APR-81 MAY-81 VOL III NO IV

AUSTRALIAN UNIX USERS GROUP NEWSLETTER

CONTENTS

AUSTRALIAN UNIX USERS GROUP NEWSLETTER

Next AUUG Meeting

The next meeting will be on the 29th of August at Union College, University of Queensland, Brisbane. You should all have received notification in the mail by now, but if not I have reproduced the information at the end of this issue.

While at Berkeley, Bill Joy expressed interest in coming to this meeting and I should know more soon. If he is coming I shall mail a note to that effect.

In This Issue

I know it's July. I know this issue is a month late. I think it's worth it.

I attended the US winter meeting in Austin, Texas, and a summary of what happened appears in this issue. Numerous papers, glossies and other good stuff have also been included, making this one of the biggest issues yet.

Change of AUUGN Editorship

Two (sort of) applications have been received for the position of AUUGN editor, and money manager and chief query answerer etc etc.

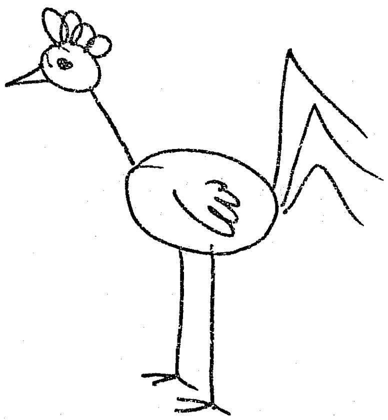
Dave Horsfall asked 'without wishing to dob myself in.....' what was involved and recently Bob Kummerfeld and Chris Rowles have thrown their hats into the ring as a team.

AUUGN Subscription Time Again

Volume three ends with the August-September issue. That time of year is not very far off, so all you people with huge slow moving purchasing departments, GET MOVING! Send subscriptions to volume four to me, made payable to the Australian UNIX Users Group Newsletter (AUUGN). The price is still twelve little Australian dollars.

When the new editor is chosen, I shall print the new address and forward any subscriptions on to him (her, them?).

My Penultimate Art Work



Peter Ivanov
Dept. of Computer Science
Electrical Engineering
University of N.S.W.
PO Box 1
Kensington 2033
AUSTRALIA

(02) 662-3781

Texas UNIX Users Conference
June 23 - 26, 1981

GENERAL CONFERENCE INFORMATION

Message Center

The Thompson Conference Center, 471-4652, will take messages for the participants and place them on the message board in the lobby of the LBJ Auditorium.

Villa Capri -- 476-6171.
Hilton Inn -- 451-5757.

Shuttle Bus Information

Shuttle buses will be provided from the Hilton Inn to the LBJ Auditorium and the Thompson Conference Center. Buses will run from 8:00am-10:00am Tuesday through Friday from 4:30pm-6:30pm Tuesday through Thursday, and from 8:00am-11:00pm Wednesday through Thursday.

Restaurants

A copy of the Austin Conventioneer which contains a list of local restaurants will be available in the lobby of the LBJ Auditorium.

The Thompson Conference Center has a noon lunch -- sandwiches, salads, etc.

Othello Contest

The Othello contest will be held in the Childress Room of the Villa Capr Conference Center on Thursday from 7:00pm - 12:00 midnight.

Vendor Exposition

The following vendors will be exhibiting in the McKinney Room at the Villa Capri Conference Center on Tuesday, 1:00pm - 5:00pm, Wednesday 9:00am - 5:00pm, and Thursday 9:00am - 5:00pm and 7:00pm - 10:00pm.

Able Computer
BBN Computer Corporation
Human Computing Resources
Interactive Systems Corporation
Perkin Elmer and The Wollongong Group
Santa Cruz Operation
Whitesmiths and Western Scientific
Zilog, Inc.

Vendor Presentations

The following vendors will be making presentations in the Sherman Room of the Villa Capri Conference Center on Thursday during the Vendor Exposition, 7:00pm - 10:00pm.

Human Computing Resources
Interactive Systems Corporation
Micro Focus, Inc.
Santa Cruz Operation
The Wollongong Group

June 23	Software Tools Meeting	Joe C. Thompson Center

9:00am - 5:00pm — Chairman: Allen Akin, Georgia Institute of Tech

Open - getting acquainted.
Allen Akin, Georgia Institute of Technology

The Software Tools in PASCAL.
Bill Plauger, Whitesmiths

The Sociology and Evolution of Tool Kits.
John Mashey, Bell Laboratories

Coffee Break

Data Management Tools for Large Data Sets.
Bob Burnett, BATTELLE, Pacific Northwest Laboratory

Writing a Compiler in RATFOR.
Dan Forsyth, Georgia Institute of Technology

11:30am - 1:00pm — Lunch

Enhancements to "ed".
R. R. van Tuyle, Sylvania Systems Corporation

Adding Data Structure Support for Ratfor.
Wade Shaw, EXECUCOM Systems Corporation

TOPS-20 Software Tools.
Bruce Dawson, Digital Equipment Corporation

Break

Cyber Software Tools.
Kirk Webb, The University of Texas

Use of Software Tools in a Second Computer Science Class.
A RATFOR Features Survey.
Walter E. Brown, Moravian College

Special Interest Group Meetings.

State of the Software Tools Group Address.

6:00pm - 9:00pm — Registration — Villa Capri Conference Center Entertainment Center

UNIX Conference Registration will be held in the Villa Capri Entertainment Center on the corner of East 23 1/2 Street and Swisher Street.

A cash bar will be open during the registration. Complimentary hors d'o will be served.

June 24	Wednesday	

9:00am - 11:30am	Conference Registration	LBJ Auditorium Lobby
9:30am - 11:30am	What's New with UNIX?	LBJ Auditorium

Chairman: Wally Wedel, The University of Texas

Opening remarks. Wally Wedel, The University of Texas	15 minutes
What's New at Western Electric? Bob Guffey, Western Electric	60 minutes
Coffee Break (30 minutes)	
What's New at DEC? Joel Magid, Digital Equipment Corporation	15 minutes
What's New at Perkin-Elmer? Roger McKee, Perkin-Elmer	15 minutes
Usenix Business. Lou Katz, President	15 minutes
/usr/group. Bob Marsh, President	15 minutes

11:30am - 1:30pm	Lunch	
1:30pm - 5:00pm	Operating Systems and Utilities	LBJ Auditorium

Chairman: Mike O'Dell, Lawrence Berkeley Laboratory

PDP-11/44: A VAX 11/780 UNIBUS Terminator. George Goble, Purdue University	20 minutes
What is /usr/doc/services? Gene Dronek, Berkeley	25 minutes
Four Megabytes of Memory on an 11/40. Clement Cole, Tektronix	15 minutes
Break (30 minutes)	
LocalNet and UNIX. Sam Leffler, Sytek, Inc.	30 minutes
RT Emulation - Another Viewpoint. Daniel Strick, University of Pittsburg	20 minutes
Bad Block Handling in 4.1 BSD. Bill Shannon, Digital Equipment Corporation	15 minutes
Microprogramming to Improve UNIX Performance on BBN C Machine. Steve Emmerich, BBN Computer	30 minutes

7:00pm - 10:00pm	Conference Reception	Thompson Conference Center

The Conference Reception will be held at the Thompson Conference Center which is adjacent to the LBJ Auditorium. Wine & cheese and Beer & nachos will be served.

June 25 **Thursday**

9:00am - 11:30am	UNIX Languages and Applications	LBJ Auditorium

Chairman: Wally Wedel, The University of Texas

Porting a Microprocessor to UNIX. Craig Forney, PLEXUS Computers	10 minutes
UNIX vs. Godzilla -- UNIX in an IBM Environment. Tom Lyon, Amdahl Corporation	15 minutes
An ARPANET Front-end for Berkeley VAX/VM UNIX. Mike O'Brien, Rand Corporation	20 minutes
A UNIX-based Network Operations Center. Brownwell Charlestrom, BBN Computer	30 minutes
Converting a Virtual RC4000 to a Virtual UNIX. Joshua Knight, Stanford University	10 minutes

Coffee Break (30 minutes)

Cyber C Compiler Development. Bill Lee, The University of Texas	30 minutes
Basing "MAKE"'s on Revision Levels. Gordon Kass, Olivetti	15 minutes

11:30am - 1:30pm	Lunch
1:30pm - 5:00pm	UNIX Languages and Office Automation

Chairman: Jim Peterson, The University of Texas

Concurrent Euclid -- A Brief Outline. Chris Robertson, University of Toronto	10 minutes
Compiler Construction using YACC. Barb Staudt, Moravian College	15 minutes
PEN -- The Design of a Flexible Video Editor. Robert Wells, BBN Computer	20 minutes

Break (30 minutes)

Office Power. Edward Scott, Computer Consoles, Inc.	30 minutes
DEC Bus Interfacing Ken Omohundra, ABLE Computer	30 minutes

7:00pm - 10:00pm | Vendor Exposition | Villa Capri Conference Center

The Vendor Exposition will be held at the Villa Capri Conference Center in the McKinney Room. Adjacent to this exposition will be vendor presentations and the Othello contest.

A cash bar will be open during the Exposition. Complimentary hors d'oeurves will be served.

June 26 | **Friday**

9:00am - 11:30am | Graphics and Data Base Systems | LBJ Auditorium

Chairman: Wally Wedel, The University of Texas

Towards Graphics Device Independence. Adrian Freed, LERS, Paris	10 minutes
Commercial Quality Relational DBMS for UNIX. Roger Sippl, Relational Data Base Systems, Inc.	15 minutes
Porting Mistress to the ONYX John Z. Kornatowski, Rhodnius, Inc.	10 minutes
Educating Users: ITS Approach. James Joyce, International Technical Seminars	30 minutes
Coffee Break (30 minutes)	
Alpha 1: Computer Aided Geometric Design using UNIX. Spencer Thomas, University of Utah	20 minutes
Leroy: Publications Quality Plots Ernie Harkins, University of Colorado	10 minutes

11:30am - 1:30pm | Lunch

1:30pm - 4:30pm | Miscellaneous | LBJ Auditorium

The following is a summary of what I think was said at the Software Tools and USENIX meetings held at the University of Texas at Austin, Tuesday 23rd June through Friday 26th June 1981.

The summary does not purport to be true and should I have misquoted or drawn any wrong conclusions from anything said then the speakers and their institutions have my sincere apologies.

Peter Ivanov
University of N.S.W.
Australia
8/7/81.

Software Tools User Group Meeting
Tuesday, June 23, 1981
Introduction
Chair: Allen Akin, Georgia Tech

Speaker 1: 9:00am Allen Akin, Georgia Tech

Opening Remarks - Getting acquainted

Allen opened the meeting and welcomed all attendees. He ran through a number of procedural matters and details of the rest of the day's activities before introducing the first speaker.

Speaker 2: 9:10am Bill Plauger, Whitesmiths

The software tools in pascal

Kernigan and Plauger have rewritten the software tools (book and tape) in pascal. Addison Wesley are publishing it and the software will be available on tape or discette.

Why did they do it? Bill quoted reasons such as greed, popular demand and as an exercise in portability since the pascal tools run on UNIX-Berkeley, UCSD and Whitesmiths' own pascal.

Why pascal? Among other things, pascal has proved a popular teaching language and has better typing. Why not pascal? Problems encountered include no standards for separate compilation, no standards for command argument passing, no file naming standards and no raw I/O. Significant circumlocution was needed, control flow constructs necessitated the use of a lot of 'done flags' and data typing was too strong (fixed array sizes).

Bill wants it to be known that the pascal version is not just 'RATFOR with semicolons'. A lot of work was done on extending the language without using any extensions, isolating idiosyncrasies outside the majority of source files and documentation. The new tools contain less 'sermons' or lecturing on style. They also do not contain crypt.

Finally the new tools are portable but not fast, binary I/O is not fully supported and there is no seeking primitive.

Speaker 3: 9:40am John Mashey, Bell Labs

The sociology and evolution of tool kits

John's talk might be para-phrased as 'why ugliness comes and stays forever'.

In a fast but interesting talk he covered how programs evolve and how humans and organisations interact with them. Some aspects touched upon were the sociology of programs, programming team psychological roles and the theories of creeping featurism and the ecological niche. Under the heading of psychological roles, three types were illustrated. The 'generator' who creates ideas and codes quickly, the 'analyzer' who watches users, finds unspoken needs and changes to fit reality, and the 'consolidator' who spends much of his time throwing away redundant or unused software.

Creeping featurism would be amusing if it were not so true. One starts with a 'thing' and adds a feature. Then someone else maybe adds a future feature (isn't it a neat thing, and nobody need use it if they don't want to). Then come needed features. When the next version is produced it must be upward compatible (ie. thing + release 2 of the features). Thus you finish with a mountain where a mole hill would have done as well.

Programs evolve through natural selection to fill niches. They may fill a niche and defend it against all attack, or be so general as to fill a large niche, or so small (fast or specialised) as to fill a small niche well, and then there are ice ages, but we won't talk about them.

In summary, 'We have met the enemy, and they is us.'

------ Coffee Break ------

Speaker 4: 10:50am Bob Burnett, BATELLE

Data management tools for large data sets

Bob described a data base system, implemented in RATFOR, on a VAX running VMS that uses information stored with the data base files to describe to the various parts of the data base system the shape of the data contained in the data base.

Used mostly for statistical data bases, utilities available include a formatted file to DB translator, data editor/subset generator, a sort utility and display utility called 'describe'.

Speaker 5: 11:20am Dan Forsyth, Georgia Tech

Writing a compiler in RATFOR

Why write a compiler in RATFOR? First Dan said that in his view it was quite an accomplishment to write a compiler for a prime at all, but RATFOR was chosen because PL/1 was ten times slower, Pascal was flaky and assembler language is, well, nobody in their right minds would do that.

Naturally what sort of compiler would you write but a C compiler. C is seen as a good system programming language and writing one in RATFOR would allow tuning of the compiler. The result is a two pass C compiler passing information in code tree form, producing assembler out.

Apart from RATFOR you need lots of memory, a parser generator, a dynamic storage routine (common block manager) to implement heaps and pointers (available on the ST tape) and structures were done using a macro processor.

