networks is the topic of the first session of the day, followed by several managerially-focused sessions. Several of these are presented by active OA practitioners, sharing their experiences. Wednesday afternoon's sessions center on the topic of performance in office systems, with presentations by users and Digital on performance monitoring and tuning.

  The day closes with a brief SIG Business Meeting. Come share your ideas, and meet your fellow OASIG members!

* Thursday offers technical depth on OA products. Topics discussed in this stream include: ALL-IN-1 in a clustered environment, administration of OA systems, desktop publishing, and several technical sessions about ALL-IN-1 and WPS-PLUS. In addition, there are two sessions (0010 and 0011) held from 4:00 PM to 6:00 PM in room 236/238 regarding WPS-PLUS in the PDP-11 computing environment.

  The OA Wishlist and Question and Answer sessions begin at 6:00 PM. At these sessions, an impressive array of Digital's managers and engineers are present to respond to questions about their products and to get YOUR ideas on how they can make their products better. Come and watch the feedback process to Digital in full operation!

* Friday offers a variety of sessions focusing on management issues in OA, including capacity planning and videotex. The OA activities at the Symposium close at the Wrapup sessions.

The OA SIG Steering Committee is looking forward to your participation in our activities. We welcome your ideas and suggestions. Please join us for our best Symposium yet!

Mitch Brown
Symposia Co-ordinator

Sal Gianni
Symposia Assistant

The OA Tape Swap Lives!

Summertime, summertime, and the living is easy..... so they say. Well now that the lucky few (or many) that attended the Dallas party have settled back into the daily routine, let me request a bit of your time.

Contrary to rumors from semi-reliable sources, the OASIG tape lives! In fact, despite the fumbling efforts of yours truly, the #@&%@ thing is really shaping up. As you should know by now, all submissions received before September 1st will be judged in the "We Made it Fly" programming contest. This alone should be reason enough to send in something!!

Further sleuthing has revealed that the judging of contributions may be performed by a select group of individuals working in a foreign country on Digital's Office Automation products. Prizes of SOME VALUE (or should that be 'some value') will be awarded at the OA Magic Session in San Francisco. ** Note -- Winners need not be present **

Current submissions include such things as:

PC file transfer
Phone messages (as part of electronic mail)
Document shredding (no more Delete then Empty Wastebasket)
Secure terminal screen
Print to port fixes
Printer setups for the LN03
Hints and kinks
Work request system
Bulletin Board (Tacks not included)
and many others

What can you contribute? Almost anything (copyright software and jolly roger stuff is frowned upon). Command procedures, programs, (with source please), Datatrieve procedures, UDP's for both ALL-IN-1 and WPS, scripts, extended help text, CBI's, DECALC templates, articles and programming hints.

Stand up, be counted, send in all that great stuff thats just laying around (my system manager really wants me to fill up the spare RA60....). Any and all contributions will be greatly appreciated.

Details, Details...

Send submission on Mag tape, 1600 BPI, VMS backup format. A listing of files is helpful. A completed DECUS tape submission form (I will send you one if needed) should also be included. Media will be returned to sender with all the OASIG tape contributions between September 8th and 15th. Inclusion of a return shipping bag will ensure prompt attention. Affixing an address label and return postage will qualify you for sainthood (and save me a lot of money!).

In general the tape should include an AAAREADME.TXT file in each directory. This text file should contain a brief description of your submission, what it does, and how to use it. Use a directory tree to separate large or multiple submissions. If your submission includes FMS forms, put them in a USER.FLB file.

If you have a Vax Sig tape from the DECUS Program Library, use it as an example of how to package your contributions. Remember to fill out the DECUS Program Library Submittal Form as best you can, and include it with your tape. The 1986/1987 U.S. Chapter DECUS Program Library Software Abstracts book has a form in the back (pages 257 - 260). The form may look imposing at first but you will find that most of the items do not apply.

I want your stuff!!!

Send it too:

Randall Buck
Columbia Savings and Loan
17911 Von Karman Avenue
Irvine, CA. 92714

If I can be of any help call me at (714) 863-3030 ext 564 between the hours of 9:00 am and 5:00 pm Pacific.

PASSWORD EXPIRATION AND ALL-IN-1

De Ann Pevehouse

Ernst & Whinney
555 California Street, Suite 3000
San Francisco, CA 94104

If you are the System Manager as well as the All-in-1 Administrator or even if the two jobs are separate, password expiration for captive All-in-1 users can be a trial. The expiration message always says "see your system manager"!

When we installed VMS 4.0 at our site, we set the password expiration for 60 days. When that time rolled around, no fewer than 50 users called asking to have their passwords reset because they had either ignored the warning messages that their password was about to expire, hadn't seen them in the first place because All-in-1 clears the screen after log on, or they had been out of the office and when they did finally log on, it was past the final expiration date. Therefore, they had one last chance and blew it by logging off without changing it.

Since then, we have increased the expiration date to 180 days but still, there are numerous interruptions regarding expired passwords. As the expiration date is beginning to vary on a user by user basis, the calls are occuring constantly.

As an aid to both the users (!) and the support staff, a check password expiration program has been integrated into our All-in-1 system which directs the user at logon to change his/her password and basically won't let them proceed until they do.

First, some details about our site. Our SYSUAF is setup where the LGI points to SYS$MANAGER:A2LOGIN.COM. All "non-data processing" users are also captive.

We have also implemented a Change Password (CP) option on the Information Management menu which executes a script allowing the user to either choose his/her own password or obtain system-generated passwords. The idea is that this is a voluntary option the user may exercise at any time in order to avoid this expiration problem.

Figure A indicates the contents of the OA$LIB:SETPASS.SCP script which is called by the CP option.

The basic program (PASSWORD.BAS) is written to check for the user's password expiration date and is executed in our A2LOGIN.COM captive log on procedure. Our SYSUAF.DAT is protected against WORLD and GROUP so the program has to be installed with SYSPRV in order to work for our average non-privileged user. The program also allows a foreign symbol to be passed. In this way, the program can be run outside of All-in-1 and a username to check on passed to it. If a username is not passed, the program gets the username of the current process using a LIB$GETJPI call.

A foreign symbol might be setup something like this:

```
$  PW :== $OA$LIB:PASSWORD
```

Then a username can be passed to the program (ie: $ PW SMITH).

The program checks the user's last login date against the last password change date and compares it to the password lifetime. If the result indicates that the password is 2 days or less from expiration, a symbol EXPIRED is set to "YES". As mentioned, this is invoked from A2LOGIN.COM which then proceeds to run All-in-1.

The named data in the EWMAIN form (we renamed MAIN to EWMAIN to keep track of our changes versus the original release) checks to see if EXPIRED equals "YES" as part of its preprocessing of the main menu. If the symbol is "YES", a script is invoked to direct the user to the CP option so they will change their password before it expires and save everybody a lot of trouble. The script uses the .JUDGE directive and the user can't really escape from at least entering the SETPASS.SCP script and running SET PASSWORD.

If they still do not successfully change their password after all that because of choosing an invalid length, using CNTRL/C, etc, the script exits without further force (we don't want to get too tough on them!). They will be caught again the next time they login as the SYSUAF will not update the password change date until they actually have successfully changed it. That is why the program looks for two days or less - to give it a chance to help the user to change their password prior to the expiration instead of waiting for it to totally expire.

Figure B is the basic program. Because the program has to be installed with privileges, it needs to be compiled and linked as follows:

```
$  BASIC PASSWORD
$  LINK /NOTRACE PASSWORD
```

And then installed using:

```
$  INSTALL/COMMAND
INSTALL> REPLACE OA$LIB:PASSWORD/SHARE/PRIV=SYSPRV
```

This should also be added to SYS$MANAGER:SYSTARTUP.COM somewhere too.

Figure C (FORCE_PWCHG.SCP) is the script that is invoked if the named data on EWMAIN indicates that the symbol EXPIRED is set to "YES".

Following is the syntax in EWMAIN's named data which causes this to occur:

```
/PRE='IF CLI$EXPIRED EQS "YES" THEN DO FORCE_PWCHG'
```

We haven't been through another expiration yet but we're counting on this to help!

FIGURE A. - Script to Change Password

```
!
!  SETPASS.SCP - script to change password using system generated or choosing
!                your own - invoked by the CP option on the IM menu

     .CLEAR
     .block


                    P A S S W O R D   C H A N G E   O P T I O N S


     A new feature of VMS includes choosing a system generated
     password instead of having to pick your own.  The list of words
     the system will display are also the required minimum length of
     6 characters or longer.

     --->  Type O to choose your own new password.
     --->  Type G to display a list of system generated passwords.
     --->  Press the EXIT SCREEN key to exit this procedure.

     .end_block

.label get_type

     .text 18,1,"Please enter O or G or press the EXIT SCREEN key:  "
     .single 18,52

     .judge "{KEY 0}"
     .goto abort
     .end_judge

     .judge "O"
     .goto choose_own
     .end_judge

     .judge "G"
     .goto system_generate
     .end_judge

     .judge "{*}"
     .video_att bold
     .clear 24,1
     .text 24,1,"Please type O, G, or press EXIT SCREEN. You pressed "
     .function get oa$script_psib=oa$script_psib "."
     .text 24,53, oa$script_psib
     .video_att off
     .goto get_type
     .end_judge
```

```
.LABEL system_generate

    .clear
    .block

     Instructions for Choosing a New System Generated Password:

  At the Old Password: prompt, type in your current password and press RETURN.

  Wait for the system to display a list of 5 randomly generated passwords.  The
  column on the left are the actual passwords.  The column on the right is a
  syllable breakdown indicating a possible way to pronounce the word.

  At the New Password: prompt, either type in one of the passwords in the
  list or press RETURN to see another list of passwords.  When you type in one
  of the generated passwords, the Verification prompt will display.

  At the Verification: prompt, type the new password again and press RETURN.

  Press CTRL/C (hold down CTRL key and type C) to abort this procedure...


    .end_block

    .cursor 20,1
    get oa$dcl="ASSIGN/USER/nolog SYS$COMMAND: SYS$INPUT:"
    get oa$dcl=" set password/gen"
    .wait Press RETURN to continue...
    .goto exit

.label choose_own

    .clear
    .block

Instructions for Choosing Your Own new Password:

   At the Old Password: prompt, type in your current password and press RETURN.
   At the New Password: prompt, type in a new password and press RETURN.
   At the Verification: prompt, type the new password again and press RETURN.
   Press CTRL/C (hold down CTRL key and type C) to abort this procedure...

   NOTE:  The minimum length for your new password is 6 characters and you
          must choose a NEW password.  The same one will not be accepted again.

    .end_block

    .cursor 13,1
    get oa$dcl=" ASSIGN/USER/nolog SYS$COMMAND: SYS$INPUT:"
    get oa$dcl=" SET PASSWORD"
    .wait Press RETURN to continue....
    .goto exit
```

```
.label abort

    .if cli$expired eqs "YES" then .goto dont_allow_abort
    .clear 22,1
    .video_att bold
    .text 22,1,"Password Change procedure aborted .... "
    .video_att off
    .wait Press RETURN to continue...
    .goto exit

.label dont_allow_abort

    .video_att blink
    .text 22,1,"Please change your password.... it is expiring in 2 days or less!"
    .video_att off
    .goto get_type

.label exit
    clear form
    .exit
```

FIGURE B. - Basic Program

```
1  %title "Password -- check password expiration date"
   %ident "x00.00"

 !**********************************************************************
 ! NOTE FOR LINKING THIS PROGRAM:
 !
 ! Remember to link no traceback because needs to be installed
 ! with SYSPRV
 !
 ! $ BASIC PASSWORD
 ! $ LINK /NOTRACE PASSWORD
 !
 !**********************************************************************

   option                                                           &
       type = explicit,                                             &
       active =    (      integer overflow,                         &
                          decimal overflow,                         &
                          setup,                                    &
                          decimal rounding,                         &
                          subscript checking     )

   ! ************************************************************* !

   map (sysuaf_file)                                                &
           string fill = 4,                                         &
           string username_key = 32,                                &
           string fill = 16,                                        &
           string account = 32,                                     &
           string owner = 32,                                       &
           string fill = 256,                                       &
           long pass_lifetime(1),                                   &
           long last_pass_change(1),                                &
           string fill = 8,                                         &
           long last_login(1),                                      &
           string fill = 1008

   ! ************************************************************* !

   declare string username

   ! ************************************************************* !

   declare long                                                     &
           return_status,                                           &
           lun,                                                     &
           two_days_binary(1),                                      &
           two_days,                                                &
           clunk_diff(1),                                           &
           pass_life_days,                                          &
           no_of_days
```

```
! ************************************************************* !

external long constant                                          &
        ss$_normal

external long constant                                          &
        jpi$_username

external long constant                                          &
        bas$k_endfildev,                                        &
        bas$k_recnotfou

! ************************************************************* !

external long function                                          &
        lib$get_lun                                             &
                (long)

external long function                                          &
        lib$free_lun                                            &
                (long)
external sub                                                    &
        lib$stop                                                &
                (long)

external long function                                          &
        lib$day

external long function                                          &
        lib$get_foreign                                         &
                (string)

external long function                                          &
        lib$set_symbol                                          &
                (string,string,)

external long function                                          &
        lib$subx

external long function                                          &
        lib$getjpi                                              &
                (long,long,,,string,)

! ************************************************************* !

external long function                                          &
        sys$bintim

! ************************************************************* !
```

```
  return_status = lib$get_foreign (username)
  call lib$stop (return_status) if (return_status <> ss$_normal)

  if (username = "") then                                           &
    return_status = lib$getjpi (jpi$_username,0,,,username,)
    call lib$stop (return_status) if (return_status <> ss$_normal)
  end if

main_coding:

  gosub determine_if_password_expired
  goto end_of_program

  ! ************************************************************* !
determine_if_password_expired:

  return_status = lib$get_lun (lun)
  call lib$stop (return_status) if (return_status <> ss$_normal)

  open "sys$system:sysuaf.dat" for input as file #lun,              &
          organization indexed variable,                            &
          allow modify,                                             &
          access read,                                              &
          recordtype none,                                          &
          map sysuaf_file,                                          &
          primary key username_key

  on error goto check_no_record
  get #lun, key #0 eq username
  on error goto 0

! get the difference between date of last password change and last log on
! date which is presumably today unless this is being called with a foreign
! symbol for username

  return_status = lib$subx (last_login(),last_pass_change(),clunk_diff(),)
  call lib$stop (return_status) if (return_status <> ss$_normal)

! convert the quadword CLUNK_DIFF to NO_OF_DAYS as days in longword

  return_status = lib$day(no_of_days,clunk_diff() by ref,)
  call lib$stop (return_status) if (return_status <> ss$_normal)

! convert password lifetime from UAF to number of days in longword too

  return_status = lib$day(pass_life_days,Pass_lifetime() by ref,)
  call lib$stop (return_status) if (return_status <> ss$_normal)

! now convert 2 days ASCII delta time to quadword VAX date in clunks

  return_status = sys$bintim("2 00:00:00.00",two_days_binary() by ref)
  call lib$stop (return_status) if (return_status <> ss$_normal)
```

```
 ! then convert to longword days

   return_status = lib$day(two_days,two_days_binary() by ref,)
   call lib$stop (return_status) if (return_status <> ss$_normal)

 ! now compare the difference between the last login date minus the passchange
 ! date with 2 days as absolute values -

   if (abs(pass_life_days) - no_of_days) <= abs(two_days) then
           return_status = lib$set_symbol("expired","YES",)
           call lib$stop (return_status) if (return_status <> ss$_normal)
   else
           return_status = lib$set_symbol("expired","NO",)
           call lib$stop (return_status) if (return_status <> ss$_normal)
   end if
   on error goto 0

   close #lun
   return_status = lib$free_lun (lun)
   call lib$stop (return_status) if (return_status <> ss$_normal)

1111       return

 check_no_record:

   call lib$stop (err) if (err <> bas$k_recnotfou)
   resume 1111

 check_end_of_file:

   call lib$stop (err) if (err <> bas$k_endfildev)
   resume 1111

   ! ************************************************************* !

 end_of_program:

   end
```

FIGURE C. - Script to force password change

```
!  FORCE_PWCHG.SCP - if symbol EXPIRED eqs "YES", then this script is invoked
!                    from the main menu to get the user to change his/her
!                    password prior to the expiration date

   .show_form ewmain
   .video_att bold
   .text 22,1,"Your password will expire in 2 days or less...."
   .text 23,1,"Please type IM CP and press RETURN to change your password:"
   .video_att off

 .label get_input

   .prompt 23,69

   .judge "IM{CR}"
   .SHOW_form im
   .goto get_cp_option
   .end_judge

   .judge "IM CP{CR}"
   do setpass
   .goto exit
   .end_judge

   .judge "{*}"
   .video_att blinking
   .clear 23,1
   .text 23,1,"Please enter IM CP and press RETURN to change your password NOW:"
   .video_att off
   .goto get_input

.label get_cp_option

   .clear 23,1
   .video_att bold
   .text 23,1,"Enter CP and press RETURN to change your password:"
   .video_att off

.label get_cp_prompt

   .prompt 23,67

   .judge "CP{CR}"
   do setpass
   .goto exit
   .end_judge
```

```
   .judge "{*}"
   .video_att blinking
   .clear 23,1
   .text 23,1,"Please enter CP and press RETURN to change your password NOW:"
   .video_att off
   .goto get_cp_prompt

.label exit

 ! set symbol EXPIRED to "NO" even if password change not actually successful
 ! the program (PASSWORD.EXE) will get them again at next login...
 ! if we don't do this, they will be caught in this script everytime they go
 ! to the Main Menu during this interactive session - somewhat cruel

   get cli$expired="NO"
   form ewmain
   .exit
```

ALL-IN-1 Version 2.1 Advances the Standard

In August, Digital announced Version 2.1 of it's industry leading ALL-IN-1 Integrated Office and Information System. This new release advances the standard in Integrated Word and Document Processing, Electronic Mail, Integrated Business Solutions, Range of System Solutions, and Customer Support Services.

ALL-IN-1 Version 2.1 has new features and packaging which include: WPS-PLUS, the best integrated word and document processing; Message Router, for the most open electronic mail system; VAX FMS Forms Management System and the MicroVAX ALL-IN-1 Generator Kit to develop the best integrated solutions for your business needs.

ALL-IN-1 Version 2.1, in fully customizable form, now runs from desktop VAXstation II to the VAX 8800. This feature, combined with Digital's new DSIN on-line customer support system, gives you the flexibility to select the system and services that suit your business needs.

ALL-IN-1 customers can take advantage of these benefits immediately. Customers with service contracts will receive Version 2.1 automatically. ALL-IN-1 customers without contracts, or customers with WPS-PLUS/VMS V2.0 or DECmail, can take advantage of ALL-IN-1 Version 2.1 migration options.

Upgrading from ALL-IN-1 V2.0 is smooth, easy and desirable. All remaining Version 1 customers should plan migration to V2.1 at this time.

Key Highlights of Version 2.1

Integrated Word And Document Processing

WPS-PLUS, Digital's industry leading word and document processing system, is a standard part of ALL-IN-1 V2.1, bringing the power of integrated linguistic aids and advanced editing features to every ALL-IN-1 user.

Besides the ability to easily create, edit and print documents, WPS-PLUS gives the user such time saving features as: automatic numbered footnotes and endnotes, automatic paragraph numbering, automatic table of contents, change bars, redlining, production of form letters, mailing list maintenance, use of Digital's rich technical character set, powerful scientific equation editing, diagram and matrix editing and more.

Linguistic aids, including integrated spelling verification and correction (DECspell), word usage alert, and electronic thesaurus, are standard WPS-PLUS features with ALL-IN-1 V2.1.

To help in the use of use all these features, WPS-PLUS documentation is now included in the ALL-IN-1 User Documentation Set.

WPS-PLUS documents filed in the ALL-IN-1 V2.1 File Cabinet can be retrieved quickly and easily by name, document number or keyword -- even for occasional or limited use, based on content.

The optional DECpage product can be integrated into ALL-IN-1 to let users print typeset quality text and graphics directly from an ALL-IN-1 menu, without intermediate DECdx translations.

Electronic Mail

Digital is the recognized leader in Electronic Mail systems, and our strategic mail backbone, Message Router V2.0, is a standard part of ALL-IN-1 V2.1.

With the VMSmail Gateway services now included, users can exchange mail messages with other ALL-IN-1 users and VMSmail users on the same of different systems.

Message Router also gives every ALL-IN-1 system a base for adding optional gateways to other mail systems or services. The optional Message Router X.400 Gateway makes ALL-IN-1 the first integrated Office and Information system with the ability to exchange electronic mail with any other messaging service that uses the internationally accepted X.400 standard.

Integrated Business Solutions

For the first time, ALL-IN-1 V2.1 is fully customizable on every VAX, from MicroVAX II to VAX 8800.

ALL-IN-1 customization tools, VAX FMS and the FMS Forms Language Translator, are now standard with the ALL-IN-1 V2.1 package. System programmers can quickly and easily tailor ALL-IN-1 menus and forms to specific business needs.

The MicroVAX ALL-IN-1 Generator Kit, also included as part of the Version 2.1 package, lets system programmers create an ALL-IN-1 business solution on a central development system (including a MicroVAX II) which can then be easily deployed to any number of MicroVAX II ALL-IN-1 systems.

(Note: Regardless of the manner in which ALL-IN-1 was implemented, every VAX system running ALL-IN-1 must be properly licensed).

ALL-IN-1 Version 2.1 builds on Digital's commitment to deliver complete systems which help solve a diverse set of business problems in the office. Complete departmental solutions and a host of integrated applications developed by Digital's Cooperative Marketing Partners (CMPs) are available, including:

* The ALL-IN-1 System for Sales and Marketing, for Sales and Marketing departments

* The ALL-IN-1 System for Business Operations, for financial forecasting and management applications

* The ALL-IN-1 System for Telecommunications Management, for Telecommunications departments

* The ALL-IN-1 System for Employment Management, for employment departments

* VAX TEAMDATA, for end user data management

* VAX VTX, for corporate-wide information networks

* Ross Systems Accounting (CMP), for accounting applications

Range Of System Solutions

ALL-IN-1 Version 2.1 is available on all VAXes running a supported version of VMS Version 4, from MicroVAX IIs through multinode VAXclusters -- giving you a range of cost effective system solutions.

Digital wants its customers to achieve the most successful implementation possible. Large VAX systems and VAXclusters are an excellent way to manage growth, but it is important that any system be implemented correctly. Digital Software Services has the expertise to help you develop solutions on any configuration, especially large systems and VAXclusters.

NEW SOFTWARE PRODUCT SERVICES

As part of the ALL-IN-1 support contract, the following services were added this year and now include ALL-IN-1 Version 2.1

* ALL-IN-1 Information Update.

  This is a bi-monthly quarterly publication aimed at system managers and technical support people, to help them better manage ALL-IN-1 systems and assist their user base. It is for ALL-IN-1 what the Software Dispatch is for VMS.

* Digital Software Information Network (DSIN) is now available to ALL-IN-1 customers with BASIC and DECsupport service contracts. By dialing into this database, customers can get immediate answers to product questions, submit Software Performance Reports, and view general product information. This service is available when you need it - 20 hours a day, 7 days a week.

CAPACITY PLANNING CONSIDERATIONS

To obtain a predictable level of performance from ALL-IN-1 Version 2.1 business solutions, all customers should be implementing active capacity plans. These plans will quantify the size of the department to be supported, the percentage of this population which will actively use the system at any time, peak loads, functionality and service time requirements, and the add-ons and upgrades which will keep installations growing.

ALL-IN-1 Version 2.1 systems supporting equivalent functionality and user populations will have the same overall performance as ALL-IN-1 Version 2.0 systems. ALL-IN-1 customers using features like WPS-PLUS word processing and Message Router mail facilities for the first time must be sure they have the memory and disk capacity necessary to support these functions, and have adjusted their maximum active user expectations to account for the enhanced functionality provided.

Upgrade Considerations

Customers who have added their own menus and applications to ALL-IN-1 Version 2.0 are unlikely to need to change them for Version 2.1 - provided they have followed the customization guidelines in the ALL-IN-1 V2.0 Application Reference guide. Modifications to standard menus and forms may need to be re-entered.

File cabinets do not need to be converted when upgrading from ALL-IN-1 V2.0 to V2.1.

ALL-IN-1 Version 1 customers desiring to migrate to Version 2.1 should contact Digital Software Services to discuss Version 2.1 migration services.