------ Lunch ------

Software Tools User Group Meeting
Tuesday, June 23, 1981

Chair:

Speaker 6: 1:00pm R. R. Van Tuyl, Sylvania Systems Corp

Enhancements to ed

An editor should be easy to use with natural intuitive commands. This speaker had experience with TECO, had an HP editor and got the ST editor. The STed was extended with the addition of a 'V' command to display the line context (plus and minus a few lines), a 'P22' command to display the next page of text and limited line editing facilities. There are also commands to break and join lines.

Requires terminals with back-space, speed greater than 2400 baud, an operating system giving good response to interactive jobs and a raw mode of input.

Speaker 7: 1:20pm Dan Klein, Mellon Inst

General assembler generator

Dan described MIRAGE, the Mellon Institute Research Assembler Generator. An assembler is really a program that performs one to one mapping of source to machine code. It need not be stupid, and may even have useful error diagnostics. Each assembler is specified by supplying header information, 'variables', addressing mode specifications, op codes, pseudo op codes and literals.

Assemblers have been generated for 6502, 1802, 6809, DG-Eclipse, PDP11, PDP10, IBM360, 8080, 68000, 8048 and others.

Speaker 8: 1:40pm Wade Shaw, EXECUCOM

Adding data structure support for RATFOR

PIL, a portable implementation language, is a portable language system that runs on several different systems. Each machine has a set of environmental constants used to compile correct code.

Not a very good summary I know, but blame jet lag.

Speaker 9: 2:00pm Bruce Dawson, DEC

TOPS-20 software tools

Bruce gave an amusing talk on getting ST up under TOPS-20. One of the minor troubles encountered was insufficient storage space in the ST for file names. TOPS-20 uses a control-v character to flag a lower case character, and so lower case file names (eg. ´abc´) become ´^va^vb^vc´.

Redirection of standard error was a little difficult for similar reasons, but spawning new processes was significantly easier than some other implementations. The system will be available through DECUS when stable, but if you want it now, send a tape to Bruce at

146 Main St.,
Mainard,
ZK1-3/B21.

decvax!jbd or sultan!jbd.

------ Coffee Break ------

Speaker 10: 2:45pm Kirk Webb, University of Texas

Cyber software tools

Another implementation of ST. Kirk had to battle cyber 6bit display code, and got round it by implementing a 12bit extended display code. The result is very slow and very large but everything is there except kill, pstat, suspend and resume.

Speaker 11: 2:56pm Walter E. Brown, Moravian College

Use of ST in a second computer science class

Moravian College has been using ST in their second computer science course since 1977, based on the ST book, using RATFOR as the teaching language. The first course serves multiple audiences and so the programming languages must not be esoteric (APL, COBOL, C): use a limited subset of RATFOR.

Objectives of the second course include fostering skills at critical reading and analysis of programs, giving experience in programming language constructs, developing insight into the disciplined software development process, and giving experience in non-numeric programs and file processing tasks. Students are taught using carefully constructed examples of real problems which contribute to an environment conducive to good programming.

Evaluations of the course have shown it to be popular (but only after it is over), difficult for faculty members to teach, and it has become almost central to the curriculum. Final recommendations; read the ST epilogue first, use supplimentary reading and use lots of anecdotes and analogies.

Speaker 12: 3:25pm Walter E. Brown, Moravian College

A RATFOR features survey

At the Delaware ST meeting a survey was conducted of possible RATFOR language extensions. The survey covered 50 features which attendees were asked to rate A-F, where A was ´can´t do without it´ and F was ´definitely NO´. Kernigan and Plauger were both there, and their response was that there should be less features. Anyway, the top five, in order of acceptance were:

1. Macros with arguments (by a long way)

2. Installation cook book

3. Built in strings

4. Conditional compilation

5. Better diagnostics

The least liked, most hated first, were:

1. Ability to define symbols (eg ´+´ as ´-´)

2. Change ´[´ and ´]´ to ´{´ and ´}´

3. break n

4. do (k=1,10) body

5. FORTRAN style comments (´c´ in column one)

USENIX Group Meeting
Wednesday, June 24, 1981
What's New with UNIX
Chair: Wally Wedel, University of Texas

Speaker 1: 9:15am Wally Wedel, University of Texas

Opening Remarks

Wally ran through schedule changes and procedural matters and then gave a summary of the computing facilities at the University of Texas, ranging from Cyber-170/750s to PDP11/34s.

Speaker 2: 9:39am Bob Guffey, Western Electric

What's new at Western Electric

Bob is responsible for all software licensing, of which UNIX constitutes about 90%. At WeCo Larry Isley and Bill Murphy are the commercial contacts while educational queries should go to Susan Sellers (hope the spelling is correct). WeCo is adjusting to meet demand and Bob expressed the hope that replies to queries and processing of licenses would be very much faster. Western/Bell would be offering elementary support for government sites and more would be said about this at a later date.

The following information was presented, valid at 1st May 1980.

Licenses

Software	Commercial	Educational	Government	Total
Mini-UNIX	6	98	0	104
V6	89	319	42	450
PWB	38	43	53	134
V7	78	176	18	274
32V	33	74	11	118
Total	224	710	124	1078

Institutions

Software	Commercial	Educational	Government	Total
Mini-UNIX	8	265	0	273
V6	141	775	123	1039
PWB	70	132	22	262
V7	86	423	11	531
32V	36	121	11	531
Total	341	1716	216	2273

Bob also went to considerable effort to convey WeCo's view that software protection comes under the scope of trade secret laws. The thrust of his comments was that disclosure of even one line could be viewed as a breach of license and that distribution to staff or students of licensable material should be kept to a minimum.

Some new packages now available are:

Package	Initial	Additional	Binary	Educational
UNIX/1100	30000	10000	12000	400
UNIX/6000	30000	10000	12000	400
C/SEL	5000	2500	2500	NA
C/DATA	5000	2500	2500	NA
C/8086	5000	2500	2500	NA

UNIX/1100 is UNIX for the UNIVAC 1100, running under EXEC8. UNIX/6000 is really a set of utilities for use on the Honeywell 6000 to make GCOS look like UNIX. C for the Systems Equipment Lab 32bit machine, C for Data General 16bit and C for 8086 were not considered for educational release. At least I think that is what the 'NA' means.

UNIVAC and Honeywell UNIX had progressed past Bell 3.0 and have been reduced back to level7 technology for release. Should 3.0 be licensed then more recent versions will be made available.

The licensing review board currently has several packages before it. Of interest are a new troff (maybe August-September), UNIX 3.0 and UNIX/370.

Some other comments: Writer's workbench is later technology and so will only be released should later versions of UNIX be licensed; Exchange of information between L6, L7, etc. licensees is under consideration; OEM fees schedules may be expanded; Licensees upgrading to later versions of UNIX may receive substantial reductions in fees. Updates for C/370 will be released, free of charge, to all license holders, hopefully soon.

In all these comments Bob would not be tied down on timing. He emphasised the feeling that large wheels turn slowly.

------ Coffee Break ------

Speaker 3: 11:08am Rich Ptak, DEC

What's new at DEC

Rich sketched out changes made in the DEC UNIX group. Bill Mundsen has been kicked upstairs and replaced by Dick Pigman. Rich and Joel Magio (or Magid?) are listed under UNIX system management, while UNIX engineers are Jim Barllay, Gerry Brenner, Fred Canter, Bill Shannon and Armando Stettner.

The DEC UNIX tape is getting better. Send them a tape, a copy of your V7 license and your DEC system configuration. The tape when returned should come up with minimum trouble. The address is:

DEC
MK1-1/D29
Continental Blvd.
Merrimak NH 03054

(603) 884 4754

People are being trained in field service in a UNIX environment now.

Speaker 4: 11:22am Dave Preston, Perkin-Elmer

What's new at Perkin-Elmer

Apart from the terrible pronunciation of certain Australian place names, Dave tells us that P-E in combination with The Wollongong Group now market UNIX for P-E machines. P-E/TWG have more than 22 systems running used for simulation, CAD/CAM and software development in Canada, USA, Australia and Europe.

He stated future P-E/TWG directions include creation and maintenance of a standard UNIX environment, migration of key P-E products to UNIX, enhancements to V7 and incorporation of other products such as networking, DBMS, compilers and micro-processor development systems.

Speaker 5: 11:31am Bob Marsh, /usr/group

/usr/group

Bob is the president of "/usr/group", a commercially oriented user group formed to encourage UNIX as a defacto standard operating system and to promote software compatability. Incorporated in California it costs $100 per general member and $30 per associate.

There are presently six subcommittees looking at licensing, standards, publications, programmers, vendors and end users. For further information contact Plexus Computers, (408) 988 1755.

Speaker 6: 11:42am Lou Katz, Usenix

Usenix business

On the topic of numbers, 400-500 people attended this conference. The next meeting will be in Santa Monica, California, 27-29 January 1982 at the Miramar Sheraton Hotel, hosted by Interactive Systems. Information from the

USENIX Association Office
Box 8,
Rockefeller University,
1230 New York Ave,
NY 10021,

(212) 360 1182.

Usenix is now incorporated in the state of Delaware. People are needed to serve on various committees and also to staff tutorials planned at the Santa Monica meeting along the lines of ACM-SIG tutorials.

------ LUNCH ------

USENIX Group Meeting
Wednesday, June 24, 1981
Operating Systems and Utilities
Chair: Mike O'Dell, LBL

Speaker 7: 1:34pm George Goble, Purdue University

PDP-11/44: A VAX 11/780 UNIBUS Terminator

Mike O'Dell subtitled this talk 'Super charging your VAX'. George described a connection between a PDP-11/44 and a VAX-11/780 achieved by pulling out the VAX 9044 UNIBUS terminator and inserting the PDP UNIBUS. This is all very well, but why?

1. Offload TTY and network I/O from the VAX

2. Network route-through in case of a VAX failure

3. To take care of the supposed UBA problems

4. Use as a display system for debugging

In practice though:

1. Use of 4BSD has reduced the I/O problem to manageable proportions

2. The VAX is very reliable, most failures are devices or power supplies

3. The UBAs at Purdue work, unlike those reported at Bell and Berkeley.

4. Proved to be useful for system monitoring as illustrated by the film shown during the talk. The film displayed a modified 'ps' output obtained by a stand-alone program running in the PDP-11/44 accessing VAX memory, outputting to a DMA home built terminal. The display was updated ten times a second. You can also use the 11/44 to display VAX memory.

One hardware modification was needed to the PDP-11/44, a strapping to make addresses larger than 64KW appear as a device register. Should work for all PDP-11s but some (34, 44, 60 etc) must have the CPUs running so as not to hang the UBA.

If you are interested write to:

George Goble (Software) (317) 493 9479
Mike Marsh (Hardware) (317) 494 4226
Electrical Engineering Department
Purdue University,
West Lafayette
Indiana 47907
USA

or on uucp

pur-ee!ghg, ucbvax!pur-ee!ghg, decvax!pur-ee!ghg and
pur-ee!mike, ucb!pur-ee!mike, decvax!pur-ee!mike.

Speaker 8: 2:00pm Gene Dronek, Berkeley

What is /usr/doc/services?

This is an organisation, newly hatched, for the distribution, reprinting, collection, refereeing and mailing of UNIX documentation. It is interested particularly in articles, writeups, tutorials and even books.

Gene and friends plan to produce a catalogue by October 1981. The address

2490 Channing #503
Berkeley CA 94704

(415) 731 6154

Speaker 9: 2:18pm Clement Cole, Tektronics

Four megabytes of memory on an 11/40

Clement described the use of an Able Computer Products 'ENABLE' device to attach up to four megabytes to a PDP-11/40 style processor. Some reasons why you would want to are:

1. L6 was difficult to squeeze onto a 40, and L7 is worse. Schemes developed to solve this problem (mapped buffers and overlayed kernels) all result in the system getting larger at the expense of remaining user memory.

2. PDP-11/40 type processors are prevalent and it is often difficult to justify the money to replace them. It is cheaper to refurbish existing machines.

The system changes are on the conference tape, and a paper appears later in this issue of AUUGN. Finally, here are some benchmarks run on a range of processors:

CPU	Memory	Disc	Real	Jser	Sys
11/70	2MB	2xRP04	13:06	5:02	3:12
11/44	0.25MB	RM02	23:35	7:42	6:38
11/34	0.25MB	2xRL01	58:23	17:12	13:08
11/34+ENABLE	0.25MB	2xRL01	42:19	14:41	10:29

------- Coffee Break -------

Speaker 10: 3:00pm Sam Leffler, Sytek, Inc.

LocalNet and UNIX

Sam gave a fast, thorough description of a network package available from:

Sytek Inc.
1153 Bordeaux Dr.
Sunnyvale CA 94086

(408) 734 9000

ucbvax!menlo70!sytek!sam

LocalNet is designed to allow complete interconnectivity between any two nodes at different speeds. Supports host-host traffic at 1.5Mbits and terminal transmission and concentration at 120Kbits. Implemented as a number of modular units. The UNIX host interface, a System40, connects to the UNIBUS doing DMA transfers out to a net adapter.

Speaker 11: 3:30pm Daniel R. Strick, University of Pittsburg

RT emulation - another viewpoint

"RT11 is a system for people without enough money to buy a real system." With that quote Daniel opened his talk outlining an RT11 environment emulation package. The emulator lives in the top end of a 32KW address space, and the RT11 program survives down the bottom. Dan has implemented all bar nine of the RT11 system requests. The emulator runs on PDP11s and VAXes and is available from

Daniel R Strick
833 LIS Building
University of Pittsburg
Pittsburg PA 15260

Speaker 12: 3:53pm Bill Shannon, DEC

Bad block handling in 4.1BSD

Gone are the days when UNIX users had to pay large sums of money for perfect discs. The bad block handling in 4.1BSD is based on the DEC STD 144 and works with RK06, RK07, RM03, RM05, RM80, RP07 but not with RP06s. The support code is mostly in the HP and RK drivers. It makes use of the bad sector information stored at the end of the disc initialised by DEC standard formatters (EVRAC). The program BAD144 displays and maintains the table.

Each driver's table is initialised when first entered or when the pack is changed (using the VolValid bit as the pack change indicator). The costs are a small amount more code, nothing for good block accesses and one sector table per drive, but the replacement sectors must be perfect. Uses the BadSectorError indication from the drive.

Speaker 13: 4:04pm Brownell Chalstrom, BBN

A UNIX-based network operations center

The NOC runs on a PDP-11/70 (being ported to a C70) designed for ARPA and like networks (long haul packet switched net of nodes and hosts). The NOC is a host on ARPA, providing the following services

- Net status displays
- Fault isolation
- Debugging aids for network programming
- Network software management
- Diagnostic and statistics collection
- Management reports and long range planning
- Information distribution and complaint handling

NOC also provides information displays of node/host status, ´at a glance´ colour summaries of the network (the redder a link the higher the traffic rate over it) and logs tailored to suit users. In fault isolation you can prod and poke pieces of net hardware and software to help isolate problems.

The future holds promise of multiple NOCs on a net, multiple nets on a NOC, improved reliability and expansion.

Speaker 14: 4:35pm Walter D. Lazear, AF Data Services Center

Air force management reporting system

The air force data services center was faced with a management problem. On the large number of systems they are required support

- Customer impressions of service are unreliable
- No performance bench marks were available
- No daily performance comparisons were available
- No common reporting system was available
- Diagnostics were too verbose or terse; needed summaries and consistency

The center wide solution implemented provides

- Standard daily reports on performance, availability, utilisation and workload
- Monthly summaries
- Consistent reporting

The UNIX implementation involved no kernel changes, a probe program to sample kernel, a management summary (merging of SA and AC with some calculations) and a management report giving management and high water mark reports. The benefits include

- automatic tracking of failures
- consistent analysis techniques
- documentation of the impact of system alterations
- reliable sizing of kernel tables in binaries supplied to end users
- Documentation and measurement of system size verses user community.

Write to

Walter D. Lazear
Air Force Data Services Center
The Pentagon - Room 1D988
Washington DC 20330.

USENIX Group Meeting
Thursday, June 25, 1981
UNIX Languages and Applications
Chair: Wally Wedel, University of Texas

Speaker 15: 9:08am Craig Forney, PLEXUS Computers

Porting a microprocessor to UNIX

This talk really described the 'design of a computer for UNIX'. Craig ran through design goals, standards, reliability, usability and performance of the Plexus P/40. Much of this information is reproduced in the company literature appearing later in this issue of AUUGN.

Speaker 16: 9:22am Tom Lyon, Amdahl Corp.

UNIX vs Godzilla - UNIX in an IBM environment

Amdahl call their UNIX produce UTS. UTS, similar to V7, fits well into existing IBM environments, has been working 'in house' at Amdahl for two years and is now available under VM/370.

Tom described the Amdahl architecture as having

- Fast CPUs but with high interrupt overheads
- Lots of fairly cheap real memory
- Under VM, lots of even cheaper virtual memory
- Small discs, about the size of an RP06

UNIX adaptations for the IBM environment include a 2MB in core file system (/tmp) and 4K blocks providing drastic decrease in I/O, decrease in interrupts caused by STDIO buffering and increased memory usage.

UTS has 'stupid support' for ASCII terminals caused by front end software simulating stupid hardware, and good support for 3270 type terminals. UTS interfaces

- By batch transfers to MVS and VM
- Interactively allowing UTS to appear as a 3270 to MVS and VM
- Through clean interfaces to higher level software such as PL1 (interface generates the JCL, runs the job and returns the results), OSIRIS (auto logon to TSO and enter the data base system) and CRON which periodically submits jobs to backup MVS files and compress TSO data sets.

People's perceptions of UTS are interesting. Operators tend to cause more harm than good as they are unfamiliar with systems such as UTS. How can a quiescent machine suddenly burst into activity (CRON) at 3am? New users flock to UTS while old IBM users may change, but some don't. Typical complaints such as "It's a toy", "it's inefficient" or "not enough features" (like no VSAM) are answered by "yes it is fun, but useful", "productivity is worth inefficiency" and it has "not too many features".

This was a good talk (quote on TSO, 'It may be slow, but its hard to use!') with amusing drawings of gorillas being harased by light planes labelled UNIX. The software is available, but you may have to extract it from Amdahl since they are not officially in the software business.

Speaker 17: 9:45am Joshua Knight, Stanford University

Converting a virtual RC4000 to a virtual UNIX

The RC4000, a 24bit Danish machine, was chosen years ago for its nice environment with a primary application language ALGOL68. The RC4000 was emulated on a PDP-11/45 computer and used primarily for data analysis. The 11/45 was also used to control a solar observatory using MACRO under DOS via a PDP-11/20.

With the previous scene set, enter a VAX-11/780 running Berkeley 4BSD. The task was thus to write an RC4000 emulator and get DOS MACRO running in compatability mode. The conversion process was fairly straight forward and copy of MACRO and a minor compatability mode change to 4BSD is available on the conference tape.

Speaker 18: 10:02am Steve Emmerich, BBN

Microprogramming to improve UNIX performance on BBN C machine

I am sure you have all heard of the BBN C machine so I do not plan to summarise it here. The basic thrust of Steve's talk was that the micro-codable nature of the C machine allows:

- Tailoring of macro instruction set to support C operations
- Incorporation of system monitoring and debugging functions
- Implementation of higher level functions in firmware with corresponding speed increases.

Micro-coding certain routines has provided three fold increases in fork speed and block I/O is four times faster. In the future it is planned to micro-code getc/putc from STDIO and some floating point operations.

Speaker 19: 10:45am Mike O'Dell, LBL

ARPA front end for Berkeley VM UNIX

This summary of work proceeding on an ARPA front end for Berkeley UNIX was presented at short notice by Mike O'Dell. Unfortunately Mike speaks very quickly and on a topic that obviously means something to him, but I am afraid that as I am not an 'ARPA person' most of the collection of acronyms and buzz words was flying way above my head. I do not feel I could produce a coherent summary of what was said. This no reflection on Mike since it was clear that quite a few people found what he said interesting.

------ Coffee Break ------

Speaker 20: 11:12am Bill Lee, University of Texas

Cyber C compiler development

The motivation behind putting C on a Cyber was that there was no high level systems language available, C is a common choice for such a language, no one else had done it and it seemed that it should provide a good return on the investment.

Plan A was to use lex for the lexical analyzer, yacc for the parser and ccg (a cyber common code generator) to produce the code. It turned out to be not as easy as it looked, even with lex and yacc, and generating code ´on the fly´ from the parser was not the way to go. Also the ccg was written in assembler (´some of it even had comments!´) and proved very difficult to modify and interface to, as it was designed to run in conjunction with a cyber fortran compiler.

Plan B was similar but involved writing a new code generator. Differences between PDP11 C and Cyber C are

Item	PDP11 C	Cyber C
char	8bits	15bits
short	16bits	18bits (B register size)
int	16bits	60bits
long	32bits	60bits
float	32bits	60bits
double	64bits	60bits
external ids	7chars, upper/lower case	7chars, upper case only
character evaluation	right to left	left to right
arg evaluation	right to left	left to right

The runtime library is a bit big, but fast. C is a little larger than RATFOR from software tools and about the same speed. The compiler occupies (in two parts, cpp and cgen) 26000 words (18000 to load) and 28000 words (24000 to load). To get it you need a V7 license for your Cyber, the Cyber loader, CMM (the common memory manager) and compass assembler. Selected experimental sites will be chosen in August 1981, with general release some time after. Contact

Wally Wedel,
University of Texas Computation Center,
Austin TX 78752,

(512) 471 3241.

Speaker 21: 11:50am Gordon Kass, Olivetti

Basing MAKE's on revision levels

Gordon has made changes to MAKE to allow a better interface to SCCS maintained files. The new code in MAKE causes SCCS commands to be issued to obtain a correct source version if one does not exist. MAKE also understands a revision level specifier at the front of a MAKE file to specify levels previous to the current one.

Other changes to MAKE include

1. a '-z' option to give a readable trace of the operations MAKE is performing

2. improved interruptability

3. the '-n' option is transmitted to invoked MAKEs correctly

4. a '-I' option to allow include files to be combined into MAKE files

5. fixed '$?' bugs

6. error messages given when undefined macros are used.

For further information contact Gordon at Olivetti Advanced Technology Center, 10430 So. De Anza Blvd., Cuperinto CA 95014, (408) 996 3867

Speaker 22: 12:10pm Bill Jolitz, Berkeley

2BSD

The new 2BSD distribution will be ready to ship in a few weeks. It has an F77 that runs on I/D and non I/D PDPs. It has many similarities with 4BSD, less with L7. It has a fast file system, setuid shell scripts, a real-time option and code for file system robustness (by Tom Ferentz). FS robustness code adds 11% overhead and 250% overhead in pipe I/O, so you may not want to use it. Does not have job control or vfork. Contact Bob Kridle (415) 642 6744.

Speaker 23: 12:23pm Walt Low, Berkeley

F77 update

Walt spoke about F77 for the VAX and the PDP-11/70. The November 1980 version is available with fork and exec calls and no overlays. The system interface library is complete as is the I/O library and maths library.

F77 on the VAX is very slow (about 10 lines/second) and produces slow code (about half the speed of VMS fortran). Additional optimisation of the first pass is in progress and the result should produce code that competes with VMS fortran.

------ LUNCH ------

USENIX Group Meeting
Thursday, June 25, 1981
UNIX languages and office automation
Chair: Mike O'Dell, LBL

Speaker 24: 1:50pm Chris Robertson, University of Toronto

Concurrent Euclid - a brief outline

Chris, who really has nothing to do with Euclid, did a good job of remote presentation. Euclid is summarised in the paper appearing later in this issue of AUUGN. Available September 1981, the price is $500, or less if you can prove you are destitute. For further information contact Chris at

Computer Systems Research Group,
Distribution Manager,
121 St Joseph St,
Toronto,
Ontario M5S 1A1, CANADA,

(416) 978 6060.

Speaker 25: 2:00pm Barbara Staudt, Moravian College

Compiler construction using YACC

Barbara wrote a parser for a subset of Pascal using YACC as a student project. The parser produces C code output. YACC was a pleasure to use because it resolved shift/reduce conflicts nicely, error recovery was relatively easy and tokens have values that were used to represent data types in implementing the parser.

Speaker 26: 2:05pm Robert Wells, BBN

PEN - the design of a flexible video editor

PEN is a portable video editor that supports underlining, vertical and horizontal windowing and provides good error messages. It is usable on a number of video terminals given terminal configuration (what character sequences do what) and allows keying of commands to any function key. PEN uses minimum screen manipulation and functions well, even at 1200 baud.

The editor is object oriented, making use of lists of objects etc. Some objects manipulated are files (STDIO style), strings, vectors (type of string), nodes (generic linked lists), ilists (generic indexable lists), buffers and windows.

It runs on I/D PDPs (takes 60KB on a 70), C70, VAX and others soon.

------ Coffee Break ------

Speaker 27: 2:53pm Edward Scott, Computer Consoles Inc.

Office power

In a talk that was fast, amusing and interesting but not particularly technical, Ed depicted office automation as expanding, helped by greater general acceptance of computers, rising white collar costs and inefficiency. The costs of technology are decreasing while the cost of people is rising.

Using a wonderful dual projector slide system (I was sitting next to it...) we saw Computer Consoles' idea of what the ideal office might be. The executive enters his office, desk cluttered with the paraphernalia of business plus a computer terminal. Gradually you see how everything in the office, bar the potted plants, may be replaced by a touch of the keyboard. His diary, notebook, calculator, telephone book, dictionary, forms, filing cabinet and telephone disappear before our eyes. Slowly the office empties, leaving our executive, the chair, table and terminal. I started to feel that the executive looked a bit redundant, and yes, sure enough he at last winked out of existence.

But, I think to myself, acknowledging that I do have sexist thoughts, will a terminal really replace the standard full bosomed secretary?

Speaker 28: 3:33pm Ken Omohundra, ABLE Computer

DEC UNIBUS interfacing

The UNIBUS is the nearest thing to a matched impedance bus, its asynchronous nature is nice, it's easy to configure and there are lots of devices made for it. Ken is a UNIBUS lover. But the UNIBUS has limited bandwidth due to it being a 16bit bus. So DEC tried to get round this by taking a PDP11/45 and turning it into a PDP11/70 with the 32bit wide datapath to memory. But on the 70 NPR latency becomes a real problem in I/O bound systems. So DEC went to the SBI on the VAX. Is this the answer? (Ken didn't answer that.)

Getting back to UNIBUSes, beware of salesmen who quote transfer rates or cycle time approaching 300-400nS. Address set up time is 150nS, slave sync is 75nS and address hold is 75nS totally 300nS just for bus book keeping. Your bus should (within reason) be configured in the order - CPU, memory, unbuffered DMA (like TM11s), single buffered DMA (like RK11s), silo buffered DMA (like RM11s), communications DMA (DH11s etc) and finally the rest.

When calculating bus loading eight to ten feet of bus length should be counted as one bus load. Keep cables as short as possible. When the bus load count gets to 17 or more, think about installing a repeater. Always adjust power supplies when you alter the bus configuration.

Why do some 'UNIBUS devices', particularly second source devices, sometimes not work on the newer machines? The PDP-11/34 was the first computer to use the UNIBUS as defined (in the DEC PDP11 bus handbook), eg. the PDP-11/40 did not drop the data lines until after master sync, and the PDP-11/34 does. Beware microprocessor driven devices, like disc controllers emulating DEC devices. They are slow, and sometimes the bits don't change quickly enough.

Ken agreed to hold a UNIBUS 'birds of a feather' session later this evening.

Speaker 29: 4:00pm John Mashey, Bell Labs.

Small is beautiful and other thoughts on programming strategies

The first slide labelled what followed as the fifth iteration of a popular talk John has given from time to time. John had lots of slides, talked very fast and about the only documentation I got was a list of references and a few quotes. The references appear later in this issue and some of the quotes below are attributable to the people in that list. I have no idea which ones.

There seem to be three approaches to a programming problem

- Do it right. Spend a long time planning, design well and implement the right thing first go.