Personal Computer Special Interest Group

Newsletter

Personal Computer SIG Steering Committee

SIG CHAIRMAN

Barbara Maaskant
UT Health Science Center
7703 Floyd Curl Drive
San Antonio, TX 78284
(512) 691-7351

PRO WORKING GROUP CHAIRMAN

Thomas R. Hintz
University of Florida
IFAS Computer Network
1022 McCarty Hall
Gainseville, FL 32611
(904) 392-5181

DECMATE WORKING GROUP CHAIRMAN

Cheryl Johnson
Grinnell College
P.O Box 805
Grinnell, IA 50112-0810
(515) 236-2570

RAINBOW WORKING GROUP CHAIRMAN

Lynn Jarrett
Union Tribune Publishing
P.O. Box 191
San Diego, CA 92108
(619) 299-3131 x1130

NEW PRODUCT WORKING GROUP CHAIRMAN

Frederick G. Howard
Eastman Kodak Company
901 Elmgrove Road, D345-LP
Rochester, NY 14650
(716) 724-5331

LIBRARY COMMITTEE REP./ LIBRARIAN

Ron S. Hafner
Hafner and Associates
P.O. Box 2924
Livermore, CA 94550
(415) 449-4178

COMMUNICATIONS COMMITTEE REP./NEWSLETTER EDITOR

Kenneth LeFebvre
Sytek, Inc.
19 Church St.
P.O. Box 128
Berea, OH 44017
(216) 243-1613

PROFESSIONAL CONTRIBUTING EDITOR

Gary Rice
McDonnell Douglas
K34-C636-4W
5701 Katella Avenue
Cypress, CA 90630

PRE-SYMPOSIA SESSION COORDINATOR

Vince Perriello
Crosfield CSI
570 Taxter Road
Elmsford, NY 10523

SYMPOSIA COORDINATOR

Rick Eliopoulis
5258 Vickie Drive
San Diego, CA 92109
(619) 225-7867

SESSION NOTES EDITOR

Alan Bruns
Allied Electronics
401 E. 8th Street
Fort Worth, TX 76102
(817) 336-5401

CAMPGROUND COORDINATOR

Jim Wilson
Ntl Tech Inst for the Deaf
Rochester Inst of Tech
P.O. Box 9887
Rochester, NY 14623
(716) 475-6241

PROFESSIONAL LIBRARIAN

Peter Flack
Computer Sciences Corp
P.O. Box 12233
Research Triangle Park,
NC 27709
(919) 541-4669

MEMBERS-AT-LARGE

Michael Bowers
University of California
Animal Science Departn
Davis, CA 95616
(916) 752-6136

Theodore Needleman
Hardcopy Magazine
Seldin Publishing, Inc.
1061 S. Melrose, Suite D
Placentia, CA 92670

Russ Wertenberg
Sandia National Labs
Division 8352
Livermore, CA 94550
(415) 422-2663

DIGITAL COUNTERPARTS

PRO
Lin Olsen
Digital Equipment
Corporation
30 Porter Road (LJ02/I3)
Littleton, MA 01460

DECmate
Ron Gemma
Digital Equipment
Corporation

Rainbow
Katrina Holman
Digital Equipment
Corporation
30 Porter Road (LJ02/I3)
Littleton, MA 01460

Dear Fellow PC Enthusiasts,

We had not really planned on having an issue this month in the big DECUS Newsletter book. However, because this information is so important for those of us who will be attending the symposium, I decided to print the following submission.

I look forward to seeing you at the symposium. God bless each one of you!

Ken

Kenneth LeFebvre
PC Sig Newsletter Editor

Hi Everyone,

I thought I'd give all of you a little heads up on what the Personal computer schedule is going to look like for San Francisco so that you can do a little advanced planning.

First, I must tell you that this is the first time in the history of the PC SIG that submitted sessions had to be cancelled. It turns out that the Moscone Convention Center is smaller than other convention centers where DECUS has been held in previous years. That meant that all SIGs had to reduce there session numbers in order to fit into the space we've been allocated. If you were in Dallas, you'll remember that we nearly 'swam' in the space available. In fact, we had over 5200 attendees but it didn't seem that way because everything was so spread out. That won't be the case for the symposium fall.

The following is an overall look at the Sessions-At-a-Glance (SAG) for the entire PC SIG. Following the SAG is a complete listing of the abstracts for each session.

I'll be looking forward to seeing you in San Francisco. I'm sure this will be the most exciting and informative set of sessions offered to date by the PC SIG.

Rick Eliopoulos
PC SIG Symposium Coordinator

Monday

1000	P001	1.0	BUSINESS MTG (ROADMAP)
1100	P035	1.5	INTGRATED PC SYSTEMS
1230	P038	0.5	WINDOWING FOR DOS PCs
1300	P039	1.0	PC & TECH. WINDOWING FOR DOS
1530	P017	0.5	CP/M IS ALIVE & WELL IN PD
1600	P048	1.0	LOTUS 1-2-3 VERSION UPDATE
1700	P037	1.0	DOS TECH SESSION
2000	P016	1.0	DBASE II & III
2100	P026	1.0	FIDONET
2200	P025	1.0	PUBLIC DOMAIN (RB)
2200	P054	1.0	CONTEXT INDEXING & RETRIEVAL OF TEXT ON PCs & VAX

Tuesday

1200	P019	1.0	DEC TO IBM PC GRAPHICS
1200	P011	1.0	DEMATE PRODUCT UPDATE
1300	P012	0.5	DECMATE Q&A
1300	P034	1.0	RB UPDATE W/Q&A
1330	P014	1.0	GOLD KEY GROUP
1430	P044	0.5	HOW TO SELECT A PC WORKSTATION

Wednesday

1000	P036	1.0	MS-NET & DECNET
1200	P041	1.0	GUIDE TO DECs PC DOCUMENTATION
1300	P053	0.5	RB WORKING GROUP MTG
1400	P024	0.5	PRO PRODUCTS PANEL
1430	P030	1.0	P/OS V3.0 SYS MGT. & SERVER FUNCTION
1530	P021	1.0	TAILORING P/OS TO SAVE DISK SPACE
1630	P031	1.0	MULTI-TERMINAL ASPECTS OF PRO TOOL KIT
1730	P032	1.0	ADV. FEATURES OF PRO TOOL KIT
1800	P018	1.0	SECURITY RESOURCE SHARING IN PC EVRIRONMENT
1800	P027	1.0	REMOTE OPERATIONS OF A RB USING A RB AS A HOST
1830	P028	0.5	REVIEW OF PRO WISHLIST

Thursday

0900	P015	1.0	THE ATLANTA HOTLINE
1000	P029	1.0	USING THE PALETTE W/PC
1100	P022	0.5	PRO/SIGHT FILL CHAR & TEXT FONT
1130	P033	1.0	PRO Q&A
1300	P004	1.0	JUST A MODEST PC PROPOSAL
1400	P042	0.5	REVIEW OF RB WISHLIST
1430	P013	0.5	DECMATE WORKING GROUP MTG & WISHLIST
1500	P002	1.0	PC SIG WRAPUP
1800	P006	0.5	PRINTING GRAY TONES ... RAINBOW
2000	P005	1.5	PC MAGIC, WAR,& HORROR STORIES

Friday

1200	P045	1.0	PROB. & PITFALLS & REWARDS IN MULTIVENDOR ENV.
1300	P047	1.0	PRO/VENIX-UNIX ON DEC 16-BIT MACHINES

PERSONAL COMPUTER SIG BUSINESS MEETING AND ROADMAP
P001 MONDAY 10:00 1 HOUR

The Personal Computer SIG will address issues relating to Digital personal computing. The rapid growth of these lines requires timely presentations of relevant technical information to end users and developers alike. The SIG leadership will be presenting their goals and objectives and looking for feedback and new volunteers.

INTEGRATED PERSONAL COMPUTING SYSTEMS
P035 MONDAY 11:00 1.5 HOURS

Digital's approach to personal computing is based on networking MS-DOS-based computers to VAX/VMS systems, and using a VAX computer as a server, that is, as a resource sharing machine, for a network of personal computers. Besides providing an overview of Digital's personal computing systems architecture, this session describes the hardware of the new personal computer.

WINDOWING FOR MS-DOS PERSONAL COMPUTERS: USER PERSPECTIVE
P038 MONDAY 12:30 .5 HOURS

Windowing is the user interface of choice for Digital's new personal computing system. This session provides an overview of the windowing environment on Digital's MS-DOS personal computers. The beauty of windowing, its ease of use and benefits are discussed from a user perspective.

WINDOWING FOR MS-DOS PERSONAL COMPUTERS: TECHNICAL
P039 MONDAY 1:00 1 HOUR

This session is a technical discussion of the windowing environment for Digital's MS-DOS personal computing systems. You'll find out how the generic windowing application was modified and tailored to take advantage of Digital's personal computers. The considerations and advantages of writing your applications to run in this windowing environment are also addressed.

CP/M IS ALIVE AND WELL AND PUBLIC DOMAIN SOFTWARE IS, TOO
P017 MONDAY 3:30 .5 HOURS

A good portion of the Rainbow users' communities using CP/M with all the attention being paid to MS/DOS, this area of PC applications is being ignored. Much of the existing Public Domain software offers a variety of support operations which can make the job of the CP/M "system operator" a lot easier.

LOTUS 1-2-3 V2
P048 MONDAY 4:00PM 1 HOUR

The new version of Lotus 1-2-3 integrated spreadsheet program has been on the top of the Rainbow wish list for some time now. This session discussed the differences between V1 and V2 and addressed migrating worksheets from the older version to the new. It also touches upon the design considerations for making 1-2-3 work in the server-based environment of Digital's new integrated personal computing system.

MS-DOS TECHNICAL SESSION
P037 MONDAY 5:00 1 HOUR

This session is a technical discussion of the MS-DOS operating system for Digital's personal computing systems. Version 3.1 is compared to V2.11 and to PC-DOS. The ties between MS-DOS and the windowing environment as well as those between MS-DOS and MS-NET are touched upon briefly.

DBASE II AND DBASE III HELPS AND HINTS
P016 MONDAY 8:00PM 1 HOUR

Despite protests to the coutrary dBASE II (CP/M and MS-DOS) and dBASE III (MS-DOS) is still the most popular micro-database management system. The manuals which come with the software does not provide enough "practical" information. Not only are there several other supporting manuals to be used as reference, but experienced programmers can provide guidence to help "ease the way".

FIDONET--A WORLD-WIDE, GRASSROOTS COMMUNICATIONS NETWORK
P026 MONDAY 9:00PM 1 HOUR

For the past few DECUS Symposia, FIDO has been discussed primarily as a source of public domain software, residing on a large number of personal computers, and many of these being Rainbows. However, there is another aspect to FIDO which is often overlooked, FIDONET. Through the use of this informal dial-up network, FIDONET links almost 1000 independent FIDO nodes into a near worldwide public domain, communications network operating at the grassroots level. With FIDO nodes located in the US, Canada, England, Sweden, Holland, Norway, Finland and Indonesia, FIDONET has been providing a low-level mechanism for the exchange of ideas and information to thousands of personal computer owners.

This session will deal more with the network applications of FIDONET, than with FIDO as a source of public-domain software.

And, since we are right in FIDO's backyard, we may have a few surprises in store at the session.

RAINBOW PUBLIC DOMAIN SOFTWARE
P025 MONDAY 10:00PM 1 HOUR

An introduction to Public Domain Software for the Digital Equipment Corporation Rainbow. Topics to be discussed include: What is Public Domain, how do I obtain Public Domain software, what Public Domain Software is available, and futures of Public Domain.

CONTENT INDEXING AND RETRIEVAL OF TEXT ON VAXES AND PCS IN A LEGAL ENVIRONMENT
P054 MONDAY 10:00 1 HOUR

This session will discuss the author's efforts at designing and developing a methodology and implementation of indexing text material stored in various formats on VAXes and PCS. Information on the user interface and search techniques will be presented. Information and feedback gained from users will be discussed. The tradeoffs in developing software for the PC and VAX and the use of the PC in the development process will be examined.

DEC RAINBOW TO IBM PC: CREATING PORTABLE GRAPHICS
P019 TUESDAY 12:00 1 HOUR

The problem of porting grahics programmes between the Digital Equipment Corporation Rainbow and the IBM PC are discussed. The inherent difficulties lie in the differences between the hardware configurations per graphics in these machines. A solution to the problem lies in the use of the ANSI standard for graphics, namely the Graphics Kernel system. Examples on how portable programs can be written will be given in C.

DECMATE PRODUCT UPDATE
P011 TUESDAY 12:00 1 HOUR

This session will inlude an update on all new DECmate products since last symposia. This will include both hardware and software updates.

DECMATE QUESTION AND ANSWER
P012 TUESDAY 1:00PM 1 HOUR

Questions and answers pertaining to the DECmate covering software and hardware includes WPS-8, OS-278, CP/M, MS-DOS, DECmate I, DECmate II, and DECmate III.

RAINBOW PRODUCT UPDATE WITH QUESTIONS & ANSWERS
P034 TUESDAY 1:00PM 1 HOUR

This session will be an opportunity for DECUS attendees to get answers to questions not covered in other sessions. Panelists will be from Rainbow Engineering, Product Management, and Marketing. The intent is to clarify issues on the current Rainbow systems, hardware and software. This is also an opportunity to give Digital feedback and to voice "wish list" items for future personal computing systems.

USER SUPPORT GROUPS FOR OA PRODUCTS (GOLD KEY GROUPS)
P014 TUESDAY 1:30PM 1 HOUR

Gold Key Groups are professional groups of users of Digital Equipment Corporation's office automation products, supported by Digital for customers with more than 30 office automation users at one location. This session will discuss what a Gold Key Group does and how it can be used effectively.

HOW TO SELECT PERSONAL COMPUTER WORKSTATIONS FOR YOUR ORGANIZATION

P044 TUESDAY 2:30PM .5 HOURS

This session will assist users on how to select personal computer workstations for their organizations along with options for the PC's. this session will study as a case history the PC workstation selection policy of one part of the Naval Weapons Center. Covered in this talk will be the following topics:

- * Determining user requirements
- * Translating user requirements into system requirements
- * Developing the specifications for contracts
- * Procurement of workstations
- * Integration of workstations into the organization (Training)

People attending this session will learn tips that will assist them in making better decisions about workstation selection for their company.

PC NETWORKING: MS-NET AND DECNET

P036 WEDNESDAY 10:00 1 HOUR

For those who want to find out about the technical networking aspects of Digital's integrated personal computing system, this session discusses MS-NET protocols and services for Digital's MS-DOS personal computers. It also explains how this environment differs from DECnet, how MS-NET services can be integrated with a DECnet environment, and how Digital server software supports MS-NET operations in a VAX/VMS environment.

GUIDE TO DIGITAL'S PC DOCUMENTATION

P041 WEDNESDAY 12:00 1 HOUR

This overview of Digital's documentation for MS-DOS personal computers includes new publications as well as those in progress. Besides the traditional hardcopy form of documentation, on-line user information is discussed. Digital will respond to previously-given suggestions from customers and solicit feedback for future consideration.

RAINBOW WORKING GROUP MEETING

P053 WEDNESDAY 1:00PM .5 HOURS

This session will be for present members of the Rainbow Working Group as well as any new persons wishing to join the Rainbow Working Group. In this session, planning will take place for ideas of what the Rainbow Working Group can do for the betterment of the PC SIG.

PRO PRODUCTS PANEL

P024 WEDNESDAY 2:00PM .5 HOURS

This Panel will review the entire product set offered by Digital on the PRO, including the various expansion boxes and hardware options. It will include examples of applications for the various options. In addition to a hardware review, particular attention will be paid the applications product set, and PRO/Server. An overview of the new Synergy application will be presented.

P/OS V3.0 SYSTEM MANAGEMENT & SERVER FUNCTIONALITY

P030 WEDNESDAY 2:30 1.5 HOURS

Now that P/OS can support multiple terminals and serial users, the need to system manage in a multi-user environment arises. This session will introduce the person managing either a single user or multi-user system to setting up accounts, assigning file access privileges, and doing system back-ups using archive and BRU. It will also cover performance issues. In addition this session will present an introduction to the uses and internals of the Server functions of P/OS. PRO Server enables a PRO to act as a File Server and Print server to other PRO-300 series workstations through a network link. Disk drives are then optional on other PROs in the network. (Yes, it can boot the operating system through the network without any local disks).

TAILORING P/OS TO SAVE DISK SPACE

P021 WEDNESDAY 3:30PM 1 HOUR

Information will be provided for modifying P/OS applications that are installed on the hard disk to reduce the amount of space required for storage.

MULTI-TERMINAL ASPECTS OF PRO TOOLKIT

P031 WEDNESDAY 4:30PM 1 HOUR

This session will cover the new multi-terminal functions of P/OS. It will include how to configure the system and how to write applications to use up to 4 video terminals connected into one PRO 300 series system.

ADVANCED FEATURES OF THE PRO/TOOL KIT

P032 WEDNESDAY 5:30PM 1 HOUR

Advanced Information on the uses of the PRO/Tool Kit will cover the following topics: Example Programs, System Service Calls, How to use Core Graphics, Example calls to GIDIS, Differences between P/OS & RSX, Differences between P/OS & VMS, Most Common Errors and Problems, and Tricks and Gimmicks.

SECURITY IN A RESOURCE SHARING ENVIRONMENT
P018 WEDNESDAY 6:00PM 1 HOUR

With the proliferation of PC's, in solely PC or PC-host environments, the safty of data becomes a more and more critical issue. Since accesing files requires minimal skills alteration or even deletion of information,deliberately cryaccident is certainly ample cause for managerial mightmares. Recognition of the problems associated with this situation and resolution of those problems most go beyond reloading the latest 'backup' (if it even exists).

REMOTE OPERATIONS OF THE DEC RAINBOW USING MS-DOS
P027 WEDNESDAY 6:00PM 1 HOUR

This paper explains why and how one would set up a DEC Rainbow PC to operate as a Host computer with dial-in capabilities. With such a system at your office, you can call in from home using a simple terminal and modern to run program like WordStar, dBASE II, BASIC, FORTRAN and ets. If you use another Rainbow as the "smart" termainal, all kinds of files can be down loaded and uploaded with error checking.

REVIEW OF THE PROFESSIONAL WISH LIST
P028 WEDNESDAY 6:30PM .5 HOURS

Requests for additions or enhancements to PRO hardware and software have been submitted and added to a growing list. This session will review the progress of Digital in responding to these suggestions. Additional items can be made by attendees of the session.

THE ATLANTA HOTLINE: HOW TO USE IT EFFECTIVELY
P015 THURSDAY 9:00 1 HOUR

With the number of Digital Equipment Corporation's personal computers in use, there is a growing need for a quick and easy method of solving problems that occur with these systems. The Atlanta Hotline provides an 800 nunmber that can be called by registered users/owners to obtain hardware and/or software assistance. Experience at other DECUS Symposia has shown that there are occasional problems or misunderstandings in the use of the Atlanta Hotline. This session will examine ways to use the Atlanta Hotline effectively.

USING THE PALETTE SYSTEM WITH A PC
P029 THURSDAY 10:00 1 HOUR

This session will discuss the Polaroid PALETTE System, what it is, what it can do for you, and how to use it. This camera attachment is available for use on the RAINBOW, DECmate, Professional and other microcomputers. Suggestions will be provided for improving the appearance of graphic images and text with the use of color.

PRO/SIGHT FILL CHARACTER AND TEXT FONT EDITOR
P022 THURSDAY 1:00 .5 HOURS

The Pro/sight graphics editor on the Pro-300 allows the use of three font files. One file is for fill patterns and the other two files are for text fonts. A new program will be described that allows users to create new, modify existing, or combine several fonts that can be utilized by Pro/sight. Samples of new text fonts and fill patterns will be shown.

PROFESSIONAL SERIES Q & A
P033 THURSDAY 11:30 1 HOUR

This Q&A session is designed to answer general and technical questions on all Professional 300 Series based products. A panel of product management and technical personnel will be available. Time will be provided to discuss wishlist items for the PRO.

JUST A MODEST PC PROPOSAL
P004 THURSDAY 1:00PM 1 HOUR

Most of us at one time or another have had ideas for enhancing PC hardware and software that we would like to direct to the Digital developers. Now is your chance to present new ideas for features and/or products which will help enhance Digital's Personal Computing product lines.

This session will address proposals for all of Digital's Personal Computing products. Entries may be serious or humorous. Audio-visual aids are encouraged but not required. Visit the PC SIG Campground for further information.

REVIEW RAINBOW WISH LIST
P042 THURSDAY 2:00PM .5 HOURS

This session will give the audience a chance to ask where DEC is on resolving the issues of the Rainbow Wish List that is published in The Big One and that is also submitted to them periodically. The audience will also have a chance to make their own submissions to the Rainbow Wish List.

DECMATE WORKING GROUP MEETING AND WISH LIST
P013 THURSDAY 2:30PM .5 HOURS

Reqests for additions or enhancements to DECmate hardware and software have been submitted and added to a growing list. This session will give users a chance to add to the list and review the progress of Digital Equipment Corporation in responding to these suggestions.

PERSONAL COMPUTER SIG WRAP-UP
P002 THURSDAY 3:00PM 1.0

The Personal Computing SIG wrap-up session is intended to allow the SIG members to communicate their needs and desires for future sessions and activities. Any item relating to SIG functions or goals is appropriate for discussion. The current symposium will be evaluated and discussed with the objective being to make sessions at future symposia presented by the PC SIG fulfill the needs of attendees.

PRINTING GRAY TONES FROM THE RAINBOW COLOR GRAPHICS OPTION
HIGH RESOLUTION BITMAP ON AN LA50 PRINTER USING MACRO ASM
P006 THURSDAY 6:00PM .5 HOURS

The Rainbow Color Graphics Option High Resolution Bitmap is read into main memory using code based on the demonstration programs provided in the "Rainbow Color/Graphics Option Programmer's Reference Guide". A Macro Assembler Language Procedure then tests the bit map for what will be represented as white, black, light gray, or dark gray intensity levels. These intensity levels are then translated into bit patterns that are sent to the LA50 printer

PC MAGIC, WAR STORIES, AND HORROR TALES
P005 THURSDAY 8:00PM 1.5 HOURS

Are you a frustrated PC wizard? Have you been waiting for a chance to tell your favorite PC story, but haven't been able to find an audience? Now the PC SIG is giving you a chance to impress (or at least try to) other PC users with your magic.

Prizes will be awarded to top presentations as judged by the panel. Judging criteria include originality, technical content, and entertainment value. Visit the PC SIG Campground for further information.

PROBLEMS, PITFALLS AND REWARDS OF SOFTWARE STANDARDS IN A MULTI-VENDOR PC ENVIRONMENT
P045 FRIDAY 12:00 1 HOUR

While choosing applications software written in a high-level language should allow ease of use in a multi-vendor hardware environment, the path to standardization is far from easy. Nonetheless, when different corporate divisions and independent outside contractors select different hardware, some common denominators can be exploited.

This session explores some of the opportunities and problems presented when management attempts to develop a common set of applications to run on Digital and other PC hardware. The application considered is an well-respected vertical market software package originally written in MicroSoft BASIC for the IBM PC. The problem confronted was how to implement this package at a number of sites on theDEC Rainbow, IBM-PC, Wang PC, Televideo Multi-user system and others.

The speaker considers not only the costs and rewards of standardization, but also a number of techniques and strategies which were used with the software vendor and with Digital support.

PRO/VENIX - UNIX ON DEC'S 16-BIT WORKSTATION
P047 FRIDAY 1:00PM 1 HOUR

PRO/Venix is a full System V UNIX* implementation for DEC's Professional 350 and 380 16-bit workstations. In addition to the standard UNIX utilities PRO/Venix includes support for the Professional's bitmap graphics.

This session will provide an overview of the PRO/Venix system including the advantages and disadvantages of running the UNIX operating system on the Professional workstation.

If you are using a Professional Workstation or are looking for a UNIX based workstation this hour is aimed at you.

RSTS

RST

RSTS

Chairman
Charles Mustain
Stark County School System
Louisville, OH

Symposium Coordinator
Scott W. Pandorf
Kittle's Home Furnishings
Indianapolis, IN

Assistant Symposium Coordinator
Wef Fleischman
Software Techniques
Cypress, CA

Newsletter Editor
Open

Library Representative
Susan Abercrombie
Ventrex Laboratories Inc.
Portland, ME

DEC Counterpart
Kathy Waldron
Digital Equipment Corporation
Merrimack, NH

Pre-Symposium Seminar Coordinator
Bruce Gaarder
Macalester College
St. Paul, MN

Wish Lists Coordinator
Neal E. Goldsmith
Software Techniques, Inc.
Cypress, CA

Vice SIG Chairman
Wish Lists & Tape Copy Coordinator
Philip Hunt
System Industries
Milpitas, CA

EDUSIG Liaison
George Wyncott
Purdue University Computing Center
W. Lafayette, IN

RSTS Product Planning Coordinator
Errol E. Ethier
Information Design and Management, Inc.
Shrewsbury, MA

Members-At-Large
Ed Beadel
Instructional Computer Center
Oswego, NY

Scott Daily
Great Lakes Chemical Corp.
W. Lafayette, IN

Mark Gilmore
Cal State University
Long Beach, CA

Mark Hartman
Jadtec Computer Group
Orange, CA

Jeff Killeen
Information Design & Management
Hopedale, MA

Newton J. Munson
Rochester Institute of Technology
Rochester, NY

RSTS/e
PUBLIC SYSTEM
Dialup 1200/2400
[205] 536-2690

Hermit File Xfer

BULLETIN BOARD
SYSTEM

RSX

MULTI-TASKER

DECUS

RSX

RSX

Chairman
Dan Eisner
Perkin-Elmer Corp.
Garden Grove, CA

Symposium Coordinator
Rick Sharpe
Toledo Edison
Toledo, OH

Pre-Symposium Seminar Coordinator
Hans Jung
Associated Press
New York, NY

Communications Committee Representative
Allen Bennett
Lear Siegler Rapistan
Grand Rapids, MI

Newsletter Editor
Bruce Mitchell
Machine Intelligence &
Industry Magic
Byron MN

Store Coordinator
Jim Hopp
Carlton Financial Computation
South Bend IN

Session Note Editor
Burt Janz
Northern Telecom Inc.
Concord, NH

Librarian
Glenn Everhart
Mt. Holly, NJ

Campground Coordinator
Jerry Ethington
Prolifix Inc.
Frankfort, KY

DEC Counterparts
Lin Olsen
Digital Equipment Corporation
Nashua, NH

Dick Day
Digital Equipment Corporation
Nashua, NH

Working Group Coordinator
Sharon Johnson
Epidemiology
Minneapolis, MN

Working Group Chair
Evan Kudlajev
Philadelphia Electric Co.
Philadelphia. PA

RSX Group Chair Software Clinic Coord.
Roy S. Maull
U.S. Air Force
Offutt AFB, NE

Software Clinic Coordinator
Bruce Zielinski
RCA
Moorestown, NJ

Volunteer Coordinator
Gary Maxwell
U.S. Geological Survey
Menlo Park, CA

Multi-Processors Working Group Coordinator
Bruce Mitchell
Machine Intelligence & Indus. Magic
Hudson, WI

SRD Working Group Coordinator
Bob Turkelson
Goddard Space Flight Center
Greenbelt, MD

Accounting & Performance Working Group Coord.
Denny Walthers
American McGaw
Irvine, CA

Menu Coordinator
Ed Cetron
Center for Biomedical Design
Salt Lake City, UT

Members-At-Large
Jim McGlinchey
Warrenton, PA

Jim Neeland
Hughes Research Labs.
Malibu, CA

Anthony E. Scandora, Jr.
Argonne National Laboratory
Argonne, IL

Ralph Stamerjohn
Creve Coeur, MO

The RSX Multi-Tasker
September, 1986

"All the News that Fits, We Print"

Fine Realtime Commentary Since 1975

Table of Contents

The Editor's Corner

Bruce R. Mitchell

This is a particularly interesting issue for readers interested in improving the performance of your systems. There are two articles dealing with the topic; one by Kreigh Tomaszewski on general system tuning, and one by the editor on improving performance of the M-Plus print symbionts. There's also a very interesting article in the Bag of Tricks which I have already found useful.