- Do it over. Do it, and so learn what you really want to do, and then do it right the second time after the massive first version has failed massively.

- Do it small and do it with tools. Pick appropriate timing, emphasis and risk. Focus preplanning. Build something fast and be prepared to fail quickly.

And now to some quotes....

Failure is the norm.
Don't look back, something may be gaining on you.
Adding manpower to a late project only makes it later.
Long project + staff turnover = steady state zombie
Things don't get smaller, they get bigger.
Things don't go from bad to good, they go
from better to ok,
from ok to worse and
from worse to abysmal.
The black hole principle: The more you put in, the less you get out.
All features have users; Bugs <==> features.
Premature optimism is the root of all evil (Knuth).
We have met the enemy and they is us.

USENIX Group Meeting
Friday, June 26, 1981
Graphics and Data Base Systems
Chair: Wally Wedel, University of Texas

Speaker 30: 9:09am Adrian Freed, LERS

Towards Graphics Device Independence

Adrian described a library of C functions that implement primitive object drawing, character sets, windowing, scaling and scan conversion. Algorithms for raster and vector devices are included and the library has been used to implement one interface for three quite different devices.

The algorithms provide

General:	Scaling and translation (windowing), and clipping.
Raster Scan:	Line, circle, arc, character sets and raster scan converter.
Caligraphic:	Line, circle, arc and character sets.

A problem with the package is that it uses only 16bit arithmetic.

Speaker 31: 9:25am Roger Sippl, Relational Data Base Systems, Inc.

Commercial quality relational DBMS for UNIX

Roger described the MARATHON system which features the INFORMER query language, ENTER I data entry program, application language library, full data integrity for multiple users, audit trail backup and recovery and report writers.

There is a glossy later in this issue.

Speaker 32: 9:45am John Z. Kornatowski, Rhodnius Inc

Porting Mistress to the ONYX

I have published Mistress glossies in earlier issues. The porting effort encountered logistic problems (45%), problems with ONYX C (45%) and other (10%).

Logistics problems manifested as a lack of ONYX systems in Canada, slow delivery, limited availability of machines when they were delivered, no C compiler ordered with the machine, no documentation when C did arrive and no high speed I/O device on the ONYX (source transfered by serial line, 6hrs/MB).

The C problems started with a compiler labeled ´pre-release, use at your own risk´. There were three versions of C delivered called CC, PCC and ZCC none of which was well documented. They produced different (some incorrect) code and implemented different dialects of C. John finally used all three Cs.

There were some Mistress source problems such as ints declared as chars and some other C compiler dependencies, but it all works now. The Mistress ONYX and PDP code is now the same. To get Mistress write to John at

Rhodnius Inc,
Product Information,
PO Box 1, Station D,
Scarborough,
Ontario M1R 4Y7, Canada,

(412) 922 1743.

Speaker 33: 10:07am James Joyce, International Technical Seminars

Educating users: ITS approach

Jim demonstrated the ´tinker-toy´ teaching method he employees. Probably it would lose in the translation if I described this portion of the talk. Other educational hints are

- Visuals should be that; the less text the better and if text is there break it up.
- Exercise is the key to success. You can´t give enough examples and hands on teaching.
- Team teaching is useful. It is often useful to have a paid stooge in the audience to prompt with questions and act dumb.

------ Coffee Break ------

Speaker 34: 11:08am Spencer Thomas, University of Utah

Alpha 1: Computer aided geometric design using UNIX

Alpha-1 creates object representations using D-spline surfaces (a new way of looking B-spline surfaces), Baumgart (spelling?) topology description, data types and generators. Graphics facilities provided or planned include line drawing displays for interactive design, raster displays for high quality shaded images and numeric control of machinery. Pretty pictures.

Why UNIX? Already had it, already had C, homogeneous I/O, the shell is a good command language, MAKE, easy process control, changes relatively easy to make and other miscellaneous niceties.

Speaker 35: 11:32am Ernie Harkins, University of Colorado

Leroy: Publications quality plots

Ernie is from the Cooperating Institute for Research in Physical Sciences (CIRES) computing facility and is into 'making pretty pictures'. Leroy is written using YACC, has lots of fonts and is on the conference tape. The pictures were nice.

When he explained that a Tektronics 4012 lost information at high speed someone offered the information that filler characters could be used to get round the problem. Quote, "Filler characters? What a nauseating idea!"

Speaker 36: 11:50am Wally Wedel, University of Texas

Close

Wally thanked all the people who helped out with the conference, the speakers and USENIX board. There were twelve tapes submitted and he hopes that a consolidation of the contents will be distributed within three months.

Modifications to UNIX† to Allow Four Mega Bytes of Main Memory on a 11/40 Class Processor

Clement T. Cole

Computer Research Group
Computer Research Laboratory
Tektronix Laboratories

Sterling J. Huxley

Computer Resource Department
Technology Group

ABSTRACT

This paper describes the modifications to UNIX Version 7, to efficiently use the 4 Megabyte extented addressing capabilities of a 11/40 class processor after hardware modification. Discussed are the reasons why such changes are desirable, what parts of the machine are effected and a few comparisons of the results.

June 20, 1981

† UNIX is a Trademark of Bell Laboratories.

Modifications to UNIX† to Allow Four Mega Bytes of Main Memory on a 11/40 Class Processor

Clement T. Cole

Computer Research Group
Computer Research Laboratory
Tektronix Laboratories

Sterling J. Huxley

Computer Resource Department
Technology Group

Introduction

We will be describing the changes that were made to a standard UNIX system to allow the addition of extra main memory beyond that DEC normally allows. We will first be discussing the reasons for such radical departure from a normal system. Next we will discuss the changes made, and the effects that these non-standard hardware and software changes have to a user program. Last of all we will look at some performance statistics and see how they relate to the earlier justification for this work being started.

History

In 1972, Ritchie and Thompson released to the world the Fifth Edition of the UNIX Timesharing System[1]. This system was designed to run with minimal hardware. In 1978, the Seventh Edition of UNIX was released with the caveat that it must run on larger PDP/11 systems. The term "larger" was intended to mean DEC PDP 11's with a hardware kludge to allow a seventeenth address bit. This kludge is known as the separate instruction and data bit[1]. This bit separates instruction and data fetchs to main memory on the Unibus[2]. The term "minimal" included the 11/40 and 11/35 processors that do not support the separate I/D feature. In those 6 years and two releases of UNIX what happened?

For a first order view, UNIX was improved, at the expense of size. These improvements were assorted, from simple bug fixes, to modernize an original design decision that became inappropriate. The fundamental change was to correct for a hardware problem

† UNIX is a Trademark of Bell Laboratories.

1] Another term for this feature is: separate I and D that became acute over those same 6 years: disk technology improved drastically, forcing the earlier file block[2] to physical disk block mapping to be unusable.

The Seeking Problem

When UNIX Version 6 was released, UNIX was forced into viewing each large, DEC RP03 style, disk as many smaller pseudo disks. This feature was introduced because the fundamental data size on a PDP 11 is a 16 bit integer. An RP03 has more than 65536 physical blocks per disk. Later DEC introduced the RP06 disks that contains about 150 Megabyte of storage. Technology was bound to improve again, making the pseudo disk solution a little unbearable. Clearly, there are times when a user would wish to address every physical block on the disk.

Generic UNIX does supports a system call to allow a user program to address any byte in a file. The *seek* system call takes three parameters[3]. The descriptor of the file to seek on, the amount to seek, and a third parameter that specifies the type of seek, i.e. seek relative from the beginning, end or current file position. The UNIX operating system supports a data abstraction that allows a user program to view any device as a sequential file of bytes. Using the seek system call and this data abstraction, we can write programs to manipulate any byte on any disk[3]. Unfortunately, as we discussed before, the operating system had difficulty building the complete abstraction for large disks. These principles beget Version 7's new seek system call.

In Version 7, the seek system call was removed and replaced with a lseek (long seek) system call[4]. The difference between the two calls is that the parameter that specifies a seek count, was changed from a short integer (16 bits), to a long integer (32 bits).

The change from a short to a long had major ramifications. Not only did it nearly double the size of many of the already large tables that the operating system needed to keep in main memory, but in the operating system size was increased in tiny increments by arithmetics. What used to be a add instruction, now needs to be an inline expansion for 32 bit arithmetic. Not that these changes are terrible, but many of these expansions have the tendency to slowly add to the size of the kernel. See the appendix

2] A block is the fundamental unit of storage on a disk drive. It contains 512 bytes of 8 bits each.

3] An example of a program that would like to manipulate an entire disk at a time is a program that operates on entire file systems, like dump(1m), restor(1m), and icheck(1m).

for a demonstration of the code expansion problem.

Solutions to Address Space Limitations

The Version 7 UNIX Kernel (adding both instruction and data) would not fit into the 64k byte limit that a PDP 11/40 class processor imposes, and still be able to have a several runable processes in the kernel process tables. To be truthful, large UNIX Version 6 systems had much difficulty fitting into the 11/40's address constraints. Many different groups created ingenious schemes to win back kernel space. The most notable were the Calgary Buffer Modifications that moved the system I/O buffers out of the normal kernel address space and made the system change its memory management registers to address the data in those buffers. For small 11/40 installations these modifications were not completely necessary. However, if these modifications were not installed, UNIX would have to be "shoe horned" into fitting the address space available from the DEC 16 bit virtual addresses. When Version 7 came, these modifications became necessary for even the smallest 11's to be able to run more than five or six processes (not users) at the same time. Soon, another interesting method was suggested by people at UC Berkeley[5]. They suggest that the kernel be "overlayed." This overlay scheme is not a classical disk overlay, but for efficiency, an in memory overlay. Meaning that the whole kernel would be kept in main memory at one time, and the memory management registers changed as needed.

The "in-memory" overlay method, when it is used with the Calgary buffer modifications, seems to be among the best methods of gaining back kernel address space to date. Unfortunately, these modifications have one unhappy side effect. They trade off kernel size at the expense of available user memory. A PDP-11/40 class processor's memory management unit can only address 18 bits (256 K bytes) of main memory. It is easy for a UNIX kernel to become greater than 1/2 of that size and in some cases be closer to 3/4's of our 256 K byte limit. Remember that the reason for letting the size of the kernel become large in the first place was to allow more that a few processes to try to run at the same time. We now have enough kernel buffers, but not enough main memory; and unhappy Catch 22. To this end, the software people asked the hardware people for a little help.

Able Computer Corp. produced exactly what was needed. They have a new memory management board called: ENABLE. This board works in conjunction with the DEC memory management unit to produce 22 bit's worth of address on a machine that previously had 18 bit's worth. Making the theoretical size, of the address space of this new hybrid 11/40 class processor[4], 4 megabytes.

The ENABLE board plugs into a new backplane and creates a new 22 bit Modified Unibus[6]. It works *without* other modifications to the hardware.

With the use of this new hardware UNIX now can address the 22 bit's of address space needed. By overlaying the UNIX kernel, and for that matter, any user program, it is capable of growing extremely large. The changes to the UNIX kernel to support the ENABLE board have *no* affect on user programs.

The Able ENABLE Board

When a program runs on a unmapped PDP-11, the processor will produce a 16 bit address. With mapping turned on, this 16 bit address is referred to as a **virtual** address. To produce a 22 bit **physical** address there are two relocations that a virtual address must go through. The DEC relocation hardware will produce an 18 bit address. The ENABLE relocation will transform the DEC 18 bit address into a 22 bit physical address.

1. DEC relocation

A 16 bit DEC virtual address (VA), produced by the PDP 11 cpu, builds a 18 bit DEC physical address (PA) as follows:

> DEC VA bits 13-15 (three bits) are used to index one of 8 DEC Page Address Registers or PAR's. The lower 12 bits of the indexed DEC PAR are added to the DEC VA bits 6-12 (seven bits) to produce a 12 bit field which goes into bits 6-17 of the DEC PA. Bits 0-5 of the DEC VA are concatenated with the result to produce DEC PA bits 0-5.
>
> The 18 bit DEC PA is now gated to a normal 18 bit Unibus.

2. ENABLE Relocation

The ENABLE relocation process is analogous to the DEC relocation.

> The ENABLE PAR's are indexed by bits 13-17 of the DEC PA[5].

4] The name of the macro that we have used in code found in the appendix for this new hybrid is FORTYZ. We will have the standalone system set the global variable "cputype" to 1 greater than the normal. e.g. 41 for an 11/40 class processor and 46 for an 11/45 class processor.

5] 5 index bits means that 32 PAR's can be accessed but only 16 can be used for an 11/40 installation. The other 16 PARs can be used by an 11/45 class processor. For the 11/40 we are using 8 for kernel and 8 for user

The contents of the indexed ENABLE PAR are added to bits 6-12 of the DEC PA to yield a 16 bit field which is placed in bits 6-21 of the ENABLE PA. Bits 0-5 of the DEC PA are again concatenated with the result to produce ENABLE PA bits 0-5. Note that is means that the original DEC VA Bits 0-5 are passed unchanged to ENABLE PA bits 0-5.

This 22 bit address is now gated on to a special 22 bit MODIFIED Unibus. This new Modified Unibus is electrically the same as the one found in an 11/44.

Setting up the Two Memory Management Units.

In order to achieve 22 bit relocation, the DEC Memory Management Unit must be effectively NOP-ed. This is done by placing data into the DEC PARs so that when the DEC 18 bit translation is operating, the 18 bit result will index into the ENABLE PARs *without* adding any other relocation. Because we know that the ENABLE board uses the upper 5 bits of the DEC 18 bit PA as an index, we should only set these bits in the DEC PARs. Contained inside a short routine in mch40.s, we will once and only once, set the contents of the DEC PARs. After that time, our new UNIX will use the ENABLE PAR's like the older versions *had* used the DEC PAR's.

3. Setting up DEC Page Address Registers

The DEC Kernel PAR's (DKP) should be initialized as follows:

DKP0 = 0000
DKP1 = 0200
DKP2 = 0400
DKP3 = 0600
DKP4 = 1000
DKP5 = 1200
DKP6 = 1400
DKP7 = 7600

Notice that the lower 7 bits of each number are 0. Bits 7-10 of the DEC PAR is used as an index into the ENABLE PAR's.

To explain why 7600 is placed into DKP7, we must remember that there exists a short period of time when DEC memory management is turned on and ENABLE memory management turn off. During this window, there are references made to the I/O page when we activate the ENABLE Board. Meaning, the DEC memory

---

operations. For an 11/45, we would have 8 for kernel I space, 8 for kernel D space, 8 for user I space and 8 for kernel D space.

management must be set up to handle references to the I/O page with and without the ENABLE board active. DKP7 being 7600 means that when ENABLE memory management is on all references to the I/O page go through ENABLE PAR 31 (DKP7 bits 7-11 = 31). The DEC kernel PAR's index into ENABLE PAR's 0-6 and 31. Note therefore that we are using 7 ENABLE PAR's and then skipping the next 24 ENABLE PAR's.

DEC user PAR's are initialized with 2000 and increment by 200. Therefore, DEC user PAR's index into ENABLE PAR's 8-15.

DUP0 = 2000
DUP1 = 2200
DUP2 = 2400
DUP3 = 2600
DUP4 = 3000
DUP5 = 3200
DUP6 = 3400
DUP7 = 3600

Once the DEC PAR's are initialized it is not necessary to change them[6]. Thereafter when switching user processes only the ENABLE USER PAR's will be modified. Also, A version of UNIX can be used that allows more than 64k bytes of main store to be used by a program while only manipulating the ENABLE Kernel PAR's.

A Minor Mishap with I/O Mapping

The Unibus was designed to support 18 bit DMA I/O devices[2]. Therefore on the 22 bit PDP-11's (11/70 class), an I/O map, called Unibus Map, must also be provided to translate from 18 bit addresses produced by the DMA hardware to the 22 bit addresses that the memory system sees. It should be noted, that all DMA hardware accesses memory at different times and through different drivers than the cpu uses for an instruction or data fetch. Which means on an 11/40 and 11/45 class processor, the I/O subsystem runs unmapped, producing physical address in main memory. Able exploits the fact that both the 11/45 and 11/40 class processors do not need a Unibus Map. Fortunately, DEC did not assign any other I/O devices residing at the I/O map's location for these processors. Allowing our version of UNIX to initialize this map, on our modified machines, in the same manner

6] The only time the kernel must modify these while running normally, is when you wish to support the fuiword subroutines used by the unsupported phyio(2) system call. This is because this subroutine is using the *mfpi* instruction, which makes some rash assumptions about the state of the DEC memory management registers.

as unmodified UNIX does on the 11/70 class machines.

Unfortunately, it was found that the ENABLE board cannot be running in 22 bit relocation mode with I/O mapping turned off. Before enabling the ENABLE board both the I/O mapping and 22 bit relocation bits in the ENABLE SSR3 must have been set.

UNIX would like to set up the I/O mapping registers *after* the kernel is loaded and running in 22 bit mode. Since both I/O mapping and 22 bit relocation bits must be set at the same time the I/O map is enabled before the map is initialized! UNIX is now running with an undefined I/O map.

These two bits should be independent of each other. The system could then boot and run in 22 bit mode with the I/O map turned off. The initialization code for the I/O map could be left in machdep.c instead of having to move it into the assembler assist that sets up the memory management unit.

Some Results

The authors ran a simple bench mark on four UNIX configurations. These are: an unload 11/70 with 2 megabytes of main memory and 2 RP06s; an unload 11/44 with 1/4 megabytes of main memory and 1 RM02; an 11/34 with 1/4 megabytes of main memory and 2 RL01s; and the same 11/34 with 1/2 a megabytes of main memory and the new ENABLE unit installed. We understand that you must mellow these statistics due to the fact that were running a UCB kernel on the 11/70 and 11/44. We were running a DEC kernel on the 11/34[7]. The actual bench mark was to recompile the entire UNIX kernel from zero. What was typed into UNIX:

```
$ rm dev/*.o sys/*.o
$ time sh -x makeit
```

The file "makeit" contained the lines:

7] The plan was to show you the differences after we had brought up the UCB overlay kernel on the 34. Unfortunately, we were using borrowed memory from Mostek Corporation for the extra 1/4 mega on the 34. We had to give the memory back before we received the overlay code from UCB for the 11/34. We also were using a borrowed 11/34 from Tektronix Computer Automation Support. We had to give this machine back after one week. Once we get some system time again we will running the correct bench mark, and republish these results.

```
cd dev
make
cd ../sys
make
cd ../conf
make unix
```

Some Numbers

CPU Type	Real	User	System
11/70	13:06.0	5:02.9	3:12.3
11/44	23:35:0	7:42:0	6:38:7
11/34	58:23:1	17:12:3	13:06:2
11/34 with ENABLE	42:19:2	14:41:8	10:29:2

The 11/34 should come closer to the 11/44 performance once the UCB changes have been added. Note that the extra memory help the 11/34 because it did not have to swap to the extremely slow RL01s. When the Calgary mods are added the RL's will look much better, because this kernel will be allowed to have 50 or 60 buffers.

For a matter of reference, we used the DEC V7 kernel for the base 11/34 installation. This was because we felt that the people at DEC would have a better idea of how to tune the an unmodified 34 system than we might. It also gave us a reference base. To obtain their code write to:

The DEC UNIX Group
DEC, Continental Blvd
MK1-1/D29
Merrimack, NH 03054

I believe there is a charge for this tape, but check with DEC first.

In the case of the 11/44 and 11/70 we are running a Modified UCB Kernel, from the San Fransico, prelease 2BSD system. We have added a new TTY driver. Hopefully this will come in sync with the UCB one. To obtain this tape, write to:

Bob Kridle
EECS Dept.
Cory Hall
UC Berkeley
Berkeley, CA 94720

Acknowledgments

The authors would like to acknowledge some groups for helping with this effort. Armando Stettner at DEC for the 11/34 system; Mark Horton and Bob Kridle at UCB for help the 11/70 system; Larry Brown and his crew at Tek's Computer Automation Support for the use of an 11/34 and helping us when our hardware died; Mary Driscoll at Mostek Corp, for lending us the 22 Bit memory; and Ken O'Mohundro and Norm Kiefer at Able Corp for the use of a pre-release ENABLE board.

References

1.) Ritchie, D. and Thompson, K., the UNIX Timesharing system., Western Electric Corp.

2.) DEC Processor handbook, Digital Equipment Corp.

3.) UNIX Programmer's Manual, Sixth Edition, Western Electric Corp.

4.) UNIX Programmer's Manual, Seventh Edition, Volume 1 & 2, Western Electric Corp.

5.) Haley, C. and Joy, W., revised by Jolitz, W., Running Large Text Processes on Small UNIX Systems, UC Berkeley Technical Report. Spring 1980

6.) ENABLE user's guide, Able Computer Corp. Pre-release Version, March 1981

Appendix A

Demonstration of Code Expansion - code.c

```
/*
 *      a demonstration of the add problem
 */

short   glob_short;     /* This is short in global space */
short   short_glob;     /* This is short in global space */

long    glob_long;      /* This is long in global space */
long    long_glob;      /* This is long in global space */

main ()
{

short   local_short;    /* This is short in local space */
short   short_local;    /* This is short in local space */

long    local_long;     /* This is long in local space */
long    long_local;     /* This is long in local space */

glob_short = 0;
short_glob = glob_short + 10;

local_short = 0;
short_local = local_short + 10;

glob_long = 0;
long_glob = glob_long + 10L;

local_long = 0;
long_local = local_long + 10L;

}
```

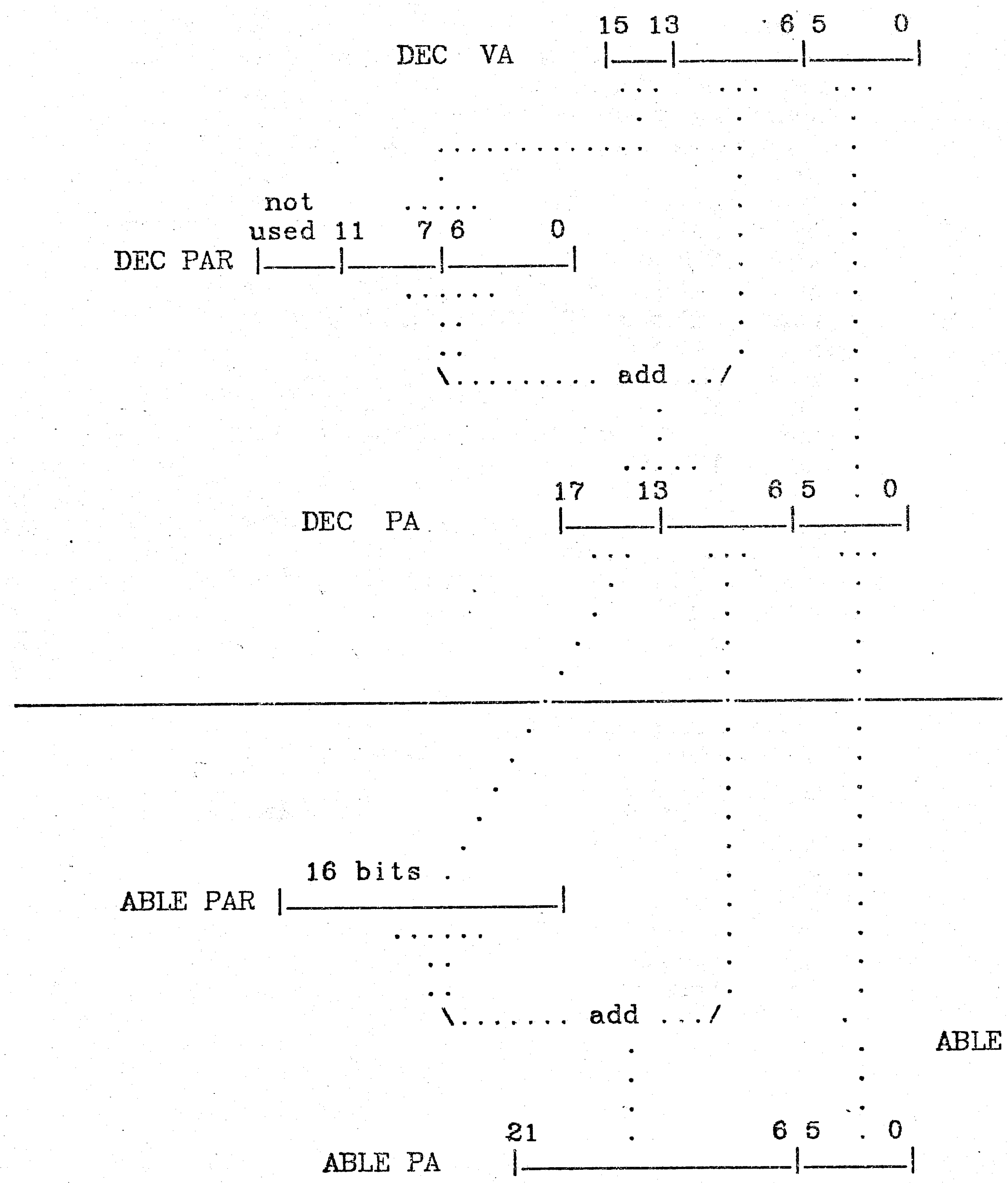
Appendix B

Demonstration of Code Expansion - code.s

```
        .data
        .comm  _glob_sh,2.
        .comm  _short_g,2.
        .comm  _glob_lo,4.
        .comm  _long_gl,4.
        .text
        .globl _main
_main:  jsr    r5,csv
        sub    $12.,sp
        clr    _glob_sh          / line 20, file "code.c"
        mov    _glob_sh,r0       / line 21, file "code.c"
        add    $10.,r0
        mov    r0,_short_g
        clr    -8.(r5)           / line 23, file "code.c"
        mov    -8.(r5),r0        / line 24, file "code.c"
        add    $10.,r0
        mov    r0,-10.(r5)
        clr    _glob_lo          / line 26, file "code.c"
        clr    _glob_lo+2.
        mov    _glob_lo+2.,r1    / line 27, file "code.c"
        mov    _glob_lo,r0
        add    $10.,r1
        adc    r0
        mov    r0,_long_gl
        mov    r1,_long_gl+2.
        clr    -14.(r5)          / line 29, file "code.c"
        clr    -12.(r5)
        mov    -12.(r5),r1       / line 30, file "code.c"
        mov    -14.(r5),r0
        add    $10.,r1
        adc    r0
        mov    r0,-18.(r5)
        mov    r1,-16.(r5)
        jmp    cret
```

Appendix C

Virtual to Physical Address Mapping



Appendix D contains UNIX

source code — send me

a copy of your licence should

you want a copy

Peter I
17/7/81

DECUS Miami June 81

CONCURRENT EUCLID: COMPARISON WITH C AND PASCAL.

J.R. Cordy, R.C. Holt
B. Spinney, C. Lewis
Computer Systems Research Group
University of Toronto
Toronto, Ontario, Canada

ABSTRACT

It is generally recognized that it is desirable to do system programming in a high-level language. The C language, coming out of the UNIX (Western Electric trademark) operating system is attrative because it yields efficient machine code. The Pascal language is attractive because it provides relatively elegant structuring combined with strong type checking. By comparison with these two languages, Concurrent Euclid offers their combined advantages, including both efficient code and strong typing. In addition, it provides controlled access to machine-dependent features, separate compilation and language-specified concurrency. Portable Concurrent Euclid compilers are under construction with replaceable code generators for various computers including PDP-11, VAX, MC68000 and MC6809.

INTRODUCTION

Concurrent Euclid was designed with these goals: convenient features for developing compiler/systems software, small fast compiler, portable selfcompiling compiler, and high-quality code.

Concurrent Euclid thus has goals similar to those of the C Language. It diverges from C primarily in enforcing a programming discipline that improves reliability and decreases maintenance costs. For example, strict visibility rules minimize interaction among modules. Generally, code generated by the Concurrent Euclid compiler is at least as good as that of C, and considerably better for procedure calling.