Without further ado, here's this month's editorial broadside.

----- The "New DECUS" and the "Old Guard" -----

There has been much *sub rosa* discussion within the Society over the last year on the so-called "New DECUS" as opposed to the "Old Guard". Much smoke has been generated. The Editor perceives two main viewpoints among the discussors, and presents them here for consideration by readers who may not be *au courant* with such matters.

The "New DECUS" viewpoint: DECUS and Digital have come a long way since the formation of the Society. The days of the single-user, isolated system and the technically oriented user are almost ended. Today's machines are large, multi-user, and networked; the users of those machines are, in general, technically unsophisticated. DECUS should change its orientation to reflect this change in the Digital user base.

The "Old Guard" viewpoint: DECUS was established as a user society for end users - viz., programmers - and promotion of technical excellence in those users. The majority of those users are not particularly interested in the latest office automation products from Digital; they are interested in accomplishing a job of work. DECUS should retain and promote its orientation toward technical excellence among the users of Digital machines.

This is a matter which affects all of us as members of the Society. The Editor withholds his personal opinion, having no wish to gore anybody's ox in print. He does, however, welcome *constructive* comments from readers on this topic. (The Editor welcomes comments from readers on almost *any* topic!)

----- Submitting Articles to the Multi-Tasker -----

Please, oh pretty please, submit machine readable media if you can. RX01/RX02 floppy or 800/1600 BPI 9 channel magtape are best. Any format is acceptable except ROLLIN, PRESRV or VMS backup. BRU and DOS FLX formats are well-liked by the Editor's tape drive.

Submissions which aren't machine readable take longer to get into print. The editor is lazy and types mass quantities only once a month when progress reports are due.

If you preformat a submission in RUNOFF format, please set left margin 10, right margin 75, and when changing margins use incremental changes rather than absolute. The editor will bless you for the consideration.

Send all submissions to:

Bruce R. Mitchell
Machine Intelligence and Industrial Magic
PO Box 816
Byron, MN   55920

----- Answer to Last Month's Quiz -----

The first letter of every paragraph in the Editor's Corner.

----- And That's The Way Things Are -----

... this month in Pool Lowbegone, where all the programmers smell strong, all the secretaries are good-looking, and all the task runtimes are above average.

The Bag of Tricks: MACRO-11

Jim Bostwick
Cargill, Inc.
Minneapolis, MN

This month's article is on round-robin scheduling in user tasks.

When a user task is serving more than one physical device simultaneously, a common approach is to assign each device a logical unit number (LUN), assign an unique event flag to each LUN (usually equal to the LUN), queue a read-without-wait to each device, and wait for the logical OR (WTLO$) of all the event flags. When the WTLO$ completes, at least one unit has completed I/O; each event flag is then checked to see who completed.

There are a couple of large holes in this approach. One is that if a large number of devices are served, it's easy to run out of event flags. Another is that if more than 16 event flags are required, a single WTLO$ doesn't do the trick. A final objection is that if searching always starts at unit 0, and there are 256 units, unit 255 may never be serviced by the task.

This is a better approach. Only one event flag is required, and every unit gets equal service. Read-without-waits are hung on every LUN, all of which set the event flag IOFLAG. When any I/O completes, a round-robin search algorithm is used to equalize service priorities - the unit just serviced gets lowest priority while everybody else gets a shot at the CPU.

Presumably, this type of code is used to serve many identical devices, so a single QIO DPB suffices. All that is necessary is to remember that when a read QIO is hung on a logical device, the DPB for that device must be modified to change the LUN (Q.IOLU) and the I/O status block (Q.IOSB) parameters. It is essential that read-WITHOUT-wait QIOs be used if this code is to work.

All I/O status blocks are kept in a contiguous block of words at IOSTAT. Note that the unit activity determination uses an RSX feature; the I/O status block is zero when a QIO is pending, and nonzero when it completes.

The value of ROBIN - the round-robin search variable - is unimportant when the task is initialized. It is self-correcting if garbage is left in it.

```
WAITIO: WTSE$     IOFLAG            ; Wait for I/O to complete

IOSTAT: .BLKW     <MAXUNT*2>        ; I/O status block array


          ... code to hang read QIOs on every served device ...

;         Top of recurrent program execution.  We get all I/O
;         through this I/O scanning routine.  All units get equal
;         access to the program due to round robin scheduling.

10$:      INC     ROBIN             ; Increment RR to next unit
```

```
        CMP     ROBIN, #MAXUNT    ; Trying to look at illegal unit?
        BLOS    20$               ; If not, bypass the resetting

        MOV     #1, ROBIN         ; Reset to look at unit 1 again

20$:    MOV     ROBIN, R0         ; Load base of search into R0

30$:    MOV     R0, R1            ; Copy searched unit number to R1
        DEC     R1                ; Change LUN to 0-base for lookup
        ASL     R1                ; Multiply it by 4, because I/O
        ASL     R1                ;  status blocks are 4 bytes
        TST     IOSTAT(R1)        ; Is I/O complete for this unit?
        BNE     50$               ; If so, go process it

        INC     R0                ; Set up to look at the next unit
        CMP     R0, #MAXUNT       ; Trying to look at illegal unit?
        BLOS    40$               ; If not, bypass the resetting

        MOV     #1, R0            ; Reset to look at unit 1 again

40$:    CMP     R0, ROBIN         ; Have all units been searched?
        BNE     30$               ; If not, go search the next one

        Nobody's active, so sleep and wait for an I/O to complete

        DIR$    #WAITIO           ; Wait for any I/O to complete
        BR      20$               ; And go process it when it does


        Round-robin scan completed.  <R0> is active.  Process it.

50$:    ... I/O completion processing code ...
        ... replace I/O on unit just processed as last thing ...
        BR      10$               ; Go back, check for I/O again
```

Improving M-Plus Spooler Performance

Bruce R. Mitchell
Machine Intelligence and Industrial Magic
PO Box 816
Byron, MN 55920

A local site recently purchased several "Brand Q" laser printers to replace their electrostatic printer/plotters. Their initial intention was to attach the printers from the task generating the plots, rather than spool the laser printers. However, after some discussion of the relative merits of each method, it was decided to try spooling the printers.

Some difficulties were encountered in the spooling subsystem. Most of them were readily overcome, and the printers had been running for some time before it was noticed that there was a great deal of random seeking occurring on the disk containing the plot files. It was discovered that when the print symbionts (the "spoolers") were stopped, much of the seeking disappeared.

Further investigation into the problem revealed two surprises; first, that the print symbionts are too heavily overlaid, and second, that the print symbionts do not do big-buffered or multibuffered reads.

These were interesting discoveries, to say the least. The overlaying issue was addressed first, with gratifying results.

The print symbionts are PR:5 tasks. Even so, there are still 3 APRs available to the task. It was found that LPPFSL.TSK, as built, occupies only 3072 of the available 12288 words. But, with over 9000 words available, the taskbuilder map shows that the task is partitioned into 10 overlays, overlaid 2 levels deep, and some of the overlays are only 500 words in size. This is inefficient at best under M-Plus, where FSL tasks and large memory are the norm.

With little effort, the overlay descriptor for the task was modified as follows:

```
        .NAME   LPP
        .ROOT   LPP-[1,24]LPP/LB:RECEIV:POTS:IOPRT-E-Q-*(A,B,C,D)
E:      .FCTR   SY:[1,54]RSX11M.STB
Q:      .FCTR   [1,24]QMG/LB:QMGSYM
A:      .FCTR   [1,24]LPP/LB:INIT:JOBSTR:JOBEND
B:      .FCTR   [1,24]LPP/LB:FILPRO:CONT:ERPRT:PRT-B1
B1:     .FCTR   [1,24]QMG/LB:FPRIV:PRCO
C:      .FCTR   [1,24]LPP/LB:FLPAG:FLINI:JBINI:FLGEN
D:      .FCTR   [1,24]LPP/LB:PRTDN:DONE
        .END
```

This new overlay structure reduced the overlay level from 2 to 1, reduced the number of overlays from 10 to 4, reduced the size of the task image from 38 to 27 blocks, and reduced the number of overlay reads from the system device as a side benefit; this was very useful as many files were being spooled. The in-memory size of the task increased 480 words, from 3072 to 3552 words. It was felt this was a very small price for the return on those 480 words.

It was theoretically possible to flatten the symbiont completely, but when this was attempted, the Taskbuilder didn't like it. The GBLDEFs in the taskbuild command file started to fail.

Though the spoolers were running faster, they were still generating unacceptable traffic on the spool disk. The tasks generating plot files were generating quite enough random traffic on the spool disk without the spoolers making it even worse. The possibility of big-buffering in the symbionts was considered.

It was found that big-buffering in the symbionts was indeed feasible. Three one-line changes were necessary in symbiont source code.

In the distribution kit source file [121,10]RECEIV.MAC, the line:

```
        FSRSZ$  1,512.,BUFFER
```

which defines an FSR1 block storage area of 1 block in size, 512 bytes, in PSECT BUFFER, was changed to:

```
        FSRSZ$  24.,, BUFFER
```

which defines an FSR1 block storage area sufficient for 24 disk blocks.

In the distribution kit source file [121,10]FILPRO.MAC, the line:

```
        .MCALL  OFID$R,DIR$
```

which calls the file open and directive macros, was changed to:

```
        .MCALL  OFID$R, DIR$, FDBF$R
```

so that the runtime FDBF$ (define file buffer area parameters) macro was available. Insertion of the first line in the sequence below was also necessary to restore the buffer size before each file open, as this word is destroyed by FCS after the file is opened:

```
        FDBF$R  R0,, #<24.*512.>        ;  Read 24 blocks per I/O
        OFID$R  R0                      ;  OPEN THE THING
```

This increased the task size to the close order of 10K words and caused a marked improvement in disk traffic. Before this change, the spoolers "fought" with the application tasks for disk I/O, with consequent degradation of I/O bandwidth on the spool disk. After the change, I/O bandwidth increased greatly.

A pleasant side effect of these changes was that it was possible and actually necessary to lower the spooler priority to 51 from the normal 70. At priority 70, the new versions of the spoolers became so much faster that they ate up all the time the system made available.

One final warning: The information in this article was gathered on, and is correct for, RSX-11M-Plus V2.1 Update E. V3.0 may have changed the files to which reference is made.

System Tuning Under RSX-11M-Plus

Kreigh Tomaszewski
Amway Corporation
7575 E. Fulton Road
Ada, MI 49355

The graphs which accompanied this article, regretfully, did not reproduce well enough for inclusion. My apologies to the author. --- The Editor

System tuning under RSX-11M-Plus can mean many things. At Amway Corporation, it means minimizing terminal response time while maximizing throughput of the machine. Before going into how we meet these often conflicting goals, let me tell you a little about the environment.

Amway's world headquarters (and main manufacturing facility) employs about 3500 individuals in a complex of connected buildings over a mile long. Our computer center is near the middle, and houses a large IBM shop (3081 and 3033 processors) in the "big computer room". In the "small computer room" we have one PDP-11/70 and other equipment such as an IBM System 38.

The PDP-11/70 is configured with 3 Mb of memory, an RM05 system disk, 3 RA81 user disks and 2 tape drives - TU77 and TE16. The machine has 3 DL single line terminal interfaces, a 16 line DH and enough DZs to bring the total to TT102: . Two DUP11 interfaces provide Bisync connections to the IBM network for both RJE and 3270 terminal emulation.

We use the DEC processor as an in-house timesharing service. It is used primarily by the Research and Development, and Quality Assurance areas. Because it is a timesharing system, we have very little control over the processing load placed on it by the users. To try to make sense of the load, we have defined several classes of users and support.

First are the production applications which have been justified, documented, formally reviewed and approved. They have an elevated priority on the machine. Word processing of all the control documents used in our manufacturing processes is one such production application; often there are 8 W/P users active, and half of them are list processing. We use WORD-11 for our production word processing.

However, since this is primarily a timesharing system, production applications are usually implemented on the IBM systems. As an example, a daily set of approved new or updated documents is transferred from the 11/70 to the IBM system for recall by the users. Only document maintenance is done on the DEC system.

Production applications are those which cover more than one department, or use software purchased from a vendor. Production applications often have their own UFD group, so a user may have several accounts; one for general departmental use, and one or more cross departmental application accounts. Production accounts are backed up more often than the other types of accounts on the system.

The next class of user is the departmental user. Each department is a separate UFD group on the system. Departments write and use their own applications. Many tools are available for this, such as BASIC-11, Basic-Plus-2, Fortran 77, Macro-11 and Datatrieve. A variety of editors is also available - EDT V2 and V3, EDI and KED. We also offer FMS-11 to ease screen handling.

These users all run at the system default priority, with the editors slightly elevated to improve response time. We run a lot of statistics programs, both SPSS and user-written, and we support a spreadsheet package. None of these users are privileged.

Both of these classes of users can receive support for any documented program they wrote, or for any supported system component, by calling our HELP desk. A call to one number resolves hardware problems, report distribution problems or any other problem/question dealing with data processing. The HELP desk functions as a miniature support center, logging the call and assigning it to the proper person for resolution. Management receives daily status reports on all problems, and it has become very convenient for both IBM and DEC users.

If the HELP desk support personnel - currently 2 individuals in the EDP Technical Support department - cannot resolve a problem, they take it to DEC or the appropriate vendor for further investigation. We run on the theory that support software is documented, so - if reality and the documentation disagree, one of them needs adjusting. If the software came from a vendor, we report it and let the vendor make the adjustment. (NB: Try to follow the vendor's rules when you do this; they usually know what information they need to correct your problem and they tell you how to report it. If that fails, try other methods.) Remember, if the vendor doesn't know about it, they can't fix it - and you have to live with it.

The third class is unsupported, or "mirror image" software - i.e., if you want to see who supports it, look in the nearest mirror. This consists mostly of DECUS software, and includes packages such as TECO, SOS, Pascal, C, Forth and many system utilities. These all run at a priority below the default so that they don't crowd the supported software.

The final class is Batch users in the Batch queues. No matter what they are running, if it's at a priority higher than 48 (remember that the default is 50), it is reduced to priority 25. Batch users get whatever time the terminal users aren't using. This minimizes terminal response time. This is done with a monitor developed in-house (available from the DECUS Program Library, order # 11-792) known as ZZZ. We currently run 3 batch streams.

We have over 300 users known to the system, and over 200 of them use the system in any given month. We have modified HELLO to limit the number of logged-on users to 20, excluding batch processes, at any given time. This value was developed from experience. Override accounts are available which allow logons when the system is at max users. This does occasionally cause some trouble at peak times, but it has proved to be a workable solution to insuring relatively good response time once the user is logged on.

There are many approaches to system tuning. We base ours on the theory that there are only so many CPU cycles available in a day; and we have days when we use <u>all</u> of them. These cycles can be divided into the following categories:

o <u>Null time</u>; the processor is idle
o <u>User work</u>; the CPU is running a user task or utility
o <u>System work</u>; service such as retrieving a file record
o <u>System overhead</u>; overhead such as accounting, task scheduling

Null time is something you hope never runs out, because when it does, everything slows down. There are two solutions; add more processing power, or reduce some other component of the total time being used. System overhead is the logical choice because we are trying to maximize user work.

Isolating system overhead from system work is not easy. The best solution we have found is to monitor the ratio of user to system state. If the ratio increases after a tuning change, we have reduced system overhead. If it decreases, we have increased it. Because our system is operated around the clock, we usually are forced to determine this ratio statistically rather than by running benchmarks.

One of the worst causes of system overhead is the Shuffler. When a memory allocation request fails, the Shuffler is requested to rearrange the tasks in memory to make it possible for the failing request to succeed. Fixing tasks in memory can cause such fragmentation. If you must fix tasks, install them during startup so that they are placed on the lower edge of the GEN partition, or make a separate partition for them. Adding memory to your system is another (more expensive) solution to the problem.

Another cause of system overhead is checkpointing. This occurs when a task is temporarily inactive, waiting its turn to execute, and a higher priority task requires memory. This has the effect of copying the lower priority task to disk and restoring it to memory at a later time when memory is available. No productive work is done during this period.

The Shuffler uses checkpointing to move tasks about in memory. Another cause of checkpointing is a task extending itself and colliding with another task loaded above it in memory; this causes the task to be checkpointed and reloaded in a new location where the extension can occur. If you expect a task to extend itself, install it with an appropriate increment instead. This prevents checkpointing caused by extension. The Taskbuilder is a good example of a task which extends itself.

We find that the file system has overhead which can be reduced. First: Mount each disk with its own ACP. Use the /ACP=UNIQUE switch to do this. This maximizes the amount of buffer space available for each disk and prevents ACP cache flushing.

Second: Expand the number of control blocks and buffers in memory by increasing the LRU and WIN counts above the defaults. This is particularly important on big disks with many UFDs.

Finally: Look at the ACP taskbuild command file in [1,20] and see if it is possible to enlarge the buffer area following the comments. The ACP goes to primary POOL when it runs out of internal space, so put as much buffer space in the ACP as possible.

Another tuning step is to attempt to spread the I/O across all of the devices on the system. Checkpoint files are used in the order allocated, and checkpoints task I/O. Allocate checkpoint space starting with your least used disk; consider reallocating the checkpoint space on the system disk to put it into a less used position to improved task loading by reducing the I/O load on the system disk. Just make sure that you keep one checkpoint space allocated while you reallocate the system disk.

Another possibility is to use a logical assignment for each group in the account file and make those logical assignments globally at system startup. This allows you to move a group of users transparently at any time to balance I/O.

Yet another approach is to reduce user work and system work in the applications. Preallocate files to prevent the overhead of a file extension while the task is executing. Make sure that you use efficient bucket sizes. Use common sense when designing applications. RMS indexed files must be reorganized periodically, as do entire disk volumes. There are often two ways to ask the system to do something; make sure you are using the more efficient methods.

Terminal I/O should also be considered. A DH11 uses DMA output when possible, reducing the number of interrupts that must be services. Put terminals with heavy activity on DH11s if you have them; if not, put them on the first lines of each DZ11. Consider standardizing terminal speeds to reduce bursts of interrupts that make for uneven response time when high speeds are used. We have settled on 4800 baud for production and 2400 for non-production terminals. Users are free to change this (if they know how), but they must do so every time the system is rebooted.

Another source of overhead is task loading and overlay loading from disk. If a task is to be used by several users, consider making it a multiuser task with shared code that must be loaded only once. Strive to reduce the number of overlays. We do not have users on the system disk, only support personnel; all software is loaded from the system disk for production applications. This also includes the use of resident libraries to further reduce the size of the code needed to be loaded when the application is run.

An additional source of overhead is intertask communication. Some of it is intentional, such as send data directives. There is often nothing to be done about it, but keep it in mind as a source of overhead. Some may be unintentional, such as offspring control blocks. These are created when a task is started and released when the task exits or emits status. They occupy POOL, often causing fragmentation - another source of overhead. If the OCBs are unnecessary, it is possible to emit status to make them go away. The OCB does not go away when the parent task exits.

POOL is another source of overhead. If it becomes fragmented, the system must do more work searching for a fragment large enough to service POOL requests. POOL is a limited resource even under RSX-11M-Plus with secondary POOL. Long running tasks such as batch processors and spoolers should be started before there is much activity so that their control blocks sit on the edge of POOL, reducing fragmentation.

The round-robin scheduling routines are another source of overhead because of context switching. Don't be afraid to change the interval to a longer time. Use the priority scheme instead. A useful technique for doing this is to compile a list of all the tasks to be run in the system. Rank the list in priority order by asking the question, "If I could only run one task, which task is it?". Move that task to a second list, repeating the process until all tasks are ranked. Don't forget to include the default priority tasks, although it is strongly recommended that the system tasks keep their relative position in the list.

Now go through the ranked task list and assign priorities, working up and down from the default, so that task priorities are evenly spaced. Then check whether any heavily used tasks are CPU bound - such as the Taskbuilder - and consider reducing their priority slightly to minimize the impack on terminal users. It's all right if some tasks have the same priority; just don't make the number of tasks with equal priorities too large. Now go back and re-VMR the system.

Finally, look for bottlenecks in system performance. Do your disks spend all day hopping about the computer room? Turn on disk optimization or play with the optimization parameters until they perform better. Say that the overlay rate goes up; is it due to one task being run by everyone? If so, see if the task can be flattened, memory overlaid, or broken into two cooperating tasks.

Keep good records of system activity for analysis and review. Use RMD and the system accounting files, or use a system monitor such as SPM-11. Feed the raw data into a statistical package and see what comes back.

System tuning is a continuous activity. System conditions evolve, and tuning must be adjusted to match the changing demands on the system. Keep records when changes are made. Graph tuning data; look for short and long term cycles. Look for trends; slopes and curves that flatten out usually indicate bottlenecks that can sometimes be widened to allow more throughput. Develop a profile of the average task, and see how it is increasing in sophistication by its growing demands for system resources.

RSX is a very robust operating system with many knobs to turn. The task of system tuning is learning how to turn those knobs to make the system play your tune.

The Inside Scoop on BRU

Steve Realmuto
Digital Equipment Corporation
Nashua, NH

The following article is a collection of handouts used during the general session on BRU at the Spring symposium in Dallas. While no supporting text is available, the material is of such interest that the Multi-Tasker is presenting it as it stands. In some areas these figures may be difficult to read; this is because they are second generation copies of the handouts and not reproduced from the originals. --- The Editor

The Inside Scoop

on

BRU

Steve Realmuto
Digital Equipment Corporation
Nashua, NH

The Inside Scoop on BRU

- o Past, Present, and Future of BRU
- o BRU 'Gotchas' and Helpful Hints
- o Helping Us to Help You

Past, Present, and Future of BRU

Where BRU has been...

- BRU was developed as a replacement for DSC

- Major goals were:

  - increased speed and efficiency
  - improved reliability
  - greater flexibility

Past, Present, and Future of BRU

Where BRU is...

- Reliability continues to improve

- Highlights of corrections for RSX-11M-PLUS V3.0:

  - backup multi-header files across tapes
  - restore multi-header files to mounted disk
  - recover from errors writing tape
  - correct bad block processing for /BAD:xxx
  - prevent /IMAGE:SAVE to tape
  - allow /INITIALIZE of DU: devices under VAX-11 RSX V2.0

- Highlights of corrections for RSX-11M-PLUS V3.0 Update B:

  - synchronization problem between backup and verify passes
  - display I/O error codes as signed values
  - enable /UFD to create directories under VAX-11 RSX V2.1
  - process last-track devices correctly under VAX-11 RSX V2.1

Past, Present, and Future of BRU

Where BRU is...

- Highlights of corrections for RSX-11M-PLUS V3.0 Update C:

  - recover from errors writing tape labels
  - eliminate erroneous messages during /IMAGE:RESTORE

Past, Present, and Future of BRU

Where BRU is...

- Features added to support RSX environment

- New support for RSX-11M-PLUS V3.0:
  - named directories
  - logical names
  - decimal version numbers
  - automatic skip past bootable system image

- New support for RSX-11M-PLUS V3.0 Update C:
  - large disks

Past, Present, and Future of BRU

Where BRU is going...

- o Continuing to improve BRU's reliability is a major goal for the next release.

- o Items under consideration include:

  - preventing the /INITIALIZATION of a device mounted by another user

  - preventing an /APPEND to other than the first tape

  - improving behavior when devices are not mounted properly

  - adding an /IDENTIFICATION qualifier to display BRU's version

  - adding support for named directories to Standalone BRU (BRUSYS only)

BRU 'Gotchas' and Helpful Hints

- o Different versions of BRU exist for each member of the RSX family.

  - RSX-11M
  - Standalone BRU (BRUSYS and BRU64K)
  - RSX-11M-PLUS and Micro/RSX
  - VAX-11 RSX

- o Mount status of I/O devices must agree with qualifiers used.

  - Tapes are always mounted /FOREIGN (or unmounted on RSX-11M)

  - /MOUNTED refers to the input disk

/MOUNTED	=	mounted Files-11
not specified	=	mounted /FOREIGN

  - /NOINITIALIZE refers to the output disk

/NOINITIALIZE	=	mounted Files-11
/INITIALIZE	=	mounted /FOREIGN

BRU 'Gotchas' and Helpful Hints

- BRU can only restore an /APPENDed backup set from the first tape.

- BRU always restores files to their original UFD.

- File specifications entered after the output device are concatenated to the list of input file specifications.

- BRU will only backup empty UFDs during a full backup. Empty UFDs are re-created only during a full restore if the output disk is initialized.

- After a /DIRECTORY, the tape is left positioned in the middle of the backup set. Therefore, a subsequent BRU command should include the /REWIND qualifier.

- The names following the /BACKUP_SET, /IN_VOLUME, /OUT_VOLUME, and /TAPE_LABEL qualifiers should not be enclosed in quotes because the quotes will become part of the name.