Concurrent Euclid is a direct descendant of the structured language Pascal. Concurrent Euclid has been made cleaner than Pascal by clarifying type compatibility, eliminating aliasing and function side effects, providing user-controlled visibility, etc. Also provided are language features needed for systems programming, such as absolute address variables, type converters, separate compilation, and concurrent processes. Like some implementations of Pascal, optional runtime code checks subscripts and case selectors, and maintains the source line number in a register. The result is that it is seldom necessary to take a dump to do debugging.

Pluggable code generators already exist for Concurrent Euclid for the MC68000, MC6809, and the PDP-11. These generators are attached to a machine independent compiler front end (Spinney).

EVOLUTION OF CONCURRENT EUCLID

The Euclid programming language is a modification of Pascal designed for writing systems programs that can be verified using state-of-the-art techniques. The important features added to Pascal include: explicit control of visibility of names, e.g., explicit importing and exporting of identifiers from modules; partitioning pointers into distinct collections: extending types to have parameters so that a type declaration can be a template for many different instance types: module types to provide data abstraction and information hiding; generalization of Pascal constants to allow their values to be calculated at run-time: machine-dependent features for immediate access to the underlying hardware; and assert statements.

The Euclid language is described in the Euclid Report (Lampson). The language design was commissioned by the Defense Advanced Research Projects Agency of the U.S. Department of Defense in 1976.

The language was to be based on Pascal and designed to facilitate verification of programs using formal proof techniques.

An early attempt at implementation of Euclid was undertaken at the Systems Development Corporation early in 1977 but was never completed.

The first successful implementation of Euclid, called Toronto Euclid, was carried out as a joint effort of the Computer Systems Research Group at the University of Toronto and the Special Systems Division of I.P. Sharp Associates Ltd. (Wortman).

A prototype of a UNIX-compatible operating system (called TUNIS) was implemented in Toronto Euclid. Based on this experience with the Euclid language, the language Concurrent Euclid was designed. Concurrent Euclid omits difficult to compile features, notably parameterized types, and adds features needed for systems programming, notably concurrent processes. Concurrent Euclid is "nearly" a compatible subset of Euclid. It is self-compiling and was originally compiled by the Toronto Euclid compiler. The Toronto Concurrent Euclid compiler is fast (it compiles a pass of itself in about 10 minutes) compared to the Toronto Euclid compiler. A new version of TUNIS is presently being developed in Concurrent Euclid.

COMPARISON WITH C

The C language has been highly successful as a production systems implementation language. It is notable for providing good quality machine code. Most software in UNIX systems is written in C. C and Concurrent Euclid have similar goals in that both are intended to support systems programming. Both have portable compilers, allowing them to be used for a variety of machines. The most striking differences between the two are (1) C is a much more primitive, Fortran-like language and (2) C's data structuring is low-level and insecure. Where Concurrent Euclid has strongly typed records and pointers partitioned into collections, C provides "structures" without restrictions or use of the dot operators and pointers that are treated implicitly as integers. Where Concurrent Euclid optionally checks subscript bounds, C has no linguistic notion of a subscript being bounded (!).

Where Concurrent Euclid provides "modules for data encapsulation, C supports globally accessible "external" data in a way analogous to features of PL/I or Fortran.

These differences lead inevitably to much easier testing and less costly maintenance in Concurrent Euclid, because so many programming errors are caught at compile time or in initial testing. By way of comparison, in Concurrent Euclid, a wild subscript is immediately isolated, while in C, this situation leads to remote data or program corruption and laborious inspection of core dumps.

The following example from a C program illustrates the lack of checking implied by C's low-level features.

C Statement	Purpose	Result
j=-1;	Set j to -1	Set j to j-1
if(j=i)	If j equals i	Set j to i; see if j is 0
{printf(j);	Print j	Print garbage or crash
skipline;}	Skip a line	Do nothing

Each line seems reasonable and might be expected to achieve its purpose. Each could easily be produced through oversight, ignorance or clerical error. Each is legal in C. The unfortunate results come about because (1) the operator =- is ambiguous, (2) equality is written as == but = is a legal infix assignment operator, (3) the type and number of procedure parameters is not checked and (4) a procedure name, skipline in this case, without following parentheses is a value equal the procedure's address.

This example does not illustrate the most common and insidious kind of C bug, namely, pointers gone wild in innocent looking expressions. By comparison, disasters such as these are caught at compile-time in a Concurrent Euclid program.

One might guess that since C is a lower level language than Concurrent Euclid, code for C programs would be more efficient. In fact, algorithms compiled by the Concurrent Euclid compiler are typically about 10% smaller and faster than the same algorithms in C. For this comparison, we have assumed that Concurrent Euclid omits (1) subscript checks and (2) code to put the line number in a register. These two options each cost about 15% in time and space when used and are unfortunately not available to the C programmer. We have assumed that C's optional optimizer pass has been run. (Concurrent Euclid has no such pass). Our comparisons have been for PDP-11 machine code.

Concurrent Euclid's superior efficiency derives primarily from the excellent code it produces for procedure calling. For example, the "do nothing" procedure executes only one instruction in Concurrent Euclid, but about 16 instructions in C. In standard tests of procedure efficiency such as (Wichman), Concurrent Euclid typically beats C by from 50% to several hundred per cent. Since systems programs usually consist of large numbers of small procedures, this efficiency pays off.

To summarize our comparison of C and Concurrent Euclid, C is a lower-level, less secure language and Concurrent Euclid provides more efficient code.

COMPARISON WITH PASCAL

The Pascal language has proven extremely successful. It is widely used, especially on microprocessors, and has influenced many systems programming languages. It encourages structured, reliable programming via its relatively well designed data types and control constructs.

Pascal does not provide various features needed for systems programming, such as separate compilation, variables at absolute addresses and concurrent processes. These features are available in Concurrent Euclid.

Most existing Pascal compilers do not produce code that is as efficient as that of C. This lack of efficiency has prevented Pascal from being used in many time or space critical applications. Concurrent Euclid provides the elegance and reliability of Pascal without sacrificing either efficiency or features needed for systems programming.

Ten years ago Pascal was a considerable step forward in language design. However, it contains various trouble spots and omissions that are corrected in Concurrent Euclid. We will give two examples.

Most importantly, Euclid provides data encapsulation which is missing in Pascal. In the following simple example the abstract object Stack has its implementation hidden from its users. Outside the module, only the entries Push, Pop and Value can be accessed. A module such as Stack can be tuned, modified and maintained without fear that another program depends on Stack's internal structure. Conversely, Stack can access T and size, because it imports them, but can access no exterior variables.

```
{Example of a module     }
{that implements a stack}


type T =...

const size:=10

var Stack:
  module
     imports (T,size)
     exports (Push, Pop, Value)
     var storage: array 1..size of T
     var top: 0..size
     procedure Push (v:T)=
        imports (var top, var storage)
        pre (top < size)
        begin
           top:=top+1
           storage(top):=v
        end push
     :
     initially
        imports(var top)
        begin
           top:=0
        end
  end module
```

Pascal provides a "while" loop which exits at its top and an "until" loop that exits at its bottom. However, a loop cannot be exited from its middle (except by resorting to the famous goto statement). By comparison, the loop construct in Concurrent Euclid provides an explicit exit which can be placed as needed, for example:

```
loop
  ...
  exit when x>y
  ...
  exit when x+y>z
  ...
end loop
```

The exit is put at the top when a "while" loop, and in the middle as needed. Thus the loop of Concurrent Euclid is both more general and simpler than corresponding loops in Pascal.

To summarize our comparison of Pascal and Concurrent Euclid, Pascal generally provides less efficient code while Concurrent Euclid contains features that are more modern and that are needed for systems programming.

CONCURRENCY

Operating systems control asynchronous devices and multiple users. The elegant and efficient way to handle these interactions is via concurrent algorithms. These algorithms are relatively easy to write and maintain if the programming language includes explicit features to support concurrency. Concurrent Euclid provides these features by means of concurrent processes and monitors. Monitors can be thought of as modules in which at most one process is active at a time.

Neither Pascal nor C provides comparable features, so the systems programmer using these languages is forced to result to various unsavory tricks to gain the effect of these features.

OBSERVATIONS

Concurrent Euclid evolved as a language for highly reliable systems programming. It has borrowed and improved upon the elegant features of Pascal. While doing this it has maintained an efficiency that is superior to proven lower-level systems languages such as C. Clearly C is superior to assembly language. We believe that Concurrent Euclid is clearly superior to languages like C.

BIBLIOGRAPHY

(Cordy) J.R. Cordy and R.C. Holt. "Specification of Concurrent Euclid", Report CSRG-115, Computer Systems Research Group, University of Toronto.

(Lampson) B.W. Lampson, J.J. Horning, J.G. Mitchell, G.J. Popek, Report on the Programming Language Euclid, SIGPLAN Notices 12, 1 (1977).

(Spinney) B.A. Spinney, "A Technique for the Construction of Portable Production Quality Code Generators", M.Sc. thesis, Department of Computer Science, University of Toronto, 1981.

(Wortman) D.B. Wortman, R.C. Holt, J.R. Cordy, D.R. Crowe, I.H. Griggs. "Euclid: a Language for Producing Quality Software", Proceedings of National Computer Conference, Chicago, May 1981.

(Wichman) B.A. Wichman. "How to Call Procedures, or Second Thoughts on Ackerman's Function", Software Practice and Experience, Vol.7, p.319-329 (1977).

Second Distribution of Berkeley PDP-11 Software for UNIX†
(revised January 1981)

A package of software for UNIX is available from the Computer Science Division of the University of California at Berkeley. This package includes the instructional Pascal system, the editor *ex*, an improved Unix version 7 kernel, the INGRES database management system, and other software (some of which is described below). All user software except the editor will run without separate I/D space, and on both version 6 and 7 systems.

Source code, binaries and machine readabln versions of all documentation are included with the tape, which is a standard *tar* format, 800BPI magnetic tape. We will supply a magnetic tape on which the software will be written.* If you do not have tar, a version 6 PDP-11 binary is on the tape and directions are given on how to extract it. The normal format for the tape is 800BPI, blocked by a factor of 20 (10240 byte records). Thus you must have a working raw tape drive (/dev/rmt0) to be able to read the tape. *Tp* and *cpio* formats are **not** available. 1600BPI tapes are available upon special request.

Binary only tapes, for holders of Unix binary licenses, are available for the same charge, containing PDP-11 v6 and v7 binaries, on an expressly *as-is, unsupported* basis. Binary tapes are *not* recommended, since holders of binary licenses are often on machines other than the PDP-11, or on PDP-11 without separate I/D (which is required to use *ex/vi*) and because the inevitable problems which require recompilation cannot be dealt with. The sources to this distribution cannot be distributed without a Unix source license because the editor *ex* and the shell *csh*, along with several other smaller programs, are based on version 6 Unix programs such as *ed* and *sh*.

To receive the tape fill out the attached agreement and send it with

1. A check for $150 US‡ made out to
   The Regents of the University of California
2. A copy of your UNIX license agreement.#
3. A clear indication of your mailing address.
4. A filled in copy of the agreement (which follows).

to

Berkeley Software Distribution for UNIX
c/o Professor Susan L. Graham
Computer Science Division, Department of EECS
Evans Hall
University of California, Berkeley
Berkeley, California 94720

---

†UNIX is a trademark of Bell Laboratories.
*RK cartridge distribution of the software is **not** available.
‡Purchase orders will be accepted for a surcharge of $50 US to cover the additional processing involved. Please send a check if at all possible.
#If you have a version 6 UNIX license and wish to receive the **−me** macro package and the modified version of the standard I/O library, then you must have (and include a copy of) a license for the version 7 phototypesetter package.

Included with the tape are copies of all documentation for the programs on the tape. If you wish additional copies of the documentation, send $15 for each additional copy. (You can also get just the documentation; in this case you need only send $15 and a copy of your UNIX license). Questions about this tape can be directed to Mark Horton at the address above or at (415) 642-4948. Messages can be left at (415) 642-1024. Arpanet mail can be sent to **2bsd@berkeley**, and *uucp* mail can be sent to **ucbvax!2bsd**.

The tape also contains more recent versions of the csh, ex/vi, Mail, Pascal systems, and a few smaller programs, all intended for version 7. It is expected that these latest versions can also be run on version 6 with little or no work.

This distribution is intended for the PDP-11. It will also probably run on other processors such as the VAX but there is a different (fourth) distribution which is more appropriate for the VAX. Inquiries about the fourth distribution should be directed to Laura Tong at the address above.

The second distribution is continually being updated as new versions of programs appear. It is aimed at vanilla version 6 and 7 PDP-11 systems. Programs included do not require the operating system enhancements made at Berkeley.

New Berkeley Software

In addition to the Pascal system, the distribution tape includes programs such as the following.

The "ex" editor

Two different versions of ex are included: versions 2 (currently 2.13) and 3 (currently 3.6). Version 2 will (barely) fit on a PDP-11 with separate I/D and will run on a standard version 6 or version 7 system. Version 3 (the current production editor) is larger and requires either a VAX or the "user overlay" feature in the supplied kernel.

This version improves the terminal driving capabilities of the editor to provide screen editing facilities using intelligent terminals. All terminal capabilities are described in a file so that the editor can drive new terminals after their descriptions are edited into the file. Padding and other requirements of intelligent terminals are dealt with in the editor. The editor is particularly designed so as to be usable as a screen editor on low speed dialup lines with intelligent and unintelligent terminals. The size of the editing window can be made small, and will expand as you work. This prevents long delays, e.g. after a search when the window will be redrawn in its smaller size.

The screen editing facilities of the editor have been extended so that it is possible to perform editing completely within "visual" or "open" modes. Open mode has been extended so that it works on hardcopy terminals.

Other new facilities include:

— Parameterless keystroke macros and word abbreviation. This facility allows a user who is repeatedly typing the same long sequence of commands or text to define a keystroke or word abbreviation which expands into the long sequence. Keystrokes can be any character or function key. This facility is used to cause multi-character arrow keys to work. (The macro facility is available only in version 3.)

— The capability to filter parts of the editor buffer through specified commands, and to read/write parts of the buffer from/to command.

— Text registers into which lines may be placed and carried over when editing new files.