BRU 'Gotchas' and Helpful Hints

- Creating synonym directories will cause the files in those directories to be backed up once for each directory. This may cause problems when restoring the files to a mounted volume.

- Under VAX-11 RSX, BRU cannot continue onto a second tape on MS: devices (TU80, TSV05, TK25, TS11) due to a problem with the VMS TSDRIVER.

Helping Us to Help You

- Submit an SPR with all the information we need to investigate the problem.

- All SPRs on BRU should include:

  - instructions to reproduce the problem.

  - the version of BRU used.

  - a console log showing the commands used to mount the devices involved, the BRU command line, and all messages displayed by BRU.

  - any media necessary to reproduce the problem.

  - a printout of any error log entries during the period of time BRU was active.

THE mini-tasker

DECUS
RT-11 SIG NEWSLETTER

RT-11 SIG
DECUS
U.S CHAPTER

RT

RT-11 SIG Steering Committee

RT-11 SIG Chairman
John T. Rasted
JTR Associates
58 Rasted Lane
Meriden CN 06450
203 / 634-1632

Newsletter Editor
COBOL Contact
Bill Leroy
The Software House, Inc.
P. O. Box 52661
Atlanta, GA 30355-0661
404 / 231-1484

Standards Coordinator
Kenneth L. Aydlott
Teledyne Hastings-Raydist
P. O. Box 1275
Hampton, VA 23661
804 / 723-6531

Tape Copy Generation Contact
Ralston Barnard
Division 7523
Sandia Laboratories
Albuquerque, NM 87185
505 / 844-5115

APL Contact
Doug Bohrer
Bohrer and Company
903 Ridge Road, Suite 3
Wilmette, IL 60091
312 / 251-9449

MACRO Contact
Nick Bourgeois
NAB Software Services Inc
P. O. Box 20009
Albuquerque, NM 87154
505 / 298-2346

TECO Contact
Product Planning Contact
John Crowell
CROWELL Ltd.
145 Andanada
Los Alamos, NM 87544
505 / 662-3893

DECNET Contact
Ken Demers
Adaptive Automation
5 Science Park
New Haven, CT 06511
203 / 786-5050

RT-11 Hardware Contact
C Contact
Carl Lowenstein
Marine Physical Lab
Scripps Inst Oc'graphy
San Diego, CA 92152
619 / 294-3678

Wish List Contact
UNIX Contact
Bradford Lubbell
L. A. Heart Lab
UCLA A3-381 CHS
Los Angeles, CA 90024
213 / 825-9290

TSX Contact
Jack Peterson
Horizon Data Systems
1899-E Billingsgate Cir
Richmond, VA 23233
804 / 740-9244

FMS Contact
CommComm Representative
Susan Rasted
Software Dynamics Inc.
85 Barnes Road
Wallingford, CT 06492
203 / 265-2226

Symposia Coordinator
Ned Rhodes
Software Systems Group
1684 Gude Road
Rockville, MD 20850
301 / 340-2773

Tape Copy Distribution
RT DECUS Library Contact
Tom Shinall
General Scientific Corp.
1684 East Gude Drive
Rockville, MD 20850
301 / 340-2773

Pre-Symposia Seminar
RT-11 Suite Manager
Bruce Sidlinger
Sidlinger Computer Corp
4335 N.W. Loop 410, #209
San Antonio, TX 78229
512 / DIG-ITAL

BASIC Contact
Ed Stevens
E M D A Inc.
77 N Oak Knoll #104
Pasadena, CA 91101
818 / 795-5991

CAMAC Contact
J. W. Tippie
Kinetic Systems, Inc.
11 Mary Knoll Drive
Lockport, IL 60441
815 / 838-0005

LUG Contact
Personal Computers
Bill Walker
Monsanto Research Corp.
P. O. Box 32
Miamisburg, OH 45342
513 / 865-3557

RUNOFF Contact
FORTRAN Contact
Robert Walraven
Multiware, Inc.
139 G Street, Suite 161
Davis, CA 95616
916 / 756-3291

Notes from the RT-11 World

Copyright's

All copyrights in the RT-11 Mini-Tasker belong to the owner/submitter of the material in the Mini-Tasker, and not to the RT-11 SIG, DECUS or DEC. If you have a question about any article in any past or future issue of the Mini-Tasker, please contact the author directly - not me - and not DECUS.

We DO solicit signed articles for insertion in the Mini-Tasker, on or about bugs, features, nifty things, etc., all about the RT-11 operating system and its environment. Write it up, send it to me (with a note to rewrite if you wish) and I will try and get it in an upcoming issue of the Mini-Tasker.

- -

Trivia

With thanks to Diana Miller, RT-11 Product Manager, DEC put together an interesting DECUS program last year on PDP-11 trivia. Following are some interesting RT-11 facts of life.

1. On what PDP-11 processor(s) will a SWAB not change the overflow bit in the PSW?
2. What does EMACS stand for?
3. What does MINCE stand for?
4. How many toggle switches were on the first PDP-11?
5. DIGITAL opens in Maynard, MA with three employees and 8500 square feet of production in a converted woolen mill. What was the month and year?
6. Where would one visit to "walk the halls" of the PDP-11 birthplace?
7. What does DSM stand for?

- -

November Issue Submittal Deadline Advanced

Judy Arsenault of the DECUS U. S. Activities staff has announced that her printers deadline for the November issue is September 23. Therefore I need your submissions by FRIDAY, SEPTEMBER 19th at the latest.

- -

Physical Address of FORTRAN Virtual Arrays

Jack Crowell has an interesting article on how to determine the physical address of FORTRAN virtual arrays, and a few important reminders. The software supplied is in the public domain, but the author assumes no responsibility for the use or reliability of this software on equipment which is not supplied by someone.

RT-11 SIG Tape - On-line Distribution

Tom Shinal has some more notes in this issue on the electronic distribution of the RT-11 SIG tape, as well as its usage. Tom also reports the following:

> During May and June, I collected the following statistics:

TAPE	LOCATION	LOG-INS	CONNECT TIME	CPU TIME
FALL '85	ANAHEIM	29	38 hrs 26 min	40 min 01 sec
SPRING '86	DALLAS	16	30 hrs 24 min	2 hrs 12 min 31 sec

> Looking at the statistics, it becomes obvious that during May, most of the users were browsing and did find some file transfers, but in June, a tremendous amount of file transfers took place. This may be because of the vast interest in getting the latest and greatest DECUS C compiler.
>
> It appears that the electronic transfer experiment is a success and will be continued so long as there is disk available and there is no abuse of the service.
>
> -- Tom Shinal

New versions of Fortran and Pascal now available

Diana Miller, our beloved RT-11 DEC REP, reports the following: FORTRAN-77/RT-11 (QA609-xx) V5.0A and PASCAL-2/RT-11 on the PRO (QAAD5-C3) are both available and shipping from the SDC.

1986 Fall Symposia - San Francisco

Please read Ned Rhodes interesting notes on scheduling for the San Francisco Symposia, October 6 - 10, 1986. RT-11 is sponsoring 22 sessions this time for RT-11 and TSX-Plus users. Please plan to attend.

Diana Miller will have a MicroPDP/whatever in the campground in a BA123 box (the one with the wheels), configured with an RX50, TK50, and hard disk, along with a VT220.

MS Handler Patches?

Do any of you out there know of any patches to the MS handler for RT-11 V5.01 and V5.02, and TSX-Plus V6.01 that will make certain tape couplers function properly? If so, contact Nick Bourgeois at (502) 298-2346.

Trivia - Answers

1. PDP-11/15, PDP-11/20.
2. Edit MACroS.
3. Mince is not complete EMACS.
4. 25.
5. August, 1957.
6. Mill Building 5, floor 2.
7. Digital Standard MUMPS.

And finally, I am always looking for something to print.

Please send your submissions to The Mini-Tasker (RX-01, RX-50, or pieces of paper) to me at:

Bill Leroy
The Software House, Inc.
2964 Peachtree Road, NW #320
P. O. Box 52661
Atlanta, GA 30355-0661

404/231-1484 or DCS (LEROY)

LOCATING VIRTUAL ARRAYS

John M. Crowell
Crow4ell, Ltd.*
Los Alamos, New Mexico

At nearly every DECUS symposium for the last three-or-four years, I have been asked how to determine the physical address of FORTRAN virtual arrays. The principal reason for wanting to know where in memory a virtual array resides is so that one can program one's DMA device to read data from or write data to the virtual array without going through the WRITE or READ process. The function subroutine given here will return to the FORTRAN program the physical address of a virtual array in all RT-11 environments. I, quite frankly, haven't the slightest idea whether it will work under TSX+.

A few reminders about virtual arrays may be in order here:

1. All VIRTUAL specifications are concatenated. (This is why,for example, you don't have VIRTUAL COMMON and VIRTUAL EQUIVALENCE.)

2. All virtual arrays start on 32-word boundaries. If one VIRTUAL declaration doesn't completely fill the last 32-word chunk, the next declaration will leave a gap, and start at the beginning of the next 32-word chunk. The subroutine given here actually returns the starting address of the requested virtual array divided by 64. Any 22-bit starting address can be specified in 16 bits.

3. Under SJ and FB, virtual arrays begin at address 160000. This makes it possible to have VM: and virtual arrays overlap. Overlapping virtual array space and VM: can be useful in a perverted sort of way, and very unfortunate if you do it by accident. (If VM: is loaded, and you run a program with overlapping virtual arrays, you'll get your wrist slapped. But once your program is running, VM: may be fetched with an OPEN statement without catastrophe.)

4. Under XM, virtual arrays are mapped into virtual address space starting at 160000. So be wary of using XM virtual arrays in programs that try to access the I/O page directly.

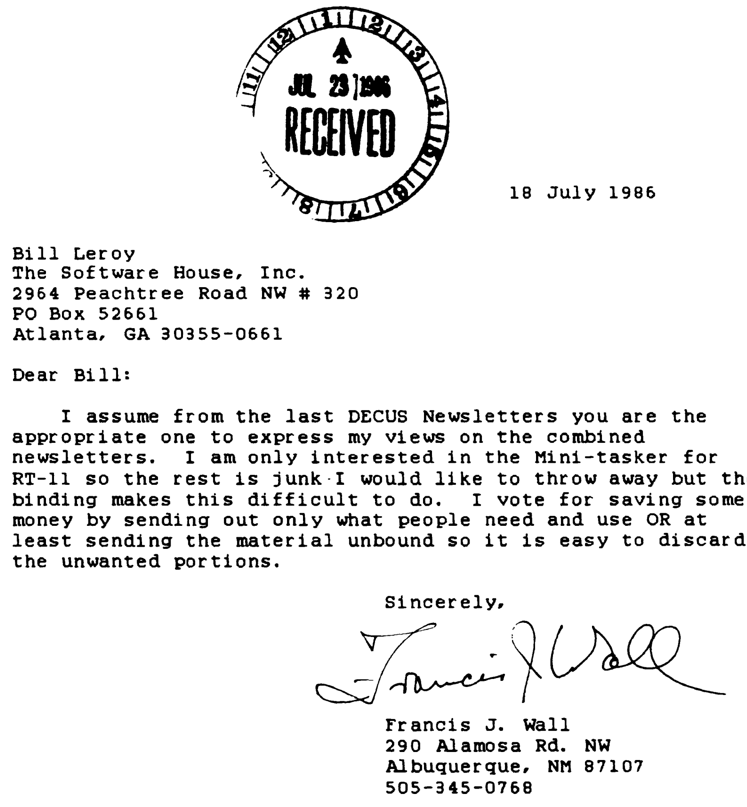
5. FORTRAN-IV and FORTRAN-77 pass virtual array specifications to subroutines differently. (Surprise!) This can make a difference if you are writing MACRO subroutines for use with both.

OK, with that out of the way, here is the subroutine which, when called from your FORTRAN program wi'.l give you the physical address of your virtual array.

* but not very

```
.ENABL  MCL
.MODULE IVADR, RELEASE=JC01, VERSION=1, COMMENT=<Locate Virtual Arrays>

;****************************************************************************
;
;       Author:         John M. Crowell
;                       Crow4ell, Ltd.*
;                       145 Andanada
;                       Los Alamos, NM 87544
;
;       COPYRIGHT (C) NOT-AT-ALL
;
;       This software is furnished under license  for  use  on  any
;       computer system and may be copied with or without inclusion
;       of  this  notice.  This  software,  or  any  other   copies
;       thereof, may be provided or otherwise made available to any
;       other person.  Title to and ownership of the software shall
;       at all times remain in the public domain.
;
;       The information in  this  software  is  subject  to  change
;       without notice, and should not be construed as a commitment
;       by the author, Crow4ell, Ltd.*, or Multiware, Inc.
;       I understand  the  Digital Equipment Corporation  can't  be
;       blamed either.
;
;       The  author  assumes  no  responsibility  for  the  use  or
;       reliability  of  this  software  on  equipment which is not
;       supplied by someone.
;
;       * but not very
;****************************************************************************

; FORTRAN callable routine to find physical address of VIRTUAL array.

;       Calling Method          VIRTUAL VARRAY(n)
;                               I = IVADR(VARRAY)
;
;       Function returns integer value equal to the physical address
;       of the VIRTUAL array divided by 64. (All virtual arrays start
;       on 32-word (64-byte) boundaries.
;
;       If any error occurs (e.g. routine cannot get data on current job),
;       a value of -1 is returned.

;       This subroutine distinguisheds among FORTRAN-IV(SJ/FB),
;       FORTRAN-IV(XM), and FORTRAN-77 virtual array specifications.
;

SYSPTR  = 54            ; Monitor pointer
CONFIG  = 300           ; Offset to configuration word
  $XM$  = 10000         ; XM bit in configuration word
MEMPTR  = 430           ; Offset to memory control block pointers
.PAGE
.PSECT  USER$D,RW,D,LCL,REL,CON         ; F-IV .PSECT convention. Why not?
AREA:   .BLKW   3                       ; EMT parameter area
JOBDAT: .BLKW   12.                     ; .GTJOB data return area

.PSECT  USER$I,RW,I,LCL,REL,CON
```

```
IVADR:: MOV     R1,-(SP)
        MOV     #1600,R1                ; Assume SJ/FB
        .GVAL   #AREA,#CONFIG           ; Get configuration word
        BIT     #$XM$,R0                ; are we running XM ?
        BEQ     3$
        .GTJB   #AREA,#JOBDAT,#-1       ; YES, get info about our job
        BCC     2$                      ; I don't know how this error can
1$:     MOV     (SP)+,R1                ;     occur, but just in case ...
        MOV     #-1,R0
        RETURN

; Find our region control blocks

2$:     MOV     R3,-(SP)
        .GVAL   #AREA,#MEMPTR           ; Get the memory control pointers
        MOV     R0,R3
        .PEEK   #AREA,#SYSPTR           ; Get actual monitor location
        ADD     R0,R3
        ADD     #6,R3                   ; Get address of RCB offset
        .PEEK   #AREA,R3                ; Now R0 = impure RCB offset
        ADD     JOBDAT+8.,R0            ; R0 -> our region control blocks
        MOV     R0,R3
        ADD     #6,R3                   ; VIRTUAL's in second region
        .PEEK   #AREA,R3                ; Get address of this RCB
        MOV     R0,R1                   ; R1 has base address of VIRTUAL's
        MOV     (SP)+,R3

; Now we need to know whether it's FORTRAN-IV or FORTRAN-77
;
; (F77 supports VIRTUALS only under XM, but we are allowing for possible
; SJ/FB VIRTUALS in the future [not likely], and we are assuming that if it
; ever supports non-PLAS VIRTUAL's, that they will start at 160000 just
; like F4.)
; Thanks to Robert Walraven of Multiware, Inc. for this kludge!

3$:     MOV     @#$EXIT,R0              ; Get first word of exit routine
        CMP     R0,#152737              ; Is is F4 ?
        BNE     4$
        MOV     2(R5),R0                ; Yes, pick up VIRTUAL array offset
        BR      5$

4$:     CMP     R0,#13703               ; Is it F77 ?
        BNE     1$                      ; If not, I don't know what we are.
        MOV     @2(R5),R0               ; Yes, pick up VIRTUAL array offset

5$:     ADD     R1,R0                   ; Add offset of VIRTUAL base
        MOV     (SP)+,R1                ; Restore R1
        RETURN

        .END    ; IVADR
```



18 July 1986

Bill Leroy
The Software House, Inc.
2964 Peachtree Road NW # 320
PO Box 52661
Atlanta, GA 30355-0661

Dear Bill:

I assume from the last DECUS Newsletters you are the appropriate one to express my views on the combined newsletters. I am only interested in the Mini-tasker for RT-11 so the rest is junk I would like to throw away but the binding makes this difficult to do. I vote for saving some money by sending out only what people need and use OR at least sending the material unbound so it is easy to discard the unwanted portions.

Sincerely,

Francis J Wall

Francis J. Wall
290 Alamosa Rd. NW
Albuquerque, NM 87107
505-345-0768

Editor's reply - DEC is trying to save money -- by combining all the SIG Newsletters into one publication. Many of us have an interest in at least three SIG's, 1) operating system - such as RT-11 SIG, 2) language - such as COMMERCIAL LANGUAGES sig, and 3) environment - such as EDUSIG or SITE MANAGEMENT.

Other members of the RT-11 SIG are welcome to reply to either Francis Wall's or my opinions. What is your vote?

San Francisco Symposia Highlights

by

NED Rhodes

RT-11 Symposia Coordinator

The scheduling is now complete for the San Francisco Symposia that will be held in the Moscone Center on October 6-10, 1986. The RT-11 Special Interest Group (SIG) is sponsoring 22 sessions for this symposia. Because of the compatibility between TSX-Plus and RT-11, all the sessions are of potential interest to the TSX-Plus user, and you are invited to attend and become more involved in the RT-11 world. ("TSX-Plus" is a trademark of S&H Computer Systems, Inc. -- "RT-11" belongs to DEC).

The day begins with the RT-11 SIG Business Meeting and Roadmap sessions. The RT-11 Product Panel will discuss the newest release of RT-11 and its layered products. Some applications in BASIC-PLUS for RT-11 will be presented in a session, followed by a session for new users of FORTRAN-77 under RT-11.

Monday night contains four sessions that will be of interest to TSX-Plus users. The first session, presented by Jan Bramlett of S&H, will talk about TSX-Plus Windowing and Process Control. Next, Bob Walraven will discuss techniques for running really large applications under both RT-11 and TSX-Plus. Then Milton Campbell will discuss how to use Shared Memory Regions. Finally, TSX-Plus Magic will feature new and unusual things to do with your system. This is THE session to attend if you have questions!!

The sessions on Tuesday are varied. First, the VAX/RT Progress Report will be given. I might add that this session follows a stream of low-end VAX product announcements and is very appropriate. The next session will look at running MS-DOS and CP/M programs under TSX-Plus utilizing some of the commercially available Q-BUS co-processing products. The next RT-11 session will discuss the use of the KXJ co-processing board that some will find useful in data acquisition and control applications.

On Wednesday, the RT-11 Engineering Group will be presenting two technical sessions on the RT-11 Symbolic Debugger and Handler, and Utility Interfacing. The interfacing session will discuss the changes that were made to the RT-11 handlers, and why.

A magic session will be presented for those users of Saturn products under RT-11 on Wednesday, along with the RT-11 Third Party Software Applications Forum, where users of third party software products can discuss the merits or shortcomimgs of products that they have used.

A session on Interupt Response under RT-11 will be given on Thursday, along with a session that will document the VTCOM/TRANSF protocol interface. RT-11 users will finish the day (literally) with the RT-11

Users Speakout session. Like the TSX-Plus Magic session, this one will explore the internals of RT-11. The most fun is watching the RT-11 "experts" attempt to out do each other.

Friday is traditionally the wrap-up day, but rather than just having wrap-up sessions, a technical session was also scheduled. The RT-11 Engineering Group will present a session on IND examples. Following that will be the RT-11 Feedback session where Digital will defend all past design decisions concerning RT-11 and embrace all new ideas as their own. Planning for the next symposia will begin at the Wrap-up session that follows.

The real value of attending the sysposium is to meet other RT-11 and TSX-Plus users. If at all possible, make plans to attend the San Francisco Sysposium and interect with other users.

Spring 1986

Dallas Symposia Tape

On-Line Distribution

by

Tom Shinal

RT-11 Tape Copy Distribution

The distribution of the Spring 1986 RT-11 Symposia tape is now available from the NLO (National LUG (Local User Group) Organization). All LUG's who desire the tape are to contact the NLO to make arrangements to receive it.

There are many individuals who do not have access to a magnetic tape drive, and cannot take advantage of all the useful programs available. To accomodate these individuals, a new voluntary program has been initiated by the RT-11 SIG which will make the tape index and the save images available "on-line". The following sites will initiate service as indicated below. Depending upon the test results, more will follow. Any volunteers???

- -

Site:	GENERAL SCIENTIFIC CORPORATION
Service:	Binaries and index
Protocol:	KERMIT and VTCOM/TRANSF
Phone:	301-340-2776
Time zone:	Eastern
Hours:	6:00 p.m. to 8:00 a,.m.
Data rate:	1200 bps
Log-on:	DECUS
Pass-word:	GUEST

- -

Site:	SIDLINGER COMPUTER CORPORATION
Service:	Binaries and index
Protocol:	KERMIT
Phone:	512-344-4845
Time zone:	Central
Hours:	11:00 p.m. to 8:00 a,.m.
Data rate:	2400 bps (Watch for 1200 bps update)
Log-on:	DECUS
Pass-word:	GUEST

- -

Site: COMPUSERVE

On-Line Symposia Tape Distribution

Consult your Compuserve manual for information. Programs will be available within the PDP-1 SIG area.

- -

CAVEAT -- CAVEAT

These services are on a volunteer basis, and the system resources are also on a donated basis. Please don't abuse them, else we loose them...

Also, please note carefully the indicated times of availability. Calls outside of these times will be rudely disconnected and you will have willingly contributed to the pension and profit-sharing plan of your local telephone company.

The following patch to KED will "un-disable" the auto-repeat function of your VT-nnn terminal. The net result is a very tasty performance increase when repositioning the cursor with either the arrows or keypad functions.

This patch works by preventing the editor from sending the auto-repeat enable/disable escape sequences both before and after any cursor repositioning commands.

The patch must be installed with SIPP on KED V1.20. You can see what flavor you have by pulling up the help screen while in the editor. If you are not running this version of KED, then you must find the proper offset yourself.

Good luck.

David A. Lethe
Digital Computing Systems, Inc.
511 University Drive E #206
College Station, TX 77840

```
 .sipp sy:ked/a/d
 Base?
 Offset? 23432

   Base        Offset      Old   New?
    000000      023432   055433  \

   Base        Offset      Old   New?
    000000      023432     033   200
    000000      023433     133
    000000      023434     077
    000000      023435     070
    000000      023436     154
    000000      023437     200
    000000      023440     033   200
    000000      023441     133   ^Y

Checksum = 125231
```

Brian Nelson
Computer Services
University of Toledo
2801 West Bancroft
Toledo, Oh 43606

July 10, 1986

(419) 537-2841
BRIAN@UOFT02.BITNET

There appeared a letter in the July RT11 newsletter about Kermit and VTCOM loosing the high bit on all characters received during terminal mode. This is not the fault of Kermit-11 or VTCOM. The problem affects RT11 only and is caused by the XL/XC handler stripping this bit. It should not be a problem to edit XL.MAC and remove the second occurance of:

```
        BIC     #^C<177>,R5
```

Then, of course, one needs to assemble and relink the handler. Also, please be aware that Kermit-11 V3.51 has many new RT11 and TSX+ features, including a completely rewritten terminal connect module that is now entirely interupt driven (when using XL or XC). Updates may currently be obtained in the following manners:

Bitnet:

```
from VM/CMS:    CP SMSG RSCS MSG UOFT02 KERMSRV DIR
                CP SMSG RSCS MSG UOFT02 KERMSRV SEND K11*.*

from VMS Jnet:  $ SEN KERMSRV@UOFT02 SEND K11*.*
                $ SEN KERMSRV@UOFT02 VMSDUMP K11*.SAV
```

Dialup:

(419) 537-4411
Service class VX785A
User: KERMIT
Password: KERMIT

Source and hex files are in KER:, binaries are in KERBIN:

DECUS

SITE

SITE SIG STEERING COMMITTEE

CHAIRMAN
DMS SIG Liason
Larry W. Hicks
Relational Database services
P.O. Box 644
121 S. Main St.
Kernersville, NC. 27285-0644
(919) 996-4882

SYMPOSIA COORDINATOR
Sue Abercrombie
48 Malilly Rd.
Portland, ME 04103
(207) 772-2837

SESSION NOTE EDITOR
Large SIG liason
Gary Bremer
Emerson Electric Co.
8100 W. Florisant
St. Louis, MO. 63136
(314) 553-4448

NEWSLETTER EDITOR
Networks SIG Liason
OA SIG Liason
Gregory N. Brooks
Washington University
Behavior Research Labs
1420 Grattan St.
St. Louis, MO. 63104
(314) 241-7600 ext. 257

LIBRARY COORDINATOR
RSTS SIG Liason
Timothy Frazer
Specialized Bicycle Components
15130 Concord Circle #77
Morgan Hill, CA. 95037
(408) 779-6229

HARDWARE COORDINATOR
HMS SIG Liason
Emily Kitchen
A.H. Robbins Co.
1211 Sherwood Ave. RT-2
Richmond, VA. 23220
(804) 257-2925

COMMUNICATIONS COMMITTEE REP
AI SIG Liason
Terry C. Shannon
Digital Review
160 State St.
6th Floor
Boston, MA. 02109
(617) 367-7190

PRE-SYMPOSIA SEMINAR COORD
Phillip Ventura

STAFF MANAGEMENT
Adam Zavitski
Simmonds Precision ICD
3100 Highland Blvd.
Raliegh, NC. 27625
(919) 872-9500

MEMBERS-AT LARGE

Ann Goergen
Texas Instruments
13510 N. Central
M/S 437
Dallas, TX. 75266
(214) 995-4629

HMS SIG Liason
RT SIG Liason
David Hunt
Lawrence Livermore National Lab
MS L-230
P.O. Box 808
Livermore CA. 94550
(802) 656-3190

Gary Siftar
Digital Equipment Corporation
Tulsa, OK.

DEC COUNTERPARTS

Joe Allen
Digital Equipment Corp.

Lil Holloway
Digital Equipment Corp.

Susan Porada
Digital Equipment Corp.

Letter from the Editor

Welcome to the September issue of the Site Newsletter.

This issue of the newsletter marks one full year of publication for the BIG ONE. The combined SIG Newletters. Although the format has not pleased everyone, the BIG ONE appears to be a success with the majority of the membership. It has become a success from DECUS' stand point since subscriptions and submissions continue to roll in. There are efforts underway to bolster the format and the content for the second year of the BIG ONE, aimed at providing a quality publication. We hope you wil be pleased with our efforts. You comments are always welcome.

This issue contains a letter from the SIG's new chairman, Larry Hicks. Letter From the Chairman will be a regular feature in all future issues. We are also instituting a column, "Ask Siteman." Siteman will field questions from readers on any topic related to system and/or site management, training, documentation, computer rooms, etc. We are soliciting for questions now so we have some questions to field for the next issue.

The Site SIG just completed a "woods" meeting July 25th thru 27th. The woods meeting helped us get organized properly and formulate long range plans to keep the Site SIG healthy and progressive. More of the details of the woods meeting will be forthcoming in the next issue.

Articles are needed for up coming issues of the Site Newsletter. If you have an article, a Site Management hint, could transcribe a cassettes tape, (a fresh load of Dallas Site SIG Symposia session tapes have arrived) or even just an idea for an article, I would like to hear from you. My address and phone number are listed below.

I can accept the following formats: 1600 or 6250 BPI magtape (VAX backup, ANSI, RT, DOS, etc.) RX01, RX02, RL02, and RK05 disks in RT format, and TU-58's.

Gregory N. Brooks
Washington University
Department of Psychology
Behavior Research Labs.
1420 Grattan St.

St. Louis, MO. 63104
(314) 241-7600 ext. 257

LETTER FROM THE CHAIRMAN

Greetings from the New Chair:

This past spring brought several changes to the Site, Management and Training SIG steering committee and its members:

I was elected new SIG chair, Sue Abercrombie became Symposium Coordinator, Joe Allen became our DEC Field Service Counterpart and Adam Zavitski returned to the steering committee, following an 18 month hiatus. Tim Frazer and Adam changed employers and moved to the East Coast. Terry Shannon changed employers and moved to Boston. Debbie Boole got married and named Ann Goergen of her company as a replacement to the steering committee.

I know you will join me in extending our heartfelt thanks to Dave Hunt for his hard work and leadership these past two years, to Mike Weaver for his hard work and efforts in attracting and scheduling sessions the past two years, and to Rogers Bent for all his help with Field Service Presentations, new offerings and general Digital support. I hope to keep Dave involved in the SIG through our new Long Range Planning committee and to involve Mike in the development and presentation of a new Security PSS.

Having been a member of the Site Steering Committee since 1981, I have witnessed the emergence and adaption of SITE SIG from predominately a site management SIG to include system management and training. With your help and input, we will continue to grow and adapt to your needs.

As an indication of our determination to meet your needs, we held a Woods meeting in July to adapt our SIG and operating procedures to the "new" DECUS guidelines and to plan project and marketing activities for the next two to three years. Many of these plans and their results will be appearing in this and future newsletters.

Larry W. Hicks
RDBS, Inc.
121 S. Main St.
Kernersville, NC. 27285

SITE MANAGEMENT AND TRAINING SIG

Larry W. Hicks, SIG Chairman
Susan M. Abercrombie, Symposia Coordinator

The Site Management and Training SIG will have a full and varied schedule for its memebers attending the San Francisco Symposium. The Monday morning roadmap session will provide information on sessions sponsored by the SIG and on those from other SIGs which will be pertinent to managers and trainers.

The SIG has scheduled suite hours at the Hilton on Tuesday evening following the reception, and on Thursday evening from 7:00 pm to 11:00 pm. Attendees are invited to stop by at either time to relax, to become acquainted with people of similar interests, or to become involved in the SIG.

Monday's sessions during the day will relate largely to training issues. Monday evening will be "magic" and survival night. There will be munchies and a cash bar to help you relax and enjoy the fun. Digital will present sessions on the management of the DECUS exhibit hall, and on frequently overlooked hazards of computer room construction and operation. This will be followed by our "Site Magic," and by a highly interactive and thoroughly amusing session called "What to Do When the Computer Is Down."

Tuesday's sessions will cover capacity planning, user support, and documentation. The day will begin with a Digital session at which senior management from Field Service, Educational Services, and Software Services will provide a panel discussion with Q A on DEC's new Optimum Services, via a live, interactive satellite hookup.

Wednesday the SIG will focus on physical facilities. Sessions will be presented on fire protection, computer room design, network power and maintenance, and on disaster planning and recovery from the viewpoints of both the system manager and senior corporate management.

Thursday morning's session will be a tutorial on writing documentation. Thursday afternoon there will be sessions on system security in general and on VAX system management and system security.

On Friday, a noted DEC-watcher will try to answer the question "Where is DEC going?" Several sessions on migration issues will be presented in the morning, followed by sessions on distributed computing and on techniques for managing and distributing various software versions via network or backup sets, and a presentation on techniques to access and analyze VMS system performance monitor data. Our SITE business meeting and future directions session will also be held on Friday afternoon. You are encouraged to come to this meeting if would like to become active in the SIG or just to share common interests.

This should be an exciting Symposium, and we look forward to seeing you in October.

ROADMAP OF SESSIONS FOR THE SITE, MANAGEMENT TRAINING SIG

Sue Abercrombie, Site SIG Symposia Coordinator

The SIG has scheduled sessions relating to training, system management, security, documentation, physical facilities, migration, "magic," and miscellaneous sessions not otherwise classified. The following is a listing of these presentations in chronological order. DEC sessions are designated by a * in the last column.

Title	Code	Topic	Time	DEC
MONDAY				
Site, Management & Training Roadmap	S044	Misc	9:00- 9:30	
DIGITAL Software Licensing Overview	S008	Mgmt	9:30-10:30	*
DEC/IBM-PC Training	S005	Training	10:00-10:30	
New Hire Training - New Employees Need More Than Tech. Training	S047	Training	11:00-11:30	
Successful In-House Training	S006	Training	11:30-12:30	

How Not to Train New Users	S007	Training	2:00- 2:30	
Computer Illiterates at Work - Shielding Your System from Your Users & Vice Versa	S017	Training	2:30- 3:00	
Interpersonal Relations for Comp. Professionals	S033	Misc	3:00- 4:00	
DECUS Exhibit Hall System Man.	S032	Mgmt	7:30- 8:30	*
How To Destroy Your Hardware	S046	Facility	8:30- 9:30	*
Management & Training Magic	S042	Magic	9:30-10:15	
What to Do When the Computer Is Down	S045	Magic	10:15-11:00	

TUESDAY

DEC's New Optimum Services -- A Live Interactive Satellite Sess.	S022	Mgmt	9:00-10:00	*
Capacity Planning Early Warning System	S018	Mgmt	10:00-11:30	
Colorado Customer Support Center Overview	S025	Mgmt	12:00- 1:00	*
Establishing Overall User Supp.	S028	Training	1:00- 2:00	
Planning Training	S020	Training	2:00- 3:00	*
Designing Computer Forms: System Mgr Tutorial	S002	Docu	3:00- 4:00	
Minimize Costly On-Going User Support -- Provide Self-Instructional Doc.	S029	Docu	4:00- 5:00	

WEDNESDAY

Designing & Moving into a New Computer Room	S016	Facility	9:00-10:00	
Computer Room Fire Protection	S036	Facility	10:00-11:00	
Disaster Planning and Your Organization	S003	Mgmt	11:00-12:00	
Computer Room Design and Const.	S038	Facility	1:00- 1:45	
Computer Room Survival without Computer Room "Extras" - No Frills Computer Room	S031	Facility	1:45- 2:30	
Clean Power for Your Network	S019	Facility	2:30- 3:30	*
DEC's Multi-vendor Network Maint	S024	Facility	3:30- 4:30	*
Disaster Recovery Planning	S023	Mgmt	6:00- 7:00	*

THURSDAY

How to Write Documentation that Works -- a Technical Writing Tutorial	S037	Docu	9:00-10:30	
Evaluation of Computer Security	S013	Security	1:00- 2:00	
VAX Security and System Man.	S021	Security	2:00- 3:00	*
DIGITAL's Tempest Program	S012	Security	3:00- 4:00	*

FRIDAY

Where Is DEC Going?	S048	Misc	9:00-10:00	

PDP-11 to VAX Migration	S049	Migration	9:00-10:00	*
Developing a Hardware/Software Request for Proposal	S051	Mgmt	10:00-11:00	
Planning VAX Growth	S050	Migration	11:00-12:30	*
Automation of Distributed Compuer Services	S039	Misc	1:30- 2:30	
Site, Mgmt & Training SIG Future Directions	S043	Misc	2:30- 3:00	
Organize, Maintain & Distribute Products on Networked VAX Systems	S040	Mgmt	3:00- 4:00	
Using SAS to Process Raw System Performance Monitor (SPM) Data	S026	Mgmt	4:00- 5:00	

ASK SITEMAN

Ask Siteman is a new column feature to be fully implemented in the next issue, but we need questions, NOW!!!

If you have a question, please send it to Gregory N. Brooks at Washington University. The full address is listed in the Letter from the Editor and the Steering Committee list.

Questions should be on some topic related to system/site management, training, documentation, computer rooms, etc. Detailed questions that are operating system dependent should generally be submited to the SIG responsible for that particular operating system. However, many tuning fundamentals are generic. If in doubt, ask. If we can not field the question we will attempt to pass it on to the appropriate SIG.

UNISIG

DECUS

UNISIG Chair

James W. Livingston, Jr.
Measurex Corporation
1 Results Way
Cupertino, CA 95014
408-255-1500 x5556

ihnp4!decwrl!jwl

UNISIG Symposia Coordinator

Stephen M. Lazarus
Ford Aerospace, MS X-20
3939 Fabian Way
Palo Alto, CA 94303
415-852-4203

ihnp4!fortune!wdl1!sml

UNISIG Session Notes Editor

Kurt L. Reisler
Hadron Incorporated
9990 Lee Highway
Fairfax, VA 22030
703-359-6100

decvax!seismo!hadron!klr

UNISIG Newsletter Co-editor

William Toth
Harvard-Smithsonian
Center for Astrophysics
60 Garden Street, P-353
Cambridge, MA 02138
617-495-7181

harvard!hrvsmth!toth

UNISIG Newsletter Co-editor

James W. Livingston, Jr.

UNISIG Administrative Daemon

Dorothy Geiger
The Wollongong Group
49 Showers Drive, 451
Mountain View, CA 94040
415-948-1003

ihnp4!decwrl!dgeiger

UNISIG Tape Librarian

Carl Lowenstein
Marine Physical Laboratory
Scripps Institute of Oceanography, P-004
La Jolla, CA 92093
619-294-3678

(ihnp4|decvax|akgua|dcdwest|ucbvax)
!sdcsvax!mplvax!cdl

UNISIG Usenet Liason

Joe Kelsey
FlexComm Corporation
711 Powell Ave. SW
Renton, WA 98055

allegra!fluke!joe

UNISIG Standards Coordinator

Jeff Gilliam
National Semiconductor
2900 Semiconductor Drive, MS C2303
Santa Clara, CA 95051
408-721-3801

ihnp4!nsc!voder!jeff

UNISIG Minister Without Portfolio

Norman Wilson
Bell Laboratories, 2C-529
600 Mountain Avenue
Murray Hill, NJ 07974
201-582-2842

(decvax|ihnp4)!research!norman

UNISIG DEC Counterpart

Roseann Maclean
DEC, MKO2-1/H10
Continental Boulevard
Merrimack, NH 03054
603-884-5702

decvax!maclean

NEWSLETTER OF THE VAX SYSTEMS SIG

Pageswapper

Our Mascot

Forms at the End

General material for publication in the Pageswapper should be sent (US mail only -- no "express" services please) to:

Larry Kilgallen, PAGESWAPPER Editor
Box 81, MIT Station
Cambridge, MA 02139-0901
USA

Preference is given to material submitted as machine-readable text (best is Runoff source). Line length should not exceed 64 characters and the number oftext lines per page should not exceed 48 (these limits are particularly important for sample commands, etc. where simple text justification will not produce a meaningful result).

Please do not submit program source, as that is better distributed on the VAX SIG tape.

Please do not submit "slides" from DECUS Symposia presentations (or other meetings) as they are generally a very incomplete treatment for those readers of the Pageswapper who are not so fortunate as to be able to travel to Symposia. Please DO write articles based on such slides to get the content across to a wider audience than is able to attend.

For information about on-line submission to the Pageswapper dial:

(617) 262-6830

(in the United States) using a 1200 baud modem and log in with the username PAGESWAPPER.

Change of address, reports of non-receipt, and other circulation correspondence should be sent to:

DECUS U.S. Chapter
Attention: Publications Department
249 Northboro Road (BPO2)
Marlborough, MA 01752
USA

Only if discrepancies of the mailing system are reported can they be analyzed and corrected.

Editor's Workfile

by Larry Kilgallen, Pageswapper Editor

European newsletter delivery may have improved -

I/O submission 506 from Switzerland arrived in the US mail dated July 9 and was written on a form from the June Pageswapper. That indicates a delay of no more than 5-6 weeks after US publication. The paper was of a different size than we use in the US, presumably meaning it was a reprint by DECUS in Europe.

Deadline moved to September 25 -

The DECUS staff has announced that they must have copy for the November newsletters by September 25 (rather than the expected September 30). Pageswapper contributions should be submitted to me sufficiently early, as appropriate for their conformance to format, spelling and grammar standards.

Thanks to Gary Grebus -

for the fine work he has done over the years as SIR coordinator for the US Chapter VAX Systems SIG. Unfortunately for the SIG, Gary has taken a position with DEC in Massachusetts, and SIG leadership positions are not open to DEC employees. The result of Gary's move may be beneficial to us all, however, as the persistance he has put into the SIR process will presumably be applied to the quality of products DEC produces.

Letters

July 18, 1986

Digital Review
160 State Street
Sixth Floor
Boston, MA 02109

Dear Sirs:

In your Regular Rumor Roundup column for May 1986, you reported that Ford Motor Company is discontinuing the use of RA81 disk drives on VAX computer systems. To the best of my knowledge, this is far from the truth.

At the site where I am system manager, we have just purchased 9 RA81 disk drives to augment the 3 that have been providing faithful service for the past three years. A quick count indicates that there are over 70 RA Series drives in use in the Dearborn, MI area alone. There are no serious plans, that I know of, for any sites to "dump" RA's in favor of anything else. An informal poll of system managers indicates plans to acquire 20 to 30 additional RA Series drives in the next 18 months.

Although we experienced the "glue problem" with some of our drives, quick response by our Digital Field Service Engineer, Stuart Fuller, meant minimal downtime with many being replaced before any failures occured.

All in all, my VAX system and RA81 disk drives have proven to be the most reliable pieces of equipment I have worked on in my 10 years of computing experience.

Jack Patteeuw
Software Specialist
Electrical and Electronics Div.
Ford Motor Company

Pageswapper Editor's Comment

Jack also sent us a copy of a similar letter he sent to Hardcopy which had printed the same rumor. How come the *Pageswapper* never gets any exciting rumors?

INPUT/OUTPUT

A SIG Information Interchange

A form for INPUT/OUTPUT submissions is available at the back of the issue.

For information about on-line submission to the Pageswapper dial:

(617) 262-6830

(in the United States) using a 1200 baud modem and log in with the username PAGESWAPPER.

```
================================================================
Note 484.1              Ditto Foreign Tape Copy             1 of 1
"Dave Close"                               3 lines  12-JUL-1986 13:12
                     -< BIGTPC tape copy >-
----------------------------------------------------------------
```

Check program BIGTPC (big tape copy) on a recent SIG tape. Copies tapes as a single file onto disk, then back again. Can handle very large blocks.

Dave Close
Anadex/Printronix
1080 Avenida Acaso
Camarillo, CA 93010
805/987-9660

```
================================================================
Note 486.1      TeX DVI post-processor for terminals      1 of 1
"Bob Hassinger"                        8 lines  18-JUL-1986 11:46
           -< TeX DVI output previewers for VDUs >-
----------------------------------------------------------------
```

There is a previewer program available to do this called DVItoVDU. See pages 25 through 36 in the March 1986 (Vol 7, Num 1) issue of the TUGBOAT (TeX users group newletter). Kellerman and Smith (505-222-4234) included a copy on the VAX/VMS TeX tape we recently got from them.

In the same TUGBOAT issue see pages 55 and 56 for a rundown on output previewers in general.

Bob Hassinger
Liberty Mutual Research Center
71 Frankland Road
Hopkinton, MA 01748
617-435-9061

```
================================================================
Note 501.1    RK05 driver neede for VAX750 (VMS V4.3)     1 of 1
"Bob Hassinger"                        5 lines  18-JUL-1986 14:10
           -< Check VAX79A and VAX80A SIG tapes >-
----------------------------------------------------------------
```

Have you looked at the code on the VAX79A and VAX80A Symposium tapes? Look at [VAX79A.RK05DRV] and [VAX80A.DRIVERS]. Most likely would need updating to work well under VMS V4.

If you get it working we would be interested too...

Bob Hassinger
Liberty Mutual Research Center
71 Frankland Road
Hopkinton, MA 01748
617-435-9061

```
================================================================
Note 502.1     SET HOST/DTE on RACAL/VADIC 3451-PA        1 of 1
"Offline Submission"                  11 lines  22-JUL-1986 00:15
        -< I have a Racal/Vadic 4224 autodial routine >-
----------------------------------------------------------------
```

I have written an auto-dial routine for the Racal/Vadic 4224 modem. With minor changes I suspect it could be used with other Racal/Vadic modems.

Art Stine
Clarkson University
ERC
Potsdam, NY 13676
Telephone: 315-268-2292

July 16, 1986

```
================================================================
Note 506.0          Printer on DMF Printer Port          1 reply
"Offline Submission"                  12 lines  17-JUL-1986 00:51
----------------------------------------------------------------
```

Has anyone attached a parallel printer to the DMF printer port? If so, were there any specials in the setting up of the queue? Our Centronics with a Data Products interface only prints the flag page!

A. Busch
Atek NC Systems AG
Promenade 26
5200 Brugg
Switzerland
Phone: 056-419951

Date: July 9, 1986

```
================================================================
Note 5Ø6.1          Printer on DMF Printer Port           1 of 1
"Dave Close"                          5 lines  29-JUL-1986 14:24
               -< Another question on DMF printers >-
----------------------------------------------------------------
```

I have a similar question as I also have a printer with a DataProducts interface that I want to attach to the DMF printer port. I know its possible but I would rather build my own cable than pay $2ØØ to DEC. Does anyone know how the cable is wired?

Dave Close
Anadex/Printronix
1Ø8Ø Avenida Acaso
Camarillo, CA 9301Ø
8Ø5/987-966Ø

```
================================================================
Note 5Ø7.Ø VMS V4.4 erroneous %MOUNT-F-DEVCOUNT for tapes No rep
"Larry Kilgallen"                    15 lines  21-JUL-1986 Ø8:24
----------------------------------------------------------------
```

VMS V4.4 incorrectly gives the MOUNT$_DEVCOUNT error in response to a MOUNT command which specifies a different number of tapes than tape drives, e.g.:

```
 MOUNT MTA3: HJQ433,HJQ434,HJQ435
```

Looking at the microfiche, it seems to me that someone adding code for disk volume shadowing incorrectly gave the old disk device count code patch to tapes (perhaps they meant to give it to non-shadowed disks).

Reference:

 Image MOUNTSHR module VMOUNT routine MOUNT_VOLUME
 line 19Ø1.

Larry Kilgallen
Box 81, MIT Station
Cambridge, MA Ø2139-Ø9Ø1

```
================================================================
Note 5Ø8.Ø          LA1ØØ Line Drawing Problem        No replies
"Jack Patteeuw"                      15 lines  22-JUL-1986 11:29
----------------------------------------------------------------
```

I have a LA1ØØ-BA that I have recently upgrade to a LA1ØØ-CA by the addition of the "Multifont Adapter" (LA1ØX-FL). I then purchased a VT1ØØ Line Drawing "Dot Pattren Set" (DPS) (LA1ØX-JN) in the hopes of being able to print doucments created by WPS+'s Two Dimensional Editor (TDE).

Although the file created by WPS+ seems to contain all the proper escape and control characters (according to the LA1ØØ manuals) the terminal does not enter "graphics" mode.

Further investigation show that even when the terminal is in Loacl Mode and the commands to enter "graphics" mode (<ESC>)Ø^N) are typed at the keyboard, nothing happens.

DEC says "call Field Service" !!

Jack Patteeuw
Ford Motor Co.
Electrical and Electronics Division
3163Ø Wyoming
Livonia, MI   4815Ø
313-323-8643

```
================================================================
Note 5Ø9.Ø                  EDT <ESC>\                   1 reply
"Jack Patteeuw"                       3 lines  23-JUL-1986 1Ø:4Ø
----------------------------------------------------------------
```

EDT an TPU both send out a <ESC>\ before starting up. This escape sequence is not documented in VT1ØØ manual. What does it do and why ?

Jack Patteeuw
Ford Motor Co.
Electrical and Electronics Division
3163Ø Wyoming
Livonia, MI   4815Ø
313-323-8643

```
================================================================
Note 509.1                  EDT <ESC>\                     1 of 1
"Bob Hassinger"                          5 lines  24-JUL-1986 11:22
      -< <ESC>\ is 7 bit form of ST (String Terminator) >-
----------------------------------------------------------------
```

<ESC>\ is the 7 bit version of the eight bit ST control character. It is called "String Terminator" and is used to mark the end of a sequence that begins with the DCS (or <ESC>P) Device Control String control character. Most likely it is being sent to be sure the terminal is not in the middle of a device control string.

Bob Hassinger
Liberty Mutual Research Center
71 Frankland Road
Hopkinton, MA 01748
617-435-9061

*

VAX Tapecopy Tape from Dallas

The VAX Tapecopy tape from the Dallas Symposium is in distribution. As usual you should be able to get it from your LUG Librarian or order it from the DECUS Library. It consists of two 2400 foot, 1600 bpi tapes, [VAX000...] and [VAX86A...] on one reel and [VAX86B...] on the other. The index is back, however, it only goes through the Fall 1985 tape. The composite index is in a compressed format. Instructions for decompressing it are in [VAX000.INDEX].

Glenn Everhart, RCA, did most of the construction of the tape. I put in some late submissions and did some rearranging. However, I failed to update the file [VAX000]VAX86ABBRIEF.SUMMARY (dated 2 Jun 1986) prior to putting the tape in distribution. A corrected copy appears below.

Joe Bingham
VAX Systems SIG Librarian
ManTech Services Company
2320 Mill Road
Alexandria, VA 22314
(703) 838-5600

VAX SYSTEMS SIG Spring 1986 Tapes VAX86A and VAX86B Submissions Overview.

VAX86A Tape

[.ARIZONA]
Idle job killer for V4 incorporating "Break Buster".
Mail utilities to automagically forward mail to people's home nodes anywhere on a net, to purge deleted people from VMSMAIL.DAT as needed, keeping aliases in.
Joel Snyder, Univ. of Arizona.

[.BATTELLE]
DELBFID - Delete file by file ID.
Files - find files by owner or size.
FINDFILE - find file given disk address.
FLUSH - flush DCL recall buffer.
VBN - edit a file given filespec and virt block number.
Gary Grebus, Battelle.

[.BELONIS]

NETSUBMIT and NETPRINT - submit to arbitrary queues on a remote DECnet node.
ATNODE update for interactive remote commands.
SENDME pgm for notice of batch completion when remote. Also updates to HOST and MODEM7 communications packages. A
TAR reader and some menu driven BACKUP interface things.
James Belonis II, Univ. of Washington.

[.BJORND]

BATCH - issue one or more DCL commands in batch job. Biorhythm.
Fixes for HEXIFY and DEHEX for BIG files.
Directory editor (selective delete/examine).
EDTPLUS extensions to TPU EDT emulator (w/multiple windows, replace with confirm, etc.) and LSE enhanced interface too.
Runoff preprocessor for figure/tab numbers by section no.
SETUSER (a "become username" utility).
WHO - shows users, works with virt terminals.
Portia Bjorndahl, Hughes Aircraft.

[.BNELSON]

VMSTPC - rewrite of TPC in C for FAST tape-disk-tape copies on VAX in native mode. (limited testing as yet though).
Kermit-11 update.
Brian Nelson, University of Toledo. (A last-minute revision to VMSTPC.C was received. The source but not the .obj or .exe is on the tape. Compile VMSTPC.C if you have not extracted the

[.BZL]

Utilities to automatically run AUTOLOG at night but not by day.
DIAL support for set/host/dial. A
DRAWTREE that works on V4 disks.
Instruction speed monitors.
WPSINI.EDT - EDT keypad defs that look like WPS keys.
CORPHONE that replaces All-in-1 phone directory. (Does not need all in 1.).
LNØ3 support.
Bart Lederman.

[.CUTLER]

INDEX - moved to [.MIVAXLUG.EDS.CUTLER]

[.DFWLUG]

NATSIMI - TPU section file, VAXNET support for the Vadic VA24ØØPA modem.
PALMER - VAX based editor, games (Moria and Rouge which are in [.GAMES...), a terminal emulation.
TECHDEVEL - consolidate disk usage by user and report via DECgraph.