— A "tags" facility for editing large programs which are broken into many files. The editor can be told the name of a tag (usually a function name) and will then switch to the file where that function resides and position itself at that function.

While "ex" edits only one file at a time, it remembers the line position in the previous file, so that it is possible to easily switch back and forth between two files.

An improved Kernel

The source for a version of the Unix version 7 operating system is included on the tape.* This kernel is also expected to be available on the San Francisco Usenix tape.

It contains a number of improvements over the Bell v7 kernel. A number of performance improvements (such as hashing buffers, moving buffers and clists out of kernal space, and the 1K block filesystem) have been made. The kernel supports kernel overlays, allowing it to run on non-separate I/D machines such as the PDP-11/23. It also supports user overlays, so that ex version 3 can be run, even on non-separate machines!

The Berkeley tty driver is included, which correctly handles erase and kill characters on crt and printing terminals, including correctly backspacing over tabs and control characters. As of this printing, it is expected that the process control features of the Fourth Berkeley Software Distribution will be included.

Changes to the kernel are conditionally compiled with mnemonic names, making it convenient to find the necessary changes to the kernel to implement any given feature in your own kernel, and making it easy to turn on and off features you decide you do or do not want. This kernel was contributed to by, and includes, or will include, code from, the Fourth Berkeley Distribution, the U.S.G.S. system, and Tektronix.

The Relational Database management system "INGRES"

INGRES is a database management system presenting data as a collection of tables ("relations"). Facilities include the usual *append*, *delete*, *replace*, and *retrieve*, as well as a variety of database utilities including bulk load and store, dynamic storage structure reconfiguration, full crash recovery, data integrity controls, views, and macros.

INGRES requires separated I?D space and floating point hardware. The version on the tape is aimed at Version 7 PDP-11 systems, although it is expected that a few compiltion flags could be changed allowing it to work on Version 6 systems.

A shell "csh"

This shell incorporates good features of the earlier shells, previous Berkeley shells and the version 7 shell. It also has a history mechanism (similar to that of INTERLISP) so that previous commands can be repeated and/or corrected.

*In order to receive the kernel you must have a version 7 license.

An error message understanding program "error"

This program looks at the output from a program compilation generated by *make* or a compiler, inserts the error messages as comments into the program or programs affected at the proper point in the program, and invokes the editor on those files, positioned at the first error. It understands the error messages generated by the Ritchie and Portable C compilers, lint, make, Berkeley Pascal, the vax assembler, the loader, and f77. The program obviates the need for writing errors down on scraps of paper before entering the editor. Several useful interactive options exist.

A macro package for nroff/troff: –me

This macro package is easy to use and especially easy to adapt to different formats. It works both with nroff and troff and provides a large number of hooks for special requirements. A beginners introduction and a reference manual are also available.†

A "Mail" program

This mail program uses "mail" to do the actual mailing and concentrates on providing a simple and friendly environment for processing large amounts of mail.

The Berkeley Network

The Berknet provides a facility for file transfer between machines, network mail, and remote execution of commands, in a batch request fashion. It is used at Berkeley to connect a network of UNIX systems. It can be run using terminal ports to connect machines.

---

†If you have a version 6 UNIX license and wish to get the -me macro package and the "stdio" modifications, you must include a copy of your license for the version 7 phototypesetter package;

Documents Supplied with the Tape:

Vi Quick Reference Card
An Introduction to Display Editing With Vi
Edit: A Tutorial
Ex Reference Manual
Ex Differences: Version 1.1 To 2.0
Ex/Edit Command Summary
An Introduction to the C Shell
Mail Reference Manual
Writing Papers with Nroff Using –me
–Me Reference Manual
Berkeley Pascal User's Manual
An Introduction to The Berkeley Network
Network System Manual
Screen Updating and Cursor Movement Optimization: A Library Package
INGRES version 6.3 Reference Manual
A Tutorial on Ingres
Creating and Maintaining a Database Using Ingres
Setting up Unix – Berkeley/USGS Version 7
New USGS/UCB System Details

Manual sections for all programs on the tape are also provided.

LICENSE AGREEMENT

THIS LICENSE AGREEMENT is made and entered into this __________ day of ____________ , 19___ , by and between THE REGENTS OF THE UNIVERSITY OF CALIFORNIA, a California corporation, hereinafter called "LICENSOR", and ____________________________ , a ____________________ having its principal office at ____________________________________ , hereinafter called "LICENSEE";

WITNESSETH:

WHEREAS, WESTERN ELECTRIC COMPANY, INCORPORATED ("WESTERN") is the proprietor and owner of the UNIX Time Sharing System, the UNIX Time Sharing System, Seventh Edition, or the PWB/UNIX Time Sharing Operating System, Edition 1.0 ("UNIX"); and

WHEREAS, LICENSOR is the proprietor and owner of enhancements and additions to UNIX, which together with UNIX comprise computer programs and documentation entitled, "Second Berkeley Software Distribution ("Licensed Material")"; and

WHEREAS, LICENSOR has been authorized by WESTERN to distribute the Licensed Material to licensees who are also WESTERN's licensees for UNIX; and

WHEREAS, LICENSEE warrants and represents that it is a WESTERN licensee for UNIX; and

WHEREAS, LICENSEE desires to obtain from LICENSOR, and LICENSOR desires to grant to LICENSEE, a license to use the aforementioned computer programs and documentation;

NOW, THEREFORE, in consideration of the mutual covenants, conditions and terms hereinafter set forth, and for other good and valuable consideration, LICENSOR hereby leases to LICENSEE the Licensed Material, the components of which are described on annexed Schedule A, subject to a non-transferable, nonexclusive license ("License"), which is hereby granted to LICENSEE, to use such Licensed Material upon the terms and conditions hereinafter set forth; and LICENSEE hereby accepts such lease subject to the License solely upon such terms and conditions.

1. **Term.** The term of this Agreement shall commence on the date hereof, and, unless sooner terminated as hereinafter set forth, shall extend indefinitely.

2. Fees. The fee for this license and one physical copy of the Licensed Material is one hundred fifty dollars ($150.00). LICENSEE may obtain additional copies of the Licensed Material and copies of new releases of the Licensed Material which the LICENSOR may make available by paying a duplication fee of one hundred fifty dollars ($150.00) per copy. All such copies shall be subject to the terms and conditions of this agreement. All fees are due and payable in advance.

Such fee does not include local, state or federal taxes, and LICENSEE hereby agrees to pay all such taxes as may be imposed upon LICENSEE or LICENSOR with respect to the ownership, leasing, licensing, rental, sale, purchase, possession or use of the Licensed Material.

3. **Maintenance and Update Services.** Neither maintenance services nor update services are included in this Agreement. As used in the Agreement, the term "maintenance services" includes notice to LICENSEE of latent errors in the Licensed Material and rectification thereof.

4. **Title.** LICENSEE agrees that the Licensed Material is, and shall at all times remain, the property of LICENSOR. LICENSEE shall have no right, title or interest therein or thereto except as expressly set forth in this Agreement.

5. **Duplication and Disclosure.** LICENSEE and LICENSOR agree that the Licensed Material contains proprietary UNIX Operating System Software belonging to WESTERN and licensed by WESTERN as part of UNIX. LICENSEE represents that it has obtained a valid license from WESTERN to use UNIX and has provided a copy of this license to LICENSOR. LICENSEE agrees to treat WESTERN's included proprietary software in the same manner as if such included proprietary software had been received directly pursuant to LICENSEE's UNIX license from WESTERN. It is agreed that should the LICENSEE's UNIX license from WESTERN be modified, for example, to include additional CPU's, the modified license will govern the use of the Licensed Material. LICENSOR need not be notified of such modifications. Likewise, it is agreed that if LICENSEE's UNIX license from WESTERN is terminated, LICENSEE's license from LICENSOR is also terminated.

LICENSEE may license the use of the Licensed Material to third parties provided that such licensing is acceptable to WESTERN. Material so provided may be either modified or unmodified. LICENSEE may, but need not, charge a fee for such licenses. LICENSOR need not be notified of such licensing.

6. **Proper Credit and Recognition.** In the use and/or distribution of the Licensed Material, LICENSEE will take all reasonable measures to give appropriate credit to The Regents of the University of California and the Electrical Engineering and Computer Sciences Department at the Berkeley Campus of the University of California for their roles in the development of the Licensed Material, and shall make this condition a requirement of any agreement by which LICENSEE distributes the Licensed Material.

7. **Warranty and Limitation of Liability.** LICENSOR MAKES NO WARRANTIES, EITHER EXPRESS OR IMPLIED, AS TO ANY MATTER WHATSOEVER, INCLUDING WITHOUT LIMITATION, THE CONDITION OF THE LICENSED MATERIAL, ITS MERCHANTABILITY OR ITS FITNESS FOR ANY PARTICULAR PURPOSE.

LICENSEE agrees to indemnify, defend, and hold harmless LICENSOR, its agents, officers, and employees, either in their individual capacities or by reason of their relationship to LICENSOR, and its successors, in respect to any expense, claim, liability, loss or damage (including any incidental or consequential damage) either direct or indirect, whether incurred, made or suffered by LICENSEE or by third parties, in connection with or in any way arising out of the furnishing, performance or use of the Licensed Material. LICENSEE's obligations under this paragraph include, but are not limited to, its obligation to indemnify, defend, and hold LICENSOR, its agents, officers, and employees harmless in the case of any claim of copyright or patent infringement based in any manner on LICENSEE's use of the Licensed Material.

8. **Legal Expenses.** In case legal action is taken by either party to enforce this Agreement, all costs and expenses, including reasonable attorney's fees, incurred by the prevailing party in exercising any of its rights or remedies hereunder or in enforcing any of the terms, conditions, or provisions hereof shall be paid by the other party.

9. **Severability.** If any part, term or provision of this Agreement shall be held illegal, unenforceable or in conflict with any law of a federal, state or local government having jurisdiction over this Agreement, the validity of the remaining portions or provisions shall not be affected thereby.

10. **Governing Law.** This Agreement shall be construed and enforced according to the laws of California as applied to contracts made and to be performed in California. If LICENSEE is located outside the United States of America, the parties hereto agree that any dispute arising in connection with this agreement will be settled by the ICC Court of Arbitration.

11. **Paragraph Headings.** The headings herein are inserted for convenience only and shall not be construed to limit or modify the scope of any provision of this Agreement.

12. **Entire Agreement.** This Agreement contains all the agreements, representations, and understandings of the parties hereto and supersedes any previous understandings, commitments or agreements, oral or written.

13. **Authority.** Each of the undersigned warrants that he/she has the authority to bind to this Agreement the party which he/she represents.

IN WITNESS WHEREOF, the parties hereto have executed this Agreement as of the day and year first above written.

THE REGENTS OF THE
UNIVERSITY OF CALIFORNIA
(Licensor)

By ________________________________

____
(Title)

____

____
(Licensee)

By ________________________________

____
(Title)

Berkeley Software for UNIX† on the VAX‡

4.1bsd version of May, 1981

A new version of the UNIX system for the VAX family of computers is available from the Computer Systems Research Group of the University of California at Berkeley. This is an updated package of software for UNIX/32V† licensees, and includes a refined version of the paging kernel for the VAX as well as a large number of other programs. This document describes the major differences between standard UNIX/32V as distributed by Western Electric and the May, 1981 distribution known as 4.1BSD.

The new release may be used in two ways: as a bootstrap system for new hardware (or to bootstrap systems which were previously running 3bsd or UNIX/32V), or to update a system running the 4bsd release of November, 1980 (now called 4.0bsd). Hardware configurations supported for booting are described below and in the document "Installing and operating 4.1bsd". The things most notable for sites which are updating 4.0bsd to this new release are:

1) Additional hardware support, including a number of new mass storage peripherals, and especially support for the VAX 11/750.
2) Performance enhancement to the paging portion of the system, which should provide additional throughput in virtual-memory intensive environments.
3) A number of bug fixes. Most of the bug fixes are minor, but several irritating bugs were present in 4.0bsd; this distribution, when used as an update, attempts to correct the known bugs while introducing few new ones.
4) The INGRES data base system, developed at Berkeley by the INGRES project, is now supplied with the standard system.

System facilities

Hardware support

The system runs on VAX 11/750 and VAX 11/780 processors and supports the standard DEC mass storage peripherals: RM03, RM05, RP06, RP07, RM80, and RK07 disks; TS11, TE16, TU45 and TU77 tapes. DEC standard bad block handling on disk drives is supported on all the DEC disks except the RP06. The EMULEX SC21-V UNIBUS storage module disk controller is supported with AMPEX and CDC 300 Megabyte disk drives and FUJITSU 160 Megabyte Winchester drives. The EMULEX TC-11 tape controller (which emulates a TM-11 DEC UNIBUS controller) is supported with tape drives such as the KENNEDY 9300. Any supported disk plus a tape drive is sufficient to bootstrap the system on either processor. Two

†UNIX and UNIX/32V are trademarks of Bell Laboratories.
‡VAX and PDP are trademarks of Digital Equipment Corporation.

drives are required for operation of the system using RK07's. Most RK07 systems have no tape media, and so we can supply a distribution on two RK07 packs.

For terminal interfaces, in addition to the standard (non-DMA) DEC DZ-11 terminal interface, (DMA) DH-11 emulators are supported for terminal support, such as the ABLE DH/DM (which replaces the ABLE DMAX). The system also provides support for standard line printer interfaces emulating the DEC LP-11 and the use of 1200 baud terminals such as a DECWRITER-III acting as a slow speed printer. Printer-plotters such as made by BENSON/VARIAN or VERSATEC are also supported with standard drivers.

Unlike previous releases of the system, this release supports any number of any of these devices. The devices may be placed arbitrarily on any available MASSBUS and UNIBUS interfaces. The system configures at boot time, locating available devices, using a system configuration compiled into the kernel. The configuration description contains all the information about the topology of the machine and the addresses at which the various devices are located. It is possible (and desirable) to write the description using "pattern matching" to only partially specify some of the interconnects.

The system configuration program sizes system data structures based on a specification of the maximum number of active users to be present on the system. To build a system for a larger or smaller workload you only need change this single constant. The system also now initializes the parameters to the paging system and sizes its file system buffer cache based on the amount of available memory; it is no longer necessary to adjust these by hand.