[.DMM]

Revised DM (directory management) package.

[.ELDR]

MLR structured programming macros with terminal I/O and quadword math examples.
Roderick A. Eldridge, Iowa State Univ.

[.ERI]

MACSnVAX communications program plus large volume of Macintosh public domain software.
DUPUSE - prevent users from logging into VMS twice.
RS1ARCSYS - RS/1 Archiving system.
Daniel Smith, Eye Research Institute.

[.EROS]

BOUNCER - Idle terminal killer.
CHECKER - Check passwords against a dictionary.
PASS - Prevent users from reusing passwords.
SU - Put password protection on the SETPRV privilege.
TREE - Show directory tree.

[.FORTH]

FORTH. Two versions of FIG FORTH are provided. One is RSX FORTH from an old RSX tape. The other is fully commented VAX FIG FORTH.
John Lundin, University of Richmond.

[.FTCOPY]

FTCOPY - foreign tape copy (to/from tape/disk). Also handles ASCII <-> EBCDIC translations and more.
Tomas Danforth, Woods Hole

[.HEX]

HEX - Utility designed to manipulate ASCII hex formatted files as output by cross-assemblers and linkers. Supports many popular formats.
David Moore, Telex Computer Products.

[.JAIN] Calculator - handy scientific calculator. Can be callable so it can become a "pop-up" inside your program.
SELECTONE is a curve fitting program using many possible techniques which also uses FMS.
Dilip Jain, Household Mfr. Inc.

[.JENKINS] CONTIGUIZE, BIGCONTIG - make contiguous files up for files.
TCOPY - fast tape to tape copy utility (double buffered).
Roger Jenkins, Wycliffe Bible Trans.

[.LJK] Renamed to [.PAGESWAPPER]

[.LLJ] Demo of feasibility of robust logout command file which cannot be circumvented by nonprivileged users.
Larry Johnson, Texas Instruments.

[.MIVAXLUG] COMPARDIR - compare 2 directories.
PERP - perpetually rescheduled batch jobs.
PORTWATCH - logs off inactive jobs. Kermit install via VMSINSTAL.
Local print on VT100/200 terminals.
FRAGment utility Extra features for TPU EDT interface.
DELTREE fixes.
DRAWTREE upgrades for VT200.
Unique filename generator.
Sendmessage utility.
James Fischer et. al., EDS.

[.NCAR] MAKE - a program maintenance utility like Unix MAKE or MMS.
MORE - file browse utility and ability to capture output of another command in a pipe.
Jonathan Corbet, NCAR.

[.NSWC] MAG - read and write foreign tapes.
SD - set/show directory, show directory tree.
LET - shorthand DEFINE or ASSIGN.
OWN - assume ownership of files in your directory.
Library for the above and for general use.
UIC.COM - translate between octal and alpha UIC's.

SMG.DAT - summary of SMG$ routines.

[.NU] QUEMON allows quasi-dynamic queue scheduling.
RPF ("Relative Pork Factor") gives a measure of how porked your system is. (This looks like some kind of global load measure.) Rand Hall, Northeastern Univ.

[.PAGESWAPPER] Pageswapper issues since last symposium.
Larry Kilgallen.

[.PIC] Digitized ReGIS picture of a lady for VT240, VT125 or the like.
Robert Morris, McDonnell Douglas.

[.PIPER] ARGUS - system idle daemon to log off idle terminals.
NOTICE - system notice facility.
SYSUSE - system monitoring tools.
WHELP - windowing HELP facility (DCL and callable). For VMS V4.2 up...
Derrell Piper, Univ. of North Carolina, Chapel Hill.

[.SAUTTER] CANCEL - cancel wildcard queue entries.
DO - multiple DCL commands on a line.
LOCK - lock terminal.
REMOVEDIR - delete tree.
RTL demo.
SD program.
Show quotas for UIC on devices.
Blank VTxxx.
COOKIE.
F$QUOTAS pseudo lexical function.
SEND to other users.
SETGRPUIC.
Background clock for VT100, VT200.
More.
from Bill Sautter.

[.SCANNER] SCANLIB - routines to connect to an NCS 7004 document scanner and retrieve the text scanned to a file.
Michael Sheehan, UNC Wilmington.

[.SDB] SDB - a small, simple DBMS in C. Works on VAX in native mode and on PDP11 in DECUS C.
D. Betz.

[.SEDT] Superfast and powerful editor. Multifile edits, overstrike / insert, rulers, WPS mode option, blinding speed, and keypad almost exactly like EDT.

[.SEWALL] RTL.PAS - produces an environment for some runtime library routines.
SMG.PAS gives environment for SMG$ routines.
Scott Sewall, College of St. Catherine.

[.SIMVAX] Command files. They set up terminal characts, set up letter quality printers, generate mailing lists, simple word processor system using EDT or TPU.
David Meile, Univ. of Minnesota

[.SLB] SLIB77 source librarian.
Alex Lunford, Western Area Pwr Admin.

[.TPUEDT] Enhanced TPU EDT emulator with many features from EVE and EVEPlus. Supports multiwindows, centering, rulers, rectangular cut and paste, easy learn keys, more.
Geoff Byant, Applicon.

[.UAB] LIST - screen lister in TPU.
BIGBROTHER - dynamic user display program.
GRADE - computes grades needed to make A,B,C, etc.
SMAUG -monitor CPU hogs & lower their prio.
GOLLUM - throttle any idle processes.
Mark Vevle, Univ. of Alabama.

[.VAXNET] VAXNET - Communication program. EXCELLENT and efficient virtual terminal handling, talks its own protocol or XMODEM, autodials, does raw send or capture.
MISH converts files to/from pure ASCII for send/receive. (keeps attributes too.) Supports callback, scripts, MUCH more.
VMODEM - separate (additional) MODEM7 communication utility.
Robin Miller, Northern Telecom. (Submitted by Beverly Kasper).

[.VMSTPC] BRUIN - command files to read BRU tapes onto ODS2 disks. (Note: it's preferable on VMS V4 just to use the VAX/RSX BRU utility...).
TPC - a native mode replacement for TPC, which makes a disk image from a tape and makes tapes from the disk image.
Dennis Costello, Cornell University.

[.VSH] VSH - A VMS Shell and some Un*x like utilities from the DECUS library written by Camillo Bongiovanni.

[.WATCHDOG] WATCHDOG - idle process killer. Works on VMS V4 systems.
George Walrod, Comprehensive Tech. Inter.

[.WENTZ] BECOME - VERY complete utility to "become" another user, and does not touch authorized priv mask. Changes most of the rest to the other user. Does NOT execute his login.com.
NEWPROMPT - change someone else's prompt.
MAILUAF - modify mail file utility.
FMSMENU - Interface between DCL and FMS to allow command procedures to be menu driven.
PASCAL environments for various things.
REQUEUE - requeue a batch job iff there's no copy in a queue.
Eric Wentz, GE.

[.YANKES] Parallel Library - implements many primitives for parallel programs including shared data & executable regions, handling subprocesses, and synchronizing critical regions.
Craig Yankes, DEC.

[.ZION] FNDFIL - find file by block .
USERS - continuous show users display.
Barry Zion, Fed. Res. Bank of NY.

Total size 168/2332/46707/46715

VAX86B Tape

[.GNUEMACS] VERY preliminary version of Gnu EMACS for VMS. Basically for hackers only at this point; usable but some bugs exist.
M. Sasaki, Harvard.

[.KMSKIT] SYSMGR - many system manager utilities. Lots of utilities for general uses. Includes autologoff, TEKGRAPH, CALC, TVG, power fail catcher, more.
DCL windows for LSE and TPU.
VPW V4.3 - a replacement for All in One that's Faster, Easier on your system, Simple to reconfigure, and does much more. Graphics extensions, windows with typeahead, NOTEBOOK, LN03 support for word processing, DTC, support for merging SIXEL graphs anywhere on a page, capture screens to sizel dump file, more.
Remote command execution.
RMS error codes help.
ZEUS for V4.X.
DG tape reader.
Curve fit.
By James Downward, KMS Fusion.

[.KERMIT] Kermits for various machines. Includes VMS Kermit 3.2.075, and also C Kermit, MSDOS Kermit, and CP/M Kermit.

[.RCAS86] AnalytiCalc spreadsheet update. Features some new functions (including a choose function), an install command file, input/output areas, ability to link arbitrary programs to it, more. .OLBs supplied if you lack compiler.
VTKERMIT - Kermit for IBM PC with VT102 emulator, Kermit, Xmodem, scripts, menus, autodial, all sources.
CTOOLS - a LARGE variety of C tools from DECUS C kit. Includes LZW compress and expand, many more. Most have .EXE in case you lack C compiler.
RSX FOCAL - interpreter.
FORCE - force command lines to another process. (Fixup for VMS V4 of Fall '83 tape version).
SIXEL - convert ReGIS files to sixel files for printers (free DECSlide).

NEWVPWMOD - modify the DTC in VPW submission to work with 4 digit years a la C. Garman's latest DTC so both can coexist. Also illustrates one customization of VPW.
Also a couple example goodies off Arpanet.
DATMG*.* - complete relational DBMS sources (fortran) with docs, build files, as DECUS C archives. Dearchive, build, use. Binaries supplied.
Glenn Everhart, RCA

page

VAX System SIG Committee List

As of July 10, 1986

Osman K. Ahmad - Large Systems Integration Working Group
Association of American Railroads
Technical Center, Research and Test Department
3140 South Federal Street
Chicago, IL 60616

Joe Angelico - Assistant Symposium Coordinator
US Coast Guard CCGD8(DT)
Hale Boggs Federal Building
500 Camp Street, New Orleans, LA 70130

Elizabeth Bailey - Volunteer Coordinator
222 CEB
Tennessee Valley Authority
Muscle Shoals, AL 35660

June Baker - Advisor
Computer Sciences Corporation
6565 Arlington Boulevard
Falls Church, VA 22046

Joe L. Bingham - Librarian
Mantech International
2320 Mill Road
Alexandria, VA 22314

Bob Boyd - Commercial Working Group
GE Microelectronics Center
MS 2P-04
Post Office Box 13409
Research Triangle Park, NC 27709

C. Douglas Brown - Security
Sandia Labs
Division 2644
P.O. Box 5800
Albuquerque, NM 87185

Jim Caddick - VAXcluster
General Datacom
Strait Turnpike
Middlebury, CT 06762-1299

Jack Cundiff - Symposium Coordinator
Horry-Georgetown
Post Office Box 1966
Conway, SC 29526

Tom Danforth - Handout Editor
Woods Hole Oceanographic Institute
Woods Hole, MA 02543

Jim Downward - Migration and Host Development, VAXintosh Working Group
KMS Fusion Incorporated
3941 Research Park Drive
Ann Arbor MI 48106

Jane Furze - Campground
3830 West Cochise
Phoenix, AZ 85064

Dennis Frayne - Real Time/Process Control Working Group
McDonnell Douglas
5301 Bolsa Avenue
Huntington Beach, CA 92646

Carl E. Friedberg - Internals Working Group
In House Systems
165 William Street
New York, NY 10038

Don Golden - Communications Committee Representative
c/o Shell Oil Company
Westhollow Research Center
Post Office Box 1380, Room D2132
Houston, TX 77001

B. Hancock - Network Working Group
Dimension Data Systems, Incorporated
2510 Limestone Lane
Garland, TX 75040
(214) 495-7353

Jeffrey S. Jalbert - Historian
J C C
Post Office Box 381
Granville, OH 43023
614-587-0157

Ray Kaplan - MicroVAX Working Group
Pivotal Incorporated
6892 East Dorado Court
Tucson, AZ 85715

Lawrence J. Kilgallen - Newsletter Editor
Box 81, MIT Station
Cambridge, MA 02139-0901

Margaret Knox - Chair
Computation Center
University of Texas
Austin, Texas 78712

Art McClinton - Advisor
MITRE
1820 Dolley Madison Boulevard
McLean, VA 22102

Ross W. Miller - Vice Chair and Working Group Coordinator
Online Data Processing, Inc.
N 637 Hamilton
Spokane, WA 99202

Mark D. Oakley - System Improvement Request
Battelle Columbus Labs
Room 11-6-008
505 King Avenue
Columbus, OH 43201-2669

Eugene Pal - Multiprocessor Working Group
US Army
CAORA (ATOR-CAT-C)
Fort Leavenworth, KA

Susan Rehse - System Management Working Group
Lockheed Missiles
3251 Hanover Street
Palo Alto, CA 94301-1187

Bob Robbins - Advisor
Array Computer Consultants
5364 Woodvale Drive
Sarasota, FL 33582

Larry Robertson - Real Time/Process Control Working Group
Bear Computer Systems Inc.
5651 Case Avenue
North Hollywood, CA

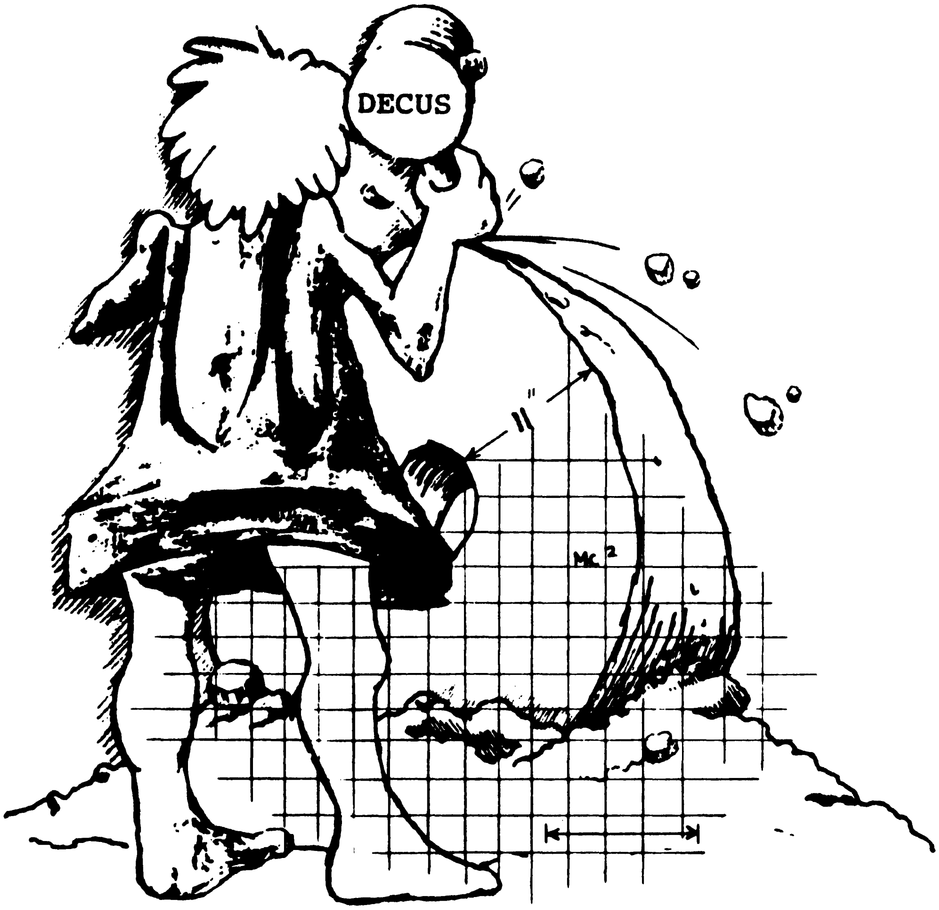
David Schmidt - LUG Coordinator, Hardware Working Group
Management Sciences Associates
5100 Centre Avenue
Pittsburgh, PA 15232

Al Siegel - Advisor
Battelle Memorial Institute
505 King Avenue
Columbus, OH 43201-2693

D. Slater - Artificial Intelligence Working Group
Institute for Defense Analysis
1801 North Beavregard Street
Alexandria, VA 22314

DECUS

LIBRARY



LIB

DECUS PROGRAM LIBRARY

NEW LIBRARY PROGRAMS AVAILABLE FOR THE PROFESSIONAL-300 SERIES OF COMPUTERS

DECUS NO: PRO-149 **Title:** CAMERA - Test of a Hamamatsu C1000 Camera on the PRO's RTIVersion: V1.0, December 1985

Submitted by: R. J. Wilden, Philips GmbH, Forschungslab. Aachen, Postfach 1980, West Germany 51 Aachen **Operating System:** P/OS V1.7 A **Source Language:** FORTRAN 77 **Memory Required:** .512 MB **Software Required:** PRO/Toolkit, PRTIL **Special Hardware Required:** PRTI, Hamamatsu C1000 Camera **Keywords:** Utilities - P/OS

Abstract: The task CAMTS1 provides a menu from which you can select tests to check all functions of the C1000 camera. The data acquisition task CAMTST allows you to input video data to a diskfile. The camera must be connected to the RTI with the IEEE-488 interface.

The tasks and all subroutines needed are written in FORTRAN 77. They work well with P/OS V1.7 A; later releases are supposed to function too. Sources are included.

All action needed to start the tasks can be found in the two indirect command-files CAMTS1 RUN. CMD and CAMTSTRUN. CMD. One point is important. If you use the PRTI-Software V2.0, you must change the RTI-Driver name from HG1.0 to HG1. A in the indirect command-files mentioned.

Documentation on magnetic media.

Media (Service Charge Code): One RX50 Diskette (JA), **Format:** FILES-11

DECUS NO: PRO-150 **Title:** APFELM - Graphical Presentation of Madelbrot_Set **Version:** V1.0, December 1985

Submitted by: R. J. Wilden, Philips GmbH, Forschungslab. Aachen, Postfach 1980, West-Germany 51 Aachen

Operating System: P/OS V2.0 A **Source Language:** FORTRAN 77 **Memory Required:** 5 MB **Software Required:** Native Toolkit **Hardware Required:** Color Monitor + Bit-Map, **Keywords:** Graphics

Abstract: APFELM displays in graphical form the so called Madelbrot_Set. With the help of a 'graphic-microscope', the complex-plane can be scanned for nice looking pictures.

When you use the graphic-microscope, the cursor position is the origin of a new picture. You can change the origin with the four Cursor-Keys and select a specific origin with the Select-Key. To continue with a new frame, you have to press the Resume-Key. When you intend to save a picture on disk, be sure to have enough space. The disk-space used for GIDIS-Metafiles is enormous.

Restrictions: Program neds P/OS V2.0 or later, must support GIDIS metafiles.

Media (Service Charge Code): One RX50 Diskette (JA) Format: Files -11

DECUS NO: PRO-152 **Title:** DIGITIZING - Graphical I/O Using a Tablet and HPGL-Plotter **Version:** V1.0, December 1985

Submitted by: R. J. Wilden, Philips GmbH Forschungslab. Aachen, Postfach 1980, West-Germany 51 Aachen **Operating System:** P/OS V2.0 A **Source Language:** FORTRAN 77 **Memory Required:** 5 MB **Software Required:** HP Industry Standard Plot Package (HPISPP), PRO/Tool Kit **Hardware Required:** DIGIT01-TABLET, HPGL-Plotter **Keywords:** Graphics, Plotting

Abstract: This package contains the following five programs:

TABTST	It tests Digital Equipment Corporation's DIGIT01-Tablet in Remote-Mode.
DIGIT1	It can digitize x/t-records of e.g. measured data.
DIAPL1	It plots the data produced by DIGIT1 on a HPGL-Plotter.
DIGIT3	It can digitize structural pictures like flowcharts. DIAPL3 It plots the data produced by DIGIT3 on a HPGL Plotter.

Digital Equipment Corporation's tablet DIGIT01 (Bit-Pad-One type) can easily be used to make X/T-Recorder measuring-data on paper available for computing. Furthermore, e.g. flowcharts outlined by hand can be digitized and so be prepared for use in papers. All digitized data can be plotted in a selectable form using a HPGL-Plotter (HP7475 A, LVP16...).

Two subroutines are used to service the tablet. TABGET is used to read the tablet's datastream. DIGITZ converts the digitized positional data to centimeter-pairs relative to the origin in the lower left corner.

The tablet can be connected to the Professional's Printer or Communication Port.

The digitized data is saved in disk-files. To distinguish the data produced by the tasks DIGITx, different data-structures (file-types) are used for saving.

The two plot-programs included enable the user to select freely, picture size and origin of the pictures. The plotter used to test the program was the HP7475 A.

Notes: HPISPP is licensed from HP.

Complete sources not included.

Media (Service Charge Code): One RX50 Diskette (JA) **Format:** FILES-11

DECUS NO: PRO-153 **Title:** LISSA2 - A Painting Game **Version:** V2.0, December 1985

Submitted by: R. J. Wilden, Philips GmbH Forschungslab. Aachen, Postfach 1980, West-Germany 51 Aachen **Operating System:** P/OS V1.7 A or later **Source Language:** FORTRAN 77 **Memory Required:** 5 MB **Software Required:** Native Tool Kit **Hardware Required:** Color Monitor plus Bit Map **Keywords:** Games

Abstract: LISSA2 computes and draws the points for a 'double' Lissajous-Figure. The layout of the resulting picture depends on eight input parameters which must be selected by the user. The name LISSA2 is derived from the well known Lissajous-Figures on which the task's main algorithm is based.

A typical set of start-parameters for the task LISSA2 is given as default values. To see the surprising amount of different looking pictures, you have to change the eight input parameters. The input parameters of good-looking pictures can be saved in a logfile. A sample logfile is included.

Media (Service Charge Code): One RX50 Disklette (JA) **Format:** Files-11

DECUS NO: PRO-154 **Title:** DELPHIN - Low Cost Modular I/O-System on the PRO's RTI **Version:** V1.0, December 1985

Submitted by: R. J. Wilden, Philips GmbH Forschungslab. Aachen, Postfach 1980, West-Germany 51 Aachen **Operating System:** P/OS V2.0 A **Source Language:** FORTRAN77, MACRO-11 **Memory Required:** .512 MB **Software Required:** PRO/Tool Kit, PRTIL **Hardware Required:** PRTI, DELPHIN-300 **Keywords:** Utilities - P/OS

Abstract: This package contains three tasks, controlling special low cost process-I/O hardware (DELPHIN-300 System) connected to the PRO's RTI (Real-Time Interface). DELTS1 is a menu driven test program for the DELPHIN-Hardware using the PRTIL-Calls. The connections between the PRO's PRTI and the DELPHIN I/O System is shown in the GIDIS-Metafile DELCON. GID. Use the PRO/Sight frame viewer or the print service with an LA50 to display the picture.

The other two tasks DELMCO and DELMEM are parts of a multitasking application. In this case, the PRTIL is not used, but the I/O is done using the device registers of the PRTI in the I/O-Page directly. Task DELMEM is a macro task which performs the actual I/O. The other task used (DELMCO) calls the function from DELMEM via subroutines. These subroutines send and receive messages to and from task DELMEM. To map the I/O-Page task DELMEM is a privileged task.

With the multitasking application, you can use the DELPHIN-300 system's data acquistion facility with the speed of the ADC used (ca. 12 KHz). Higher speeds are possible (it depends on the ADC's conversion time). To solve the problems with the PDP's lack of virtual address space, MMU-directives are used. The measured data can be saved on an RX50 diskette wih logical I/O. Read the comments in the source files for information about the functions implemented.

Media (Service Charge Code): One RX50 Diskette (JA) **Format:** FILES-11

DECUS NO: PRO-155 **Title:** RT Programs for PRO **Version:** March 1986

Submitted by: C. E. Chew **Operating System:** RT-11 V5.01, 5.02 **Source Language:** MACRO-11, NBS PASCAL **Software Required:** NBS PASCAL required to compile some programs if customization is needed. Keywords: Device Handlers, Spell, Text Formatting, Utilities - RT-11

Abstract: This is a potpourri of programs written for RT-11 V5.1 or later (except where noted) on a PRO. The following have been provided:

PL	A pipeline handler which functions in much the same way as MQ: except that not special .LOOKUP requests are needed.
CI	A console interface which allows one job to 'type' input to another.
MENU	A suite of rather crude menu control subroutines.
TYPO	A typographical error checker written in NBS PASCAL.
MORE	A file perusal utility written in NBS PASCAL.
OTHER	A program which determines which drive (0 or 1) RT is booted from and assigns logical names to it (SYS) and the other drive (DK and DSK).
WP	A program utilizing all the above to allow the creation of a cheap but effective text formatting system using KEX and RUNOFF (you have to provide your own KEX and RUNOFF).
DZCOPY	Program to make a sector by sector image of a foreign disk by using th DZ controller hardware. Has been used to read IBM format 5.25 inch disks.
XHANDL	An alternative overlay handler which can force large root segments and large overlay tables into extended memory.
PRTSCR	A screen dump utility. It can be customized for non-DEC printers, requires less low memory than the SPOOL utility and can dump in text or graphics mode but requires V5.02.

Note that some programs may require a little experience with RT and MACRO to customize, but should be fairly easy to put together.

Media (Service Charge Code): One RX50 Diskette (JA) **Format:** RT-11

NEW LIBRARY PROGRAMS AVAILABLE FOR THE RAINBOW SERIES OF COMPUTERS

DECUS NO: RB-106 **Title:** HACK **Version:** V1.0.1, February 1986

Author: D. Kneller **Operating System:** MS/DOS **Source Language:** Various **Memory Required:** 256 KB **Keywords:** Games

Abstract: Are you being digested by mysterious monsters, or perhaps being attacked by your own ghost? Gnomes, giant bats, orcs and zombies are only a small number of the hazards that await you as you hack your way down through twenty levels of dungeon, with only magic spells, potions, wands and rings, and perhaps a lot of brute strength as well as a dry sense of humor to aid you in your quest for thc Amulet of Yendor.

For addicts of Rogue, available under UNIX, HACK is an updated and extended version for personal computers. The memory required to run it is 256 KB and the main rule is simple: Hack unto other monsters before they hack (and digest) unto you!

Sources not included. Documentation on magnetic media.

Media (Service Charge Code): One RX50 Diskette (JA) **Format:** MS/DOS

DECUS NO: RB-110 **Title:** EMPIRE: The War Game of the Century **Version:** V5.0, April 1986

Operating System: MS/DOS V2.11 **Memory Required:** 256KB **Keywords:** Games

Abstract: EMPIRE is a war game where you battle the computer to take over the world. The world, constantly displayed on the screen, contains two power-hungry monarchs ... you, and the computer. Needless to say, there is only room for one of you, and if you don't get him, he will get you.

At your command are armies, troop transports, fighters, submarines, etc. and your wits. Each game starts with a unique map of the world and is likely to take hours of excitement to finish (games can be saved if you become exhausted/frustrated).

Sources not included.

Media (Service Charge Code): One RX50 Diskette (JA)

Format: MS/DOS

NEW LIBRARY PROGRAMS AVAILABLE FOR THE VAX/VMS FAMILY OF COMPUTERS

DECUS NO: VAX-169 **Title:** BIBENTRY **Version:** February 1986

Submitted by: Jack Pledger, OISE, Toronto, Ontario, Canada M5S IV6 **Operating System:** VAX/VMS **Source Language:** PASCAL **Hardware Required:** VTxx Terminal **Keywords:** Utilities - VMS

Abstract: BIBENTRY is a full screen data entry system designed to enter bibliographic data to be processed by Unilogic's Scribe* text formatting package.

Bibliography entries are entered into an indexed sequential file that can be converted to a form readable by Scribe when needed. Fields for a given entry type, such as 'book', 'article', etc. are displayed on the screen as a guide to the user. By using the VTxx arrow keys to position the cursor, the user selects the appropriate field. If needed, new entry types can be defined or the existing ones modified.

BIBENTRY indicates what information should be entered for a particular bibliography entry type, what entry types are available, what information is optional and what information is required. It also simplifies bibliographic data entry by automating the insertion of Scribe delimiters and field names. Other advantages include random access to the bibliography database, some limited string search capabilities and "user friendly" menus and prompts.

*Scribe is a trademark of Unilogic Ltd.

Media (Service Charge Code): 600' Magnetic Tape (MA) **Format:** VMS/BACKUP

NEW LIBRARY PROGRAMS AVAILABLE FOR THE PDP-11 COMPUTER FAMILY

DECUS NO: 11-836, **Title:** ReGIS to HP-GL Conversion Program **Version:** V1.J, December 1985

Submitted by: Dr. N. S. Hoult, Racal Research Ltd., Reading, Berkshire, England **Operating System:** RSX-11M-PLUS **Source Language:** MACRO-11, FORTRAN 77 **Memory Required:** 36KW **Software Required:** FORTRAN-77 run time system. **Hardware Required:** IEC-11 (though it could easily be modified for other interfaces). **Keywords:** Graphics, Hewlett Packard, ReGIS

Abstract: This program converts a file of ReGIS graphics commands (as used by the VT125 and VT240 terminals) into Hewlett-Packard Graphics Language (HP-GL) (as used on the 7470A plotter), and sends them to the plotter via an HPIB interface. All ReGIS commands are parsed, but only a subset (sufficient for line graphs with labelling, and including macrographs) is sent to the plotter. The resulting graph is designed to fit on A4 paper, and is approximately the same size as that produced using the 'expanded print' option of the VT240. The program is designed to facilitate the addition of extra ReGIS commands or the use of an alternative interface (e.g. RS232).

Restrictions: Not all ReGIS commands are interpreted, though the parsing deals with them. Commands may not be split over record boundaries.

Media (Service Charge Code): One RX01 Diskette (KA) **Format:** FILES-11

DECUS NO: 11-840, **Title:** Sample MicroPower/Pascal Programs **Version:** March 1986

Submitted by: John T. Davies III, Thermo Electron Instruments, Pittsburgh, PA **Operating System:** MicroPower/Pascal **Source Language:** IND, PASCAL **Hardware Required:** Standard MicroPower/Pascal development system **Keywords:** PASCAL

Abstract: The files included in this submission are small, general purpose MicroPower/Pascal routines. They are best used as examples of useful things that can be included in a MicroPower application. This is basically my version of the "MicroPower/Pascal Newsletter". These programs can be best used by new MPP users, but may have some interest to old hands as well.

Media (Service Charge Code): One RX01 Diskette (KA) **Format:** RT-11, 600' Magnetic Tape (MA) **Format:** RT-11

NEW LIBRARY PROGRAMS AVAILABLE FOR THE VAX/VMS FAMILY OF COMPUTERS

DECUS NO: V-SP-52, **Title:** Symposium Tape from VAX SIG, Spring 1986, Dallas **Version:** V1, July 1986

Submitted by: J. L. Bingham, Mantech Services Company, Alexandria, VA **Operating System:** VAX/VMS V4.X **Source Language:** PASCAL, MACRO-32, VAX-11 FORTRAN, DCL, VAX-11 COBOL, C, BLISS-32, VAX-11 BASIC **Memory Required:** No Specific Requirements **Keywords:** Symposia Tapes - VMS, System Management - VMS, Utilities - VMS

Abstract: This tape consists of the VAX submissions to the Tapecopy Project at the Spring 1986 DECUS Symposium in Dallas, TX. A brief description of the submissions follows. For more complete descriptions, see the AAAREADME.TXT files in each submission or the concatenated version of the AAAREADME'S IN VAX000 ; for documentation check for pointers in the AAAREADME.TXT's for files containing the string "READ" in their name and for files with .DOC, .TXT, .MEM, .RNO and .1ST extensions in the submissions.

SUBMISSION OVERVIEWS FOR VAX86A...

[.ARIZONA]	Idle job killer. Network mail utility.
[.BATTELLE]	Delete files by file ID. Find files by owner or size. Find files given disk address. Flush DCL recall buffer. Edit a file given filespec and virtual block number.
[.BELONIS]	Submit to queues on a remote DECnet node. Update for interactive remote commands. Notice over network of batch job completion. Updates to HOST and MODEM7. TAR reader.
[.BJORND]	Issue DCL commands in batch job. Biorhythm. Fixes for HEXIFY and DEHEX for BIG files. Directory editor. EDTPLUS extensions to TPU EDT. LSE enhanced interface. Runoff preprocessor. WHO. SETUSER.
[.BNELSON]	TPC in C for FAST tape-disk-tape copies. KERMIT-11 update.
[.BZL]	Run a program at certain hours of the day. Support for SET HOST/DIAL. DRAWTREE. Instruction speed monitors. EDT keypad defs that look like WPS. Phone directory.
[.ELDR]	Structured programming macros with terminal I/O and quadword math examples.
[.ERI]	MACSnVAX communications program plus large volume of Macintosh public domain software. Prevent users from logging into VMS twice. RS/1 Archiving system.
[.FORTH]	Two versions of FIG FORTH.
[.FTCOPY]	Foreign tape copy, including ASCII {-} EBCDIC translations.
[.HEX]	Utility designed to manipulate ASCII hex formatted files as output by cross-assemblers and linkers.
[.JAIN]	Scientific calculator. Curve fitting program.
[.JENKINS]	Make files contiguous. Fast tape copy utility.
[.LLJ]	Demo of feasibility of robust logout command file.
[.MIVAXLUG]	Compare directories. Perpetually reschedule batch jobs. Log off inactive jobs. Install KERMIT via VMSINSTAL. Local print on VT100/200 terminals. Disk fragmentation utility. Extra features for TPU EDT interface. DELTREE fixes. DRAWTREE upgrades for VT200. Unique filename generator. Sendmessage utility.
[.NCAR]	Program maintenance utility like UNIX MAKE or MMS. File browsing utility similar to UNIX MORE.
[.NSWC]	Read and write foreign tapes. Set/show directory, show directory tree. Shorthand DEFINE or ASSIGN. Assume ownership of files in your directory. Library for the above and for general use. Translate between octal and alpha UIC's. Summary of SMG$ routines.
[.NU]	Allow quasi-dynamic queue scheduling. Measure of how loaded your system is.
[.PAGESWAPPER]	Pageswapper issues since last symposium.
[.PIC]	Digitized ReGIS picture of a lady for VT240, VT125 or the like.
[.PIPER]	Log off idle terminals. System notice facility. System monitoring tools. Windowing HELP facility.
[.SAUTTER]	Cancel wildcard queue entries. Handle multiple DCL commands on a line. Lock terminal. Delete entire directory tree. RTL demo. SD program. Show quotas for UIC on devices. Blank VTxxx. COOKIE. F$QUOTAS pseudo lexical function. SEND to other users. SETGRPUIC. Background clock for VT100, VT200.
[.SCANNER]	Connect to an NCS 7004 document scanner and retrieve the text.
[.SDB]	A small, simple DBMS in C.
[.SEDT]	Fast and powerful editor with EDT-like or WPS-like keypad.
[.SEWALL]	Environments for some runtime library and SMG$ routines.
[.SIMVAX]	Command procedures to set up terminal characteristics, set up letter quality printers, generate mailing lists. Simple word processor system using EDT or TPU.
[.SLB]	SLIB77 source librarian.
[.TPUEDT]	Enhanced TPU EDT emulator.
[.UAB]	Screen lister in TPU. Dynamic user display program. Compute student grades. Monitor CPU hogs and lower their priority. Kill idle processes.
[.VAXNET]	Multi-featured communications program.
[.VMSTPC]	Read BRU tapes to ODS2 disks. Native mode replacement for TPC.
[.VSH]	VMS Shell and some UNIX like utilities from the DECUS library.
[.WATCHDOG]	Idle process killer.
[.WENTZ]	Utility to "become" another user. Change someone else's prompt. Modify mail file utility. Interface between DCL and FMS to allow command procedures to be menu driven. PASCAL environments for various things. Requeue a batch job if there's no copy in a queue.
[.YANKES]	Parallel Library - implements many primitives for parallel programs.
[.ZION]	Find a file by block number. Continuous show users display.

SUBISSION OVERVIEWS FOR VAX86B...

[GNUEMACS] VERY preliminary version of Gnu EMACS for VMS. Basically for hackers only at this point; usable but some bugs exist.

[.KMSKIT] Many system manager utilities. Lots of utilities for general uses. Includes auto logoff, TEKGRAPH, CALC, TVG, power fail catcher, more. DCL windows for LSE and TPU. VPW V4.3. Graphics extensions, windows with typeahead, NOTEBOOK, LN03 support for word processing, DTC, support for merging SIXEL graphs anywhere on a page, capture screens to sizel dump file, more. Remote command execution. RMS error codes help. ZEUS for V4.X. DG tape reader. Curve fit.

[.KERMIT] KERMITS for various machines. Includes VMS KERMIT 3.2.075, and also C KERMIT, MS/DOS KERMIT, and CP/M KERMIT.

[.RCAS86] AnalytiCalc spreadsheet update. KERMIT for IBM PC with VT102 emulator, KERMIT, XMODEM, scripts, menus, autodial. Large variety of C tools from DECUS C kit. Includes LZW compress and expand, many more. Force command lines to another process. Convert ReGIS files to sixel files for printers.

No guarantees are made as to the completeness, usability, or quality of the programs on the tape and the material has not been checked or reviewed.

Notes: Some submissions incompatible with earlier versions of VMS. Many file names incompatible.

Restrictions: None known, except as noted in individual submissions.

2Complete sources may or may not be included.

Media (Service Charge Code): 2400' Magnetic Tapes (PB) **Format:** VAX/VMS

DECUS NO: V-SP-53, **Title:** KERMIT Distribution **Version:** V1, April 1986

Submitted by: Glenn Everhart, Ph.D. **Operating System:** PRO/RT-11 & Many Others, P/OS, MS/DOS, VAX/VMS, CP/M, OS/8, TOPS-20, TOPS-10, RT-11, RSX-11S, RSX-11M-PLUS, RSX-11M, RSTS/E, IAS **Source Language:** PASCAL & Many Others, PAL-8, MACRO-32, MACRO-11, MACRO-10, FORTRAN 77, FORTRAN IV, C, BLISS-36, BLISS-32, BLISS-16, BASIC-PLUS, MINC BASIC, BASIC-20, ALGOL **Keywords:** KERMIT

Abstract: This distribution contains KERMIT programs for most machines for which a KERMIT distribution has been released as of about April 2, 1986. All Digital Equipment Corporation systems and OSs are represented (except possibly PDP9/PDP15) and MANY others. The KERMIT-10 and KERMIT-20 for DECSystem-10 and DECSystem-20 are not the most recent, but are up to date as of 11/85. This is essentially an up to date version.

Also included is the VTKERMIT submitted to the RSX Fall '85 SIG tape which does scripts, Xmodem and menus on 8088/PCDOS machines. Essentially, all versions come with sources except "field test" releases such as Amiga KERMIT.

The intent here is to provide a fairly full update to the KERMITS for the DECUS community since they are too large to fit all of them on symposium tapes anymore.

Restrictions: None (See relevant .BWR files)

Complete sources may or may not be included.

Media (Service Charge Code): 2400' Magnetic Tape (PC) **Format:** VMS/BACKUP

DECUS NO: VAX-170, **Title:** CED **Version:** December 1985

Author: Martin Fricker, Juergen Rued, Heinz Schellhammer, Ulrich Stauss, Fachhochschule Furtwangen

Submitted by: Kurt H. Schmidt, Fachhochschule Furtwangen, D-7743 Furtwangen, West Germany **Operating System:** VAX/VMS V4.1 **Source Language:** MACRO-32 **Memory Required:** 957 blocks **Keywords:** Editors

Abstract: CED is a screen oriented text editor with a great user compatibility to the Digital Equipment Corporation EDT. In addition to that, CED gives you the power of a programmable scientific calculator. You have the possibility to insert results of the calculator into text or to get parameters for the calculator out of text. The edit part of CED offers you nearly every function of the Digital Equipment Corporation EDT. It's very easy to become familiar with CED for anyone who is familiar with EDT.

Media (Service Charge Code): 600' Magnetic Tape (MA) **Format:** VMS/BACKUP

DECUS NO: VAX-171, **Title:** LaTeX Templates & Help Files for LSE **Version:** V1.0, March 1986

Submitted by: Lear Siegler **Operating System:** VAX/VMS V4.2 **Source Language:** LSE **Software Required:** LSE, LaTeX. This package requires LSEDIT to be installed. **Keywords:** Text Formatting

Abstract: LaTeX is a powerful, easy to use, public domain text formatting package based on TeX. This submission includes a VAX Language Sensitive Editor (LSEDIT) language definition of LaTeX. Using LSEDIT and the LaTeX language definition, a user, regardless of his/her level of experience, can quickly and easily learn to format complex documents using LaTeX. Use of LSEDIT reduces the amount of typing necessary by automatically supplying the user with a set of templates that define the basic structure of a given LaTeX style. These templates can be selected and filled in or deleted as appropriate. The novice user will use the templates extensively, while the more experienced user will use the templates as an aid in remembering infrequently used commands or formats.

The default LaTeX styles supported by LSEDIT language definition are: article, report, letter and slides (SLiTeX). This submission also includes three new styles for LaTeX: memo, MIL-STD-490 documents, and book form documents. These new styles are also supported by the LSEDIT language definition for LaTeX.

VMS format HELP library entries are included for most of the features within version 2.08 of LaTeX and SLiTeX. The LaTeX source for the "VAX Language Sensitive Editor (LSEDIT) Quick Reference Guide for use with LaTeX Environment" is included on the magnetic media.

Assoc. Documentation: Should have access to LSE and LaTeX manuals.

Media (Service Charge Code): 600' Magnetic Tape (MA) **Format:** VMS/BACKUP

DECUS NO: VAX-175, **Title:** International RUNITOFF **Version:** V1.3, February 1986

Submitted by: Lorrain Giddings, Ph.D., INIREB (Nat. Inst. Biol. Res.), Xalapa, Veracruz, 91000 Mexico **Operating System:** VAX/VMS V3.7, 4.1 **Source Language:** VAX-11 FORTRAN, DCL **Memory Required:** Variable **Software Required:** None (unsupported program OLY is included) **Keywords:** RUNOFF, Text Formatting, Word Processing

Abstract: International RUNITOFF is a program for text processing in several languages. It is based (as a preprocessor) on RUNOFF (DSR: Digital Standard Runoff for VAX computers) and retains all of the flexibility of that program. It was designed for use by secretaries, students, and other persons without specialized knowledge of computers or of the RUNOFF program, and it is quite easy to use.

The document presents instructions for beginners, and includes more detailed information for users with advanced knowledge of DSR RUNOFF. In addition to the RUNITOFF commands in Spanish and English, the program permits the direct incorporation of RUNOFF commands within the text. The RUNITOFF commands are formed of ordinary words in Spanish and English without a rigid format; Spanish and English forms can coexist in a given text.

Notes: Separate versions furnished for VMS 3.7 and 4.1

Media (Service Charge Code): 2400' Magnetic Tape (PA) **Format:** VMS/BACKUP

DECUS NO: VAX-176, **Title:** TXYZ: A Program for Semiconductor IC Thermal Analysis **Version:** V1.1, January 1985

Author: John Albers

Submitted by: Frank F. Oettinger, National Bureau of Standards, Gaithersburg, MD **Operating System:** VAX/VMS V4.2 **Source Language:** VAX-11 FORTRAN, FORTRAN 77 **Memory Required:** 2,022,400 Bytes of Virtual Address Space **Keywords:** FORTRAN

Abstract: This program uses the closed form, analytic solution to the conduction heat flow equation developed by Achilles G. Kokkas (RCA Labs.) and implemented by John Albers (NBS) to calculate the steady-state temperature at any point or set of points in three-layer rectangular structures due to planar heat sources. In its present form, TXYZ will allow up to 20 heat sources. The present limitations on the maximum number of Fourier-series terms are 500 by 500. Other limitations and theoretical discussions pertaining to the program can be found in the paper by John Albers entitled "TXYZ: A Program for Semiconductor IC Thermal Analysis", NBS spec. Publ. 400-76, April 1984. Program modifications and a user-friendly preprocessor were conceptualized and written by Stephen Ross (NBS) and Frank F. Oettinger (NBS). For additonal information, contact Frank F. Oettinger, (301) 921-3541 or Colleen H. Ellenwood, (301) 921-3801, at the National Bureau of Standards, Div. 727, Gaithersburg, MD 20899.

Release Notes are distributed with this order.

Media (Service Charge Code): 600' Magnetic Tape (MA) **Format:** VAX/ANSI

DECUS NO: VAX-177, **Title:** JP5/JP6 IMAGE MONITOR **Version:** V2.0, January 1986

Submitted by: Felix Fibich, Osterr. Bundesinst F. Gesundheitswesen, A-1010 Wien/Austria **Operating System:** VAX/VMS V4.1 **Source Language:** MACRO-32 **Memory Required:** Virtual **Keywords:** System Management - VMS

Abstract: The JP5 program displays username, image file name, and terminal ID (provided the process is interactive) of all currently active processes in a MONITOR like fashion. The JP6 program adds the image name (as inserted by linker operation) into the display. This is to prevent images to remain undetected by simply renaming the image file. To display all processes, JP5 requires WORLD, and JP6 WORLD and READALL privileges. Both programs use the $GETJPIW system service to obtain the JP5-information. JP6 additionally maps to the image header to get the image name. Both versions are easily extendable to yield other types of information. To accomodate long file names, both versions sense the terminal width in order to grant more space in 132 columns mode than in 80 columns mode.

Because of the different image header layout in RSX-Task files, only the image file name, but not the image name of those files, can be displayed.

Notes: $GETSPIW does not exist below VMS V4.0

Media (Service Charge Code): 600' Magnetic Tape (MA) **Format:** VMS/BACKUP

REVISIONS TO LIBRARY PROGRAMS

DECUS NO: 11-490, **Title:** TSXLIB: A FORTRAN Callable Library Implementation of EMTs for TSX-PLUS V6.0 **Version:** 86b27a, March 1986

Submitted by: N. A. Bourgeois, Jr., NAB Software Services, Inc., Albuquerque, NM **Operating System:** TSX-PLUS **Source Language:** MACRO-11 **Software Required:** FORTRAN compiler **Hardware Required:** MMU to support TSX-PLUS **Keywords:** FORTRAN, Libraries - RT-11, TSX

Abstract: TSXLIB is a library of FORTRAN callable routines that implement the TSX-PLUS system services which are unique to TSX-PLUS. The library has been updated to include all TSX-PLUS unique services through TSX-PLUS V6.0.

Like RT-11, TSX-PLUS offers the MACRO-11 programmer a number of system services. These services are implemented via both the RT-11 programmed requests (for those services common to both RT-11 and TSX-PLUS) and raw EMT instructions (for those unique to TSX-PLUS). RT-11 makes its system services available to the FORTRAN programmer through the system subroutine library, SYSLIB. TSX-PLUS also honors the bulk of the service requests in the SYSLIB routines. TSXLIB, however, makes the TSX-PLUS unique EMTs available to the FORTRAN programmer.

These TSX-PLUS library routines provide facilities to support communication lines, detached jobs, device allocating and deallocating, file structured device mounting and dismounting, communication between running programs, job privileges control, job status monitoring, program performance analysis, real time program execution, shared run time systems, shared files, special files information, spooler control, communication between running programs and a terminal, program control of the terminal, ODT activation mode, user name control, windowing, and several miscellaneous EMTs.

The TSXLIB distribution kit includes the MACRO-11 source modules for all the routines, a user's manual in machine readable form, an indirect command file to build the library, and the implemented library.

Media (Service Charge Code): One RX02 Diskette (LA) **Format:** RT-11, 600' Magnetic Tape (MA) **Format:** RT-11

DECUS PROGRAM LIBRARY CHANGES

DECUS Program Library CHANGES:

- For DECUS Order Number 10-LIB-12, The DECsystem-10 Library Tape 12, Version: 1986/1987, DECUS Order Number 10-364 will be included.

HOW TO SUBMIT TO A SPECIFIC SECTION OF THE NEWSLETTER

The following is a listing of the Newsletter Editors with their addresses and phone numbers. All submissions to the newsletter should be submitted directly to the appropriate Editor.

ARTIFICIAL INTELLIGENCE
Terry Shannon
160 State Street
Boston, MA 02109
(617)367-7190

BUSINESS APPLICATION
Thomas Byrne
L. Karp & Sons
1301 Estes
Elk Grove, IL 60007
(312)593-5705

COMMERCIAL LANGUAGES
Ted Bear
RAMTEK
2211 Lawson Lane
Santa Clara, CA 95950
(408)988-2211

DAARC
Ellen Reilly
William H. Rorer
500 Virginia Drive
Ft. Washington, PA 19034
(215)628-6547

DATA MANAGEMENT SYSTEMS
Russ Poisson
Seed Software Corp.
2121 Eisenhower Avenue
Alexandria, VA 22314
(703)783-4944

DATATRIEVE/4GL
Donald E. Stern, Jr. c/o
Warner Lambert Company
10 Webster Road
Milford, CT 06460
(203)783-0238

EDUSIG
Fred Bell
Taft College
29 Emmons Park Drive
P.O. Box 1437
Taft, CA 93268
(805)763-4282

GRAPHICS APPLICATION
Michael Anton
P.O. Box 591293
Houston, TX 77259-1293
(713)928-4838

HMS
William Walker
Monsanto Research Corp.
P.O. Box 32 A-152
Miamisburg, OH 45342
(513)865-3557

IAS
Frank Borger
Physics Division
Michael Reese Hospital
Lake Shore Drive at 31st St.
Chicago, IL 60616
(312)791-2515

LANGUAGES & TOOLS
Alan Folsom Jr.
Fischer & Porter Company
E. County Line Road
Warminster, PA 18974
(215)674-7154

LARGE SYSTEMS
Michael Joy
1st Church of Christ
Scientist
Boston, MA 02115
(617)262-2300 x3903

MUMPS
Janet Berryman
2405 N. Bush
Santa Ana, CA 92706
(714)953-1025

NETWORKS
Vicki Hancock
2510 Limestone Lane
Garland, TX 75040
(214)495-7353

OFFICE AUTOMATION
Therese LeBlanc
275 London
Wheeling IL 60090
(312)459-1784

PERSONAL COMPUTER
Kenneth LeFebvre
Sytek, Inc.
19 Church Street
P.O. Box 128
Berea, OH 44017-0128

RSTS
Charles Mustain
Stark County Local School System
Dept. of Education Service Ctr.
7800 Columbus Road NE
Louisville, OH 44641
(216)875-1431 x279

RSX
Bruce Mitchell
Machine Intelligence & Industry Magic
P.O. Box 816
Byron, MN 55920
(507)775-6268

RT
Bill Leroy
The Software House, Inc.
2964 Peachtre RDNW #320
P.O. Box 52661
Atlanta, GA 30355
(404)231-1484

SITE MANAGEMENT & TRAINING
Gregory Brooks
Washington University
Behavior Research Lab.
1420 Gratton St.
St. Louis, MO 63104
(314)241-7600 x257

UNISIG
James Livingston
Measurex Corp.
1 Results Way
Cupertino, CA 95014
(408)255-1500 x4468

VAX SYSTEMS
Larry Kilgallen
c/o DECUS Office
219 Boston Post Road (BP02)
Marlboro, MA 01752-1850

SUBMITTING ARTICLES TO THE HMS SIG NEWSLETTER

The purpose of the HMS SIG Newsletter is to serve as a forum to share information related to DEC hardware with the members of the SIG. As such, the existence of the newsletter is entirely dependent on your contributions. If you have an HHK item, a better or safer way to do something, product news, a tutorial article of general interest, etc., we are interested in publishing it in the newsletter. It is intended that the HMS SIG Newsletter be published at least four times a year.

There are several ways to submit material for the newsletter:

- o The Hardware Submission Form in the back of the newsletter can be used for brief items (there is not enough room if you have a lot to say).
- o You can send me camera-ready hard-copy (this saves me a lot of typing).
- o I will accept submissions on floppys. I can handle RX50's or 8" diskettes (either density, single or double sided). I prefer RT-11 format, if possible, but I can probably handle RSX or VMS stuff somehow. I will return your diskette(s), of course.
- o Those of you that have access to DCS can send things to username WALKER. I check DCS daily.
- o I am also on CompuServe as "Bill Walker 71066,24".

In any event, <u>if you have anything to submit, send it!</u> If it is a mess, but I can read it, I will get it in the newsletter somehow. Finally, if you have any question about submitting material, call me. My telephone number is listed below.

Contributions can be sent to:

```
HMS Editor                           William K. Walker
DECUS                   OR           Monsanto Research Corp.
BP02                    ==           P.O. Box 32   A-152
249 Northboro Road                   Miamisburg, OH   45342
Marlboro, MA 01752                   (513) 865-3557
```

If you need to get something to me quickly, send it to my work address.

DECUS

DECUS SUBSCRIPTION SERVICE
SIGs NEWSLETTERS
U.S. CHAPTER MEMBERS ONLY

As a member of DECUS U.S. Chapter, you are entitled to contribute and subscribe to the DECUS monthly publication, **SIGs Newsletters.** You also have the opportunity to subscribe to the Symposia Proceedings which are a compilation of the reports from various speakers at the U.S. National DECUS Symposia.

- **No Purchase Orders will be accepted.**
- The order form below must be used as an invoice.
- All checks must be made payable to DECUS.
- All orders MUST be paid in full.
- No refunds will be made.
- The address provided below will be used for all DECUS mailings; i.e. Membership, Subscription Service and Symposia.
- SIGs Newsletters Price is for a one-year subscription beginning the month following receipt of payment.

Name ____________________ DECUS Member No. ____________________
Company ____________________
Address ____________________
____
____
City ____________________ State ____________ Zip ____________
Phone ____________________

Subscription Service Offering	Qty.	Unit Price	Total
SIGs Newsletters	______	$35.00	______
Fall '85 Proceedings (FA5)	______	15.00	______
Spring '86 Proceedings (SP6)	______	15.00	______
Fall '86 Proceedings (FA6)	______	15.00	______
Spring '87 Proceedings (SP7)	______	15.00	______

TOTAL COST OF SUBSCRIPTION $____________________

□ MASTERCARD □ VISA □ DINERS CLUB/CARTE BLANCHE®

_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ Exp. Date ____________________

I understand that there will be no refunds even if I decide to cancel my subscription.

Signature: ____________________

FOR DIGITAL EMPLOYEES ONLY

Badge No. ____________________ CC: ____________
CC Mgr. Name ____________________
CC Mgr. Signature ____________________

FOR DECUS OFFICE ONLY

Check No. ____________
Bank No. ____________
Amount $ ____________

Subscription Service, DECUS (BP02), 219 Boston Post Road, Marlboro, MA 01752-1850, (617) 480-3418.

□ □ □ □ □ □

DECUS U.S.CHAPTER
APPLICATION FOR MEMBERSHIP

□ **New Membership** □ **Update to current membership profile Current DECUS Member. #** ______________________

NOTE: PLEASE PRINT CLEARLY OR TYPE!

PLEASE PROVIDE A COMPLETE MAILING ADDRESS, INCLUDE ZIP CODE IN ACCORDANCE WITH POSTAL REGULATIONS FOR YOUR LOCALITY.

ARE YOU AN EMPLOYEE OF DIGITAL EQUIPMENT CORPORATION? □ **YES** □ **NO**

Name: ____
(first) (Middle Intial) (Last/Family Name)

Company: ____

Address: ____

____

____

City/Town: ____________________ State: ____________________ Zip: __________

____

Telephone: Home ( ) ______________ Work ( ) ______________

HOW DID YOU LEARN ABOUT DECUS? Please check applicable item.

1 □ ANOTHER DECUS MEMBER
2 □ SYMPOSIA
8 □ DECUS CHAPTER OFFICE
10 □ DIGITAL STORE
4 □ DIGITAL SALES
5 □ HARDWARE PACKAGE
6 □ SOFTWARE PACKAGE
12 □ ADVERTISING
13 □ LOCAL USER GROUP
14 □ SPECIAL INTEREST GROUP
7 □ SOFTWARE DESPATCH
(DIGITAL Newsletter)

DO YOU WISH TO BE INCLUDED IN MAILINGS CONDUCTED BY DIGITAL (for Marketing purposes etc.?) □ Permission
□ Refusal

TYPE OF DIGITAL HARDWARE USED: Please check those applicable to you.

20 □ DECMATE
82 □ DECsystem-10
83 □ DECSYSTEM-20
52 □ LSI-11
3 □ PDP-8 FAMILY
50 □ PDP-11 FAMILY
21 □ PROFESSIONAL
22 □ RAINBOW
54 □ VAX FAMILY
5 □ WPS-8
51 □ WPS-11

MAJOR OPERATING SYSTEMS? LANGUAGES USED: Please check those applicable to you

1 □ ADA
2 □ ALGOL
5 □ APL
7 □ BASIC
17 □ BLISS
19 □ C
22 □ COBOL
26 □ CORAL-66
28 □ COS
34 □ DATATRIEVE
35 □ DBMS
38 □ DECnet
43 □ DIBOL
45 □ DOS-11
47 □ FOCAL
48 □ FORTRAN
51 □ GAMMA
110 □ IAS
53 □ IQL
58 □ MACRO
65 □ MUMPS
67 □ OS/8
68 □ PASCAL
72 □ PL-11
92 □ RPG
81 □ RSTS/E
83 □ RSX
91 □ RMS
109 □ RT-11
97 □ TECO
70 □ TOPS-10
71 □ TOPS-20
104 □ VMS
107 □ WPS-8

TYPE OF BUSINESS (ENVIRONMENT)/COMPUTER APPLICATIONS
Please check that which best describes your business/application

21 □ ACCOUNTANCY
7 □ BANK
64 □ BUSINESS/COMMERCIAL
74 □ BUSINESS/INFORMATION SYSTEMS
57 □ CHEMISTRY
54 □ CLINICAL LABORATORY
63 □ COMPUTATION
11 □ CONSUMER ELECTRONICS
18 □ CONSULTANT
72 □ DATA ACQUISITION
52 □ DATA COMMUNICATIONS
13 □ DATA PROCESSING SERVICES
71 □ DATA REDUCTION
17 □ DIGITAL EMPLOYEE-ENGINEERING
15 □ DIGITAL EMPLOYEE-MARKETING
16 □ DIGITAL EMPLOYEE-SERVICE GROUP
60 □ EDUCATIONAL ADMINISTRATION
1 □ EDUCATION/PRIMARY
2 □ EDUCATION/SECONDARY
61 □ EDUCATION-TECHNOLOGY
3 □ EDUCATION/UNIVERSITY
67 □ ENGINEERING
65 □ FINANCE/ACCOUNTING
77 □ GOVERNMENT
75 □ GRAPHICS
4 □ HOSPITAL
62 □ INDUSTRIAL
55 □ LABORATORY/SCIENTIFIC
14 □ LIBRARY
58 □ LIFE SCIENCES
70 □ MANUFACTURING
79 □ MARKETING
59 □ MEDICAL RESEARCH
6 □ MILITARY INSTALLATION
73 □ NUMERICAL CONTROL
68 □ OEM-COMMERCIAL
78 □ OEM-TECHNICAL
56 □ PHYSICAL SCIENCES
20 □ RESEARCH/DEVELOPMENT
10 □ RETAIL
76 □ SOFTWARE DEVELOPMENT
53 □ TELECOMMUNICATIONS
19 □ TELEPHONE/UTILITIES
51 □ TIMESHARING
80 □ TRAINING/INSTRUCTION
66 □ TYPESETTING/PUBLICATION

SPECIAL INTEREST GROUP (SIGs) ENROLLMENT
I wish to participate in the following DECUS U.S. Chapter Special Interest Groups.

33 □ APL SIG
2 □ COMMERCIAL LANGUAGES
6 □ DATA MGMT.SYS.
5 □ DATATRIEVE
7 □ BUSINESS APPL.
8 □ EDUSIG
10 □ GRAPHICS APPL
11 □ HARDWARE AND MICRO
35 □ IAS
31 □ DAARC(LABS)
27 □ LARGE SYSTEMS
16 □ LANG. AND TOOLS
14 □ MUMPS
15 □ NETWORKS
34 □ OFFICE AUTOMATION
36 □ PERSONAL COMPUTER
18 □ RSTS/E
17 □ RSX
19 □ RT-11
32 □ SITE MGMT.& TRNG
21 □ UNISIG
26 □ VAX SYSTEMS

JOB TITLE/POSITION - Please check:

1 □ CORPORATE STAFF
2 □ DIVISION OR DEPARTMENT STAFF
3 □ SYSTEMS ANALYSIS
4 □ APPLICATIONS PROGRAMMING
5 □ SYSTEMS ANALYSIS/PROGRAMMING
6 □ OPERATING SYSTEM PROGRAMMING
7 □ DATABASE ADMINISTRATION
8 □ DATA COMMUNICATIONS/TELECOMMUNICATIONS
9 □ COMPUTER OPERATIONS
10 □ PRODUCTION CONTROL
101 □ CORPORATE DIRECTOR OF DP/MIS
102 □ ADMINISTRATIVE ASSISTANT
103 □ TECHNICAL ASSISTANT
104 □ SERVICES COORDINATOR
105 □ MANAGER
106 □ ANALYST
107 □ PROGRAMMER
108 □ DATABASE MANAGER
109 □ DATABASE ADMINISTRATOR
110 □ MANAGER OF DP OPERATIONS

CITIZEN OF UNITED STATES OF AMERICA? □ Yes □ No Country: ____________________

Signature: ____________________ Date: ____________________

Forward To:

DECUS U.S. CHAPTER, MEMBERSHIP PROCESSING GROUP
219 BOSTON POST ROAD
MARLBORO, MA 01752, USA
PHONE: (617) 480-3418

Ask the WOMBAT WIZARD

Submission Form

To submit a problem to the WIZARD, please fill out the form below and send it to:

WW Editor & PIR Coordinator
Philip A. Naecker
Consulting Engineer
3011 N. Mount Curve Ave
Altadena, CA 91001

Name:______________________________ DECUS Membership No. __________

Affiliation:____

Address:____

____

____

Telephone Number:____

Statement of Problem:____

____

____

____

____

____

Please following the following guidelines when submitting support material:

1. If you are trying to demonstrate a method or a concept, please simplify the procedures, records, and other information to the shortest form possible.

2. Annotate your attachments. Simple comments or hand-written notes ("Everything worked until I added this statement.") go a long way toward identifying the problem.

3. Keep an exact copy of what you send. And number the pages on both copies. But send everything that is related to your question, even remotely.

4. If you would like a direct response or would like your materials returned, please don't forget to include a stamped, self-addressed envelope large enough to hold the materials you send.

(fold here)

Donald E. Stern, Jr., DTR/4GL SIG Newsletter Editor
Warner Lambert Company
10 Webster Road
Milford, CT 06460

(fold here)

Product Improvement Request Submission Form

Submittor: ______________________________ DECUS Membership NO: ____________

Firm: ______________________________ Telephone: (___) ___-____

Address: ____

____

Product or Products: ____

____

How to write a PIR

A PIR should be directed at a specific product or group of products. Be sure to give the full name of the product(s) and version numbers if applicable. Describe the functionality you would like to see in as complete terms as possible. Don't assume that the PIR editors or software developers know how it is done in some other software product - state specifically how you want the software to function. Provide justification of your request and give an example of its use. If you can, suggest a possible implementation of your request.

Abstract: (Please limit to one or two short sentences.)

Description and Examples: (Use additional pages as necessary.)

affix
proper
postage

PIR Editor, Philip A. Naecker
Consulting Software Engineer
3011 North Mount Curve Avenue
Altadena, CA 91001
USA

IAS WHIMS

WHAT: (Describe your WHIM) (Please print or type)

WHY: (Describe the reason for the WHIM)

HOW: (Make any suggestions for a possible implementation

Name: ______________________

Company: ______________________

____

Address: ______________________

____

____

Phone: ______________________

Please mail to:

Kathleen M. Anderson
EATON Information Management
Systems Division
2017 Cunningham Drive
Suite 208
Hampton, Virginia 23666

Phone: (804) 326-1941

1986 Large Systems SIG Menu Ballot

DECUS Membership Number:

Category 1
TOPS-10 Menu Items

( ) 1.1 Command Line Recall with Editing

( ) 1.2 Fix Disk Error Recovery and Reporting Code

( ) 1.3 Speed up File I-O

( ) 1.4 Create a SET POLICY Command

( ) 1.5 Fix RSX20F Ctl-S/Ctl-Q

( ) 1.6 Provide Support for Extended Addressing in FORTRAN

( ) 1.7 Restrict 7.04 to RAMP Issues

( ) 1.8 Diagnostics for Tri-SMP

( ) 1.9 Session audit trails

( ) 1.10 Continue development for Corporate Communications Protocols

( ) 1.11 Get Rid of Dispatch: do it On Line

( ) 1.12 Finish DECnet-10 Implementation of the DDP Device

( ) 1.13 Add AUTHOR Selector to the DIRECT Command

( ) 1.14 Provide TCP/IP on TOPS-10

( ) 1.15 TOPS-10 Documentation in Machine-Readable Form

( ) 1.16 Make TOPS-10 sources available to all

( ) 1.17 Provide a network-wide Galaxy

( ) 1.18 Provide a Structured HELP Facility, a la VMS

( ) 1.19 Bring all TOPS-10 Languages up to Current ANSI Level

Category 2
TOPS-20 Menu Items

( ) 2.1 Make TOPS-20 sources available to all

( ) 2.2 Provide TCP/IP Support in TOPS-20 V6.1 Free of Charge

( ) 2.3 Continue development for Corporate Communications Protocols

( ) 2.4 TOPS-20 Documentation in Machine-Readable Form

( ) 2.5 Fix RSX20F Ctl-S/Ctl-Q

( ) 2.6 Get Rid of Dispatch: do it On Line

( ) 2.7 GTJFN to Provide Partial Recognition of Filenames

( ) 2.8 Allow VMS-like relative directory specifications

( ) 2.9 Provide a Way to Determine Terminal Inactivity

( ) 2.10 Implement Command Line Editing in the Monitor Itself

( ) 2.11 Allow a Non-Privileged User to Check the Status

( ) 2.12 Provide a network-wide Galaxy

( ) 2.13 Make LNMST Reveal Another Job's Logical Names

( ) 2.14 Allow COPY to Write More than One Record per Block

( ) 2.15 Option to GTJFN for logical names

( ) 2.16 Improve date and time handling

( ) 2.17 Implement Access Control List Facility

( ) 2.18 Make GTJFN Understand AND, OR, NOT and Parentheses

( ) 2.19 Option to Make Any Structure obey PS: access rights

( ) 2.20 Provide a Mechanism (JSYS?) to Allow Default Forms

( ) 2.21 Provide recognition input for structure names

( ) 2.22 Support a File Comment String in the FDB

( ) 2.23 Standardize Peripheral Interfaces (PHYSIO)

( ) 2.24 Move All Terminal Support Out of the Monitor and EXEC

( ) 2.25 Implement UNIX-like Pipes in TOPS-20

( ) 2.26 Provide a Warm Boot Facility for the DEC-20

( ) 2.27 Treat One-Page Files Specially

( ) 2.28 Support 3- and 4-System Clusters

( ) 2.29 Support Cluster-wide IPCF

( ) 2.30 Support Cluster-wide Galaxy Queues

( ) 2.31 Support Cluster-wide ENQ/DEQ

( ) 2.32 Support for non-PS: Login

( ) 2.33 Support for HSC-based Tape Drives (TA78)

( ) 2.34 Provide Domain Support for Arpanet

( ) 2.35 Provide Arpanet EGP support

( ) 2.36 Update the Arpanet TCP/IP utilities

( ) 2.37 Implement Arpanet TCP/IP Over Asynchronous Lines

( ) 2.38 Integrate DECnet Support into the EXEC

( ) 2.39 Support Intersystem IPCF Between DEC-20s Running DECnet

( ) 2.40 PRINT/DEST, OPR's ROUTE/NODE and Network Node Names

( ) 2.41 Improve Data Interchange Library (DIL) Performance

( ) 2.42 Allow DIRECTORY-like selection for the DELETE command

( ) 2.43 Provide a LIST Subcommand to DIRECTORY-class Commands

( ) 2.44 Make EXEC Features Assembled with XTND Available to all

( ) 2.45 Allow ATTACH.CMD and DETACH.CMD

( ) 2.46 Prohibit Control-C during the taking of LOGIN.CMD

( ) 2.47 Provide for Forcing Periodic Password Changes

( ) 2.48 Provide an option to enforce minimum password lengths

( ) 2.49 Make TOPS-20 MIC handle all of TOPS-10 MIC's Commands

( ) 2.50 Provide Official Support for PC

( ) 2.51 Provide a Structured HELP Facility, a la VMS

( ) 2.52 Provide an ACCOUNT Subcommand to SYSTAT

( ) 2.53 Have the DISMOUNT Command Report Tape Errors

( ) 2.54 Improve Integration of History and Command Line Editor

( ) 2.55 If GET% Fails Because the File Has Invalid .EXE Format

( ) 2.56 Provide Support for TTY: Devices in LPTSPL

( ) 2.57 Implement a Lesser OPERATOR-type Privilege

( ) 2.58 Implement an INFORMATION QUEUES command

( ) 2.59 Wildcarding on Job and User Names in CANCEL and MODIFY

( ) 2.60 Bring all TOPS-20 Languages up to Current ANSI Level

( ) 2.61 Improve DUMPER performance

( ) 2.62 Add some TOPS-10 BACKUP-like functionality to DUMPER

( ) 2.63 Add a STATUS Command to DUMPER

( ) 2.64 Let SYSDPY Display Programs Running in Non-Zero Sections

( ) 2.65 Support 132-column display in DPY/SYSDPY

( ) 2.66 Teach WATCH to be Careful about its Data File

( ) 2.67 Support long filenames in FILCOM

( ) 2.68 The Autopatch Process is Still Too Complicated

( ) 2.69 Autopatch Should Provide .REL File Replacement

Category 3
VAX-related Menu Items

( ) 3.1 Implement a MOUNT queue in VMS

( ) 3.2 Implement AVR in VMS

( ) 3.3 Provide a BACKUP/OPERATOR capability

( ) 3.4 Enhance VMS MAIL

( ) 3.5 End-to-end encryption of logical links in DECnet-VAX

( ) 3.6 Make VMS prompt for unsupplied DECnet passwords

( ) 3.7 Digital should support TCP/IP on VMS

( ) 3.8 VMS utilities should use a standard output format

( ) 3.9 Provide in-line help in VMS

( ) 3.10 Provide filename completion in VMS

( ) 3.11 VMS should provide an UNDELETE facility

( ) 3.12 Support file date last read

( ) 3.13 Allow a privileged user to 'advise' another terminal.

( ) 3.14 Provide COMPILE-class commands in VMS

( ) 3.15 Maximum length of account string should be increased

( ) 3.16 Project accounting is urgently needed in VMS

( ) 3.17 VMS should provide mechanism for disk space accounting

( ) 3.18 Increase reliability of accounting records

( ) 3.19 SHOW USERS, SHOW SYSTEM should list image names

DATAGRAM

DATAGRAMs are short messages, comments, requests, or answers that are published in NETwords. Please fill in the sections below and send the DATAGRAM to:

Vickie Hancock
NETWords Editor
2510 Limestone Ln.
Garland, Tx. 75040

Title: ______________________________

Message: ______________________________
____
____
____
____
____
____
____
____
____
____
____
____

Your Name: ______________________________
Address: ______________________________
____
Telephone: ______________________________

If this is a reply to a previous DATAGRAM, what #? __

Signature: ______________________________ Date: ____

Place
Stamp
Here

Vickie Hancock
NETWords Editor
2510 Limestone Ln.
Garland, Tx. 75040

Fold Here

PC POSTSCRIPT

PC Postscripts are short requests, comments and responses to be published in the *Postscript Section* of the PC SIG Newsletter. Please respond to the following:

__ Y/N This is a reply to a previous Postscript. __ __ Issue Mo. ______ No.

Title:____

Message:____

____

____

____

____

____

____

____

____

____

____

____

____

Name: ____

Address: ____

____

Phone: (_____) __________-______________

Signature: ________________________________ **Date**__________

PERSONAL COMPUTING SPECIAL INTEREST GROUP
VOLUNTEER FORM

Name ______________________________

Company ______________________________

Address ______________________________

City ______________ State ______________ Zip Code ________

Telephone ______________________________

What special talents do you have? ______________________________

When do you attend symposia?

□ Always

□ East Coast Only

□ West Coast Only

□ Occasional Attendance

□ Other (please specify)

____

____

Please check if you are interested in helping with any of the following activities:

Symposia Related Activities:

□ Session Chairs ______________

□ Campground Volunteer ______________

□ Suite Volunteer ______________

□ DECUS Store ______________

□ Software Clinic ______________

□ Panels ______________ (indicate topics) ______________

□ Technical Sessions ______________ (indicate topics) ______________

□ Articles for Update.Daily

□ Write letters of appreciation

□ Equipment Setup

Ongoing SIG Activities:

□ Working Groups ______________ (indicate which groups) ______________

□ Newsletter ______________________________

□ Public Domain Software Project ______________________________

□ Write Software for Special SIG Needs ______________________________

Other SIG Activities: (please specify) ______________________________

Do you wish to see the PCSIG undertake any activities which it is not currently doing? Please specify.

____

Would you be willing to coordinate the activity you have listed above? □ Yes □ No

Thank you

INPUT/OUTPUT Submission Form

A SIG Information Interchange

Please reprint in the next issue of the Pageswapper

If this is a reply to a previous I/O, which number? ________

Caption: ____

Message: ____

____

____

____

____

____

Contact:

Name ____

Address ____

____

____

____

Telephone ____________________________

Signature _____________________________ Date ________________

Mail this form to: Larry Kilgallen, PAGESWAPPER Editor
Box 81, MIT Station, Cambridge, MA Ø2139-Ø9Ø1, USA

For information about on-line submission, dial (in the United States): (617) 262-683Ø and log in with the username PAGESWAPPER.

Tear out or photocopy reverse to submit an I/O item

Larry Kilgallen, PAGESWAPPER Editor
Box 81, MIT Station
Cambridge, MA 02139-0901
USA

System Improvement Request Submission Form

Page 1 of _____

Submittor: Firm:

Address: Phone:

How to write an SIR:
Describe the capability you would like to see available on VAX systems. Be as specific as possible. Please don't assume we know how it's done on the XYZ system. Justify why the capability would be useful and give an example of its use. If you wish, suggest a possible implementation of your request.

Abstract (Please limit to four lines):

Description and examples (use additional pages if required)

System Improvement Request Submission Form

Tear out or photocopy reverse to submit an SIR

Mark D. Oakley
Battelle Columbus Division
Room 11-6-008
505 King Avenue
Columbus, Ohio 43201-2369
USA

Printed in the U.S.A.

"The Following are Trademarks of Digital Equipment Corporation"

A-to-Z
ALL-IN-1
C.A.S.
CDD
COBOL-81
DAL
DATATRIEVE
DATATRIEVE/4GL
DEC
DECalc
DECgraph
DECmail
DECnet
DECSYSTEM-20/VAX
DECtalk
DECtape
DECUS
FMS
IAS
MACRO
MACRO-32
MicroPDP
MicroVAX
PDP
PDP-11
P/OS
Pro-350
Rainbow
Rdb
Rdb/VMS
RMS
RSX
RSX-11
RSX-11M
RSX-11M-PLUS
RT-11-SJ
RX50
TOPS-10
TOPS-20
ULTRIX
VAX
VAX-11/750
VAXstation
VAX/VMS
VAX VTX
VAXELN
VMS
VT100
VT220
WPS
WPS-PLUS

The information in this document is subject to change without notice and should not be construed as a commitment by Digital Equipment Corporation or DECUS. Digital Equipment Corporation and DECUS assume no responsibility for any errors that may appear in this document.

It is assumed that all articles submitted to the editor of this newsletter are with the authors' permission to publish in any DECUS publication. The articles are the responsibility of the authors and, therefore, DECUS Digital Equipment Corporation, and the editor assume no responsibility of liability for articles or information appearing in the document. The views herein expressed are those of the authors and do not necessarily express the views of DECUS or Digital Equipment Corporation.

Ada is a trademark of the U.S. Government (Ada Joint Program Office); SD-Ada is a trademark of Systems Designers PLC; PL/I, CP/M are trademarks of Digital Research, Inc.; UNIX is a trademark of AT&T Bell Laboratories; IBM-PC is a trademark of International Business Machines Corporation; FinalWord is a trademark of the Unicorn; FORTRAN-8X is a trademark of Microsoft Corporation; TSX-PLUS is a trademark of S&H Computer Systems, Inc.; WordMARC is a trademark of MARC Software International.

DECUS

DECUS SUBSCRIPTION SERVICE
DIGITAL EQUIPMENT COMPUTER SOCIETY
219 BOSTON POST ROAD, (BP02)
MARLBORO, MA 01752-1850

STATUS CHANGE

Please notify us immediately to guarantee continuing receipt of DECUS literature. Allow up to six weeks for change to take effect.

( ) Change of Address
( ) Please Delete My Membership Record
(I Do Not Wish To Remain A Member)

DECUS Membership No: ____________
Name: ____________
Company: ____________
Address: ____________
____
State/Country: ____________
Zip/Postal Code: ____________
____

Mail to: DECUS - Attn: Subscription Service
219 Boston Post Road, BP02
Marlboro, Massachusetts 01752-1850
USA

Affix mailing label here. If label is not available, print old address here. Include name of installation, company, university, etc.