The system supports access to the 11/780 console floppy disk; currently hardware (interrupt latency) problems prevents use of the 11/750 console cassette tapes; we hope to resolve this soon.

For further information on device support and recommendations for configuration of VAX systems to run UNIX see "Hints on configuring a VAX to run UNIX" by Bob Kridle and Bill Joy.

System reliability and performance

The system reboots automatically after hardware and software failures, running an automatic procedure that recovers from normal minor disk inconsistencies. If hardware or software failures cause unexpected problems on the disks, then an interactive semi-automatic repair program can be used to fix up the disks.

The system is fully and transparently demand paged. As distributed it will support individual process sizes up to 6 Megabytes each of data and stack area and 6 Megabytes of program. These numbers can be increased on systems willing to dedicate increased disk space for paging the process image.

The default loader format is load-on-demand, and allows large processes to start quickly. A *vfork* system call allows a large process to execute other processes without copying its data space.

The system performance has been enhanced in a number of ways. Relative to UNIX/32V, the basic system overheads have been reduced by tightening up the system code and improving system data structures. Disk throughput has been increased by increasing the logical block size on the disks to 1024 bytes.* File system performance may also be increased, beyond that normally obtained with UNIX/32V or the 4.0bsd release, by properly interleaving the file systems to account for device speed; instructions on this and proper interleaving constants are given in the setup instructions for the system and in the *mkfs*(8) manual page.† System algorithms such as the swapping and file system caching algorithms have also been improved to increase system performance.

Since the 4.0bsd (previously known as 4bsd) distribution of November, 1980, performance of the system under heavy paging load has been substantially improved by correcting a problem with placement of pre-paged pages. The system now pre-pages more data, greatly benefiting processes which have locality in their behavior. System degradation due to pre-paging and lock-out of other processes during heavy paging has been greatly reduced (even though more pre-paging is done).†

Since the 4.0bsd release, facilities have also been added for processes which serially reference large amounts of virtual memory to inform the system of this kind of paging behavior. This helps the system to deal with these kinds of processes, which are not as well served by using the current default paging algorithm. Processes which are known to need only a small amount of memory but which tend to accumulate large amounts of memory due to strange page referencing patterns may declare a soft limit on the amount of memory to be used.‡

For further information on system performance and recent measurements see the (revised) paper "Performance of UNIX on the VAX" by Bill Joy.

What this distribution does not contain

A number of new system facilities are under development at Berkeley and will be included in future releases of the system. We mention them here mostly to point out that they are not in this system:

1) Arpanet TCP/IP support. This is being developed at BBN, and will be integrated into the system very shortly; if you need this right away, you should contact Alan Nemeth at BBN.
2) Local network support.
3) Inter-process communication support, providing facilities for user processes to take advantage of the network support and for writing cooperating user processes.
4) Shared-segment access to large files, mapped in virtual memory.
5) Performance changes for the file system to provide a higher access rate for applications which are currently i/o bound. of very large scale image processing.

Development of the system in these areas is currently underway. The facilities should become available in roughly the order listed above. The design for these new facilities is spelled out in other documents available from our group.

---

† Systems that run UNIX/32V convert to the new format by saving files using the tape archiver and reading them into the new system. Because of the format changes in the file system and because some of the changes described here required recompilation of all programs, a bootstrap tape and all programs are distributed.

† The actual change here is to place the pre-paged pages at the bottom of the memory free list, rather than putting them into the systems global "clock" replacement loop. This is a very minor change, but has significant performance implications.

‡ If the system needs memory and there are processes which are over their declared limits, the system tends to take pages from these processes first.

User Software

The following sections detail additional user-level software available with this distribution; we describe only software which is not part of the UNIX/32V distribution. Full documentation and source for this (and all supplied) software is made available with the distribution.

Languages for the VAX

Interpreters for APL, LISP and both an interpreter and compiler for Pascal. The APL interpreter is the PDP-11 version, moved to the VAX and has not been extensively used.* The LISP system, known as "Franz Lisp", is written in C and LISP, includes both an interpreter and a compiler, and is compatible with a large subset of MACLISP. The Pascal system is the instructional system that has been distributed previously for PDP-11's. The language implemented is standard Pascal. The implementation features excellent diagnostics, and allows separate compilation and use of C and FORTRAN procedures with full type checking.

A display editor

The tape includes the display editor, *vi*, (vee-eye) that runs on over 100 different intelligent and unintelligent display terminals. This editor uses a terminal description data base; a library of routines for writing terminal independent programs using this data base is also supplied. The editor has a mnemonic command set that is easy to learn and remember, and deals with the hierarchical structure of documents in a natural way. Editor users are protected against loss of work if the system crashes, and against casual mistakes by a general *undo* facility as well as visual feedback. The editor is usable even on low speed lines and dumb terminals.

Command and mail processing programs

The tape also includes a new command processor *csh* that caters to interactive users by providing a history mechanism so that recently given commands can be easily repeated. The shell also has a powerful macro-like aliasing facility that can be used to tailor a friendly, personalized, command environment. A new interactive mail processing command supports items such as subject and carbon copy fields, and distribution lists, and makes it convenient to deal with large volumes of mail.

Job control facilities

The system now supports the multiplexing of terminals between jobs. It is no longer necessary to decide in advance that a job is to be run in the foreground or background; running jobs may be moved from the foreground to the background and vice-versa, and mechanisms exist in the C shell *csh* for arbitrating the terminal between the active jobs.

Debugger support

A version of the symbolic debugger *sdb* is included in the distribution that can be used to debug Pascal, C, and FORTRAN 77 programs. The assembler has been rewritten and the C compiler modified to reduce greatly the overhead of using the symbolic debugger.

* A number of groups are working on improved versions of the APL interpreter, and we hope to obtain a better version of APL soon.

Other software

Also included are several other useful packages including programs to simulate the phototypesetter on 200 bpi dot-matrix plotters (these programs were moved from the PDP-11 to the VAX and many fonts available on the Arpanet have been converted to the required format), a bulletin board program, routines for data compression, a slow-speed network for connecting heterogeneous UNIX systems at low cost (1 tty port per connection per machine and no system changes), and a new, flexible macro package for *nroff* and *troff* called *-me*.

LICENSE AGREEMENT

THIS LICENSE AGREEMENT is made and entered into this __________ day of ____________, 19____, by and between THE REGENTS OF THE UNIVERSITY OF CALIFORNIA, a California corporation, hereinafter called "LICENSOR", and ____________________________, a ____________________ having its principal office at ________________________________, hereinafter called "LICENSEE";

WITNESSETH:

WHEREAS, WESTERN ELECTRIC COMPANY, INCORPORATED ("WESTERN") is the proprietor and owner of the UNIX/32V Time Sharing System, Version 1.0 ("UNIX"); and

WHEREAS, LICENSOR is the proprietor and owner of enhancements and additions to UNIX, which together with UNIX comprise computer programs and documentation entitled, "Fourth Berkeley Software Distribution ("Licensed Material")"; and

WHEREAS, LICENSOR has been authorized by WESTERN to distribute the Licensed Material to licensees who are also WESTERN's licensees for UNIX; and

WHEREAS, LICENSEE warrants and represents that it is a WESTERN licensee for UNIX; and

WHEREAS, LICENSEE desires to obtain from LICENSOR, and LICENSOR desires to grant to LICENSEE, a license to use the aforementioned computer programs and documentation;

NOW, THEREFORE, in consideration of the mutual covenants, conditions and terms hereinafter set forth, and for other good and valuable consideration, LICENSOR hereby leases to LICENSEE the Licensed Material, the components of which are described on annexed Schedule A, subject to a non-transferable, nonexclusive license ("License"), which is hereby granted to LICENSEE, to use such Licensed Material upon the terms and conditions hereinafter set forth; and LICENSEE hereby accepts such lease subject to the License solely upon such terms and conditions.

1. **Term.** The term of this Agreement shall commence on the date hereof, and, unless sooner terminated as hereinafter set forth, shall extend indefinitely.

2. **Fees.** The fee for this license and one physical copy of the Licensed Material is three hundred dollars ($300.00). LICENSEE may obtain additional copies of the Licensed Material and copies of new releases of the Licensed Material which the LICENSOR may make available by paying a duplication fee of three hundred dollars ($300.00) per copy. All such copies shall be subject to the terms and conditions of this agreement. All fees are due and payable in advance.

Such fee does not include local, state or federal taxes, and LICENSEE hereby agrees to pay all such taxes as may be imposed upon LICENSEE or LICENSOR with respect to the ownership, leasing, licensing, rental, sale, purchase, possession or use of the Licensed Material.

3. **Maintenance and Update Services.** Neither maintenance services nor update services are included in this Agreement. As used in the Agreement, the term "maintenance services" includes notice to LICENSEE of latent errors in the Licensed Material and rectification thereof.

4. **Title.** LICENSEE agrees that the Licensed Material is, and shall at all times remain, the property of LICENSOR. LICENSEE shall have no right, title or interest therein or thereto except as expressly set forth in this Agreement.

5. **Duplication and Disclosure.** LICENSEE and LICENSOR agree that the Licensed Material contains proprietary UNIX Operating System Software belonging to WESTERN and licensed by WESTERN as part of UNIX. LICENSEE represents that it has obtained a valid license from WESTERN to use UNIX and has provided a copy of this license to LICENSOR. LICENSEE agrees to treat WESTERN's included proprietary software in the same manner as if such included proprietary software had been received directly pursuant to LICENSEE's UNIX license from WESTERN. It is agreed that should the LICENSEE's UNIX license from WESTERN be modified, for example, to include additional CPU's, the modified license will govern the use of the Licensed Material. LICENSOR need not be notified of such modifications. Likewise, it is agreed that if LICENSEE's UNIX license from WESTERN is terminated, LICENSEE's license from LICENSOR is also terminated.

LICENSEE may license the use of the Licensed Material to third parties provided that such licensing is acceptable to WESTERN. Material so provided may be either modified or unmodified. LICENSEE may, but need not, charge a fee for such licenses. LICENSOR need not be notified of such licensing.

6. **Proper Credit and Recognition.** In the use and/or distribution of the Licensed Material, LICENSEE will take all reasonable measures to give appropriate credit to The Regents of the University of California and the Electrical Engineering and Computer Sciences Department at the Berkeley Campus of the University of California for their roles in the development of the Licensed Material, and shall make this condition a requirement of any agreement by which LICENSEE distributes the Licensed Material.

7. **Warranty and Limitation of Liability.** LICENSOR MAKES NO WARRANTIES, EITHER EXPRESS OR IMPLIED, AS TO ANY MATTER WHATSOEVER, INCLUDING WITHOUT LIMITATION, THE CONDITION OF THE LICENSED MATERIAL, ITS MERCHANTABILITY OR ITS FITNESS FOR ANY PARTICULAR PURPOSE.

LICENSEE agrees to indemnify, defend, and hold harmless LICENSOR, its agents, officers, and employees, either in their individual capacities or by reason of their relationship to LICENSOR, and its successors, in respect to any expense, claim, liability, loss or damage (including any incidental or consequential damage) either direct or indirect, whether incurred, made or suffered by LICENSEE or by third parties, in connection with or in any way arising out of the furnishing, performance or use of the Licensed Material. LICENSEE's obligations under this paragraph include, but are not limited to, its obligation to indemnify, defend, and hold LICENSOR, its agents, officers, and employees harmless in the case of any claim of copyright or patent infringement based in any manner on LICENSEE's use of the Licensed Material.

8. **Legal Expenses.** In case legal action is taken by either party to enforce this Agreement, all costs and expenses, including reasonable attorney's fees, incurred by the prevailing party in exercising any of its rights or remedies hereunder or in enforcing any of the terms, conditions, or provisions hereof shall be paid by the other party.

9. **Severability.** If any part, term or provision of this Agreement shall be held illegal, unenforceable or in conflict with any law of a federal, state or local government having jurisdiction over this Agreement, the validity of the remaining portions or provisions shall not be affected thereby.

10. **Governing Law.** This Agreement shall be construed and enforced according to the laws of California as applied to contracts made and to be performed in California. If LICENSEE is located outside the United States of America, the parties hereto agree that any dispute arising in connection with this agreement will be settled by the ICC Court of Arbitration.

11. **Paragraph Headings.** The headings herein are inserted for convenience only and shall not be construed to limit or modify the scope of any provision of this Agreement.

12. **Entire Agreement.** This Agreement contains all the agreements, representations, and understandings of the parties hereto and supersedes any previous understandings, commitments or agreements, oral or written.

Aauan.

64a

64b

Anlagen

13. Authority. Each of the undersigned warrants that he/she has the authority to bind to this Agreement the party which he/she represents.

IN WITNESS WHEREOF, the parties hereto have executed this Agreement as of the day and year first above written.

THE REGENTS OF THE
UNIVERSITY OF CALIFORNIA
(Licensor)

By ______________________________

____
(Title)

____

____
(Licensee)

By ______________________________

____
(Title)

``Small is Beautiful and Other Thoughts on Programming Strategies''

Bibliography

John R. Mashey -- Bell Laboratories, Whippany, N. J. 07981

BEL77A Belady, L. A., and Lehman, M. M. The Characteristics of Large Systems. Proc. Conf. on Research Directions in Software Technology, P. Wegner, Ed., Oct. 10-12, 1977, Brown University.
Change, growth, entropy. See also earlier works by same authors.

BOE73A Boehm, B. A. Software and its impact: a quantitative assessment. Datamation 19, 3 (May 1973), 48-59.
Programming costs a lot; it will get worse before it gets better.

BRO75A Brooks, F. P., Jr. The Mythical Man-Month. Addison-Wesley, Reading, Mass., 1975.
Readable classic.

CHA74A Chase, W. P. Management of System Engineering. Wiley-Interscience, New York, 1974.
See ``Fallibility'' and ``Evil of Complexity'', pp. 16-20.

DAL77A Daly, E. B. Management of Software Development. IEEE Trans. on Software Eng. SE-3, 3(May 1977), 230-242.
Data on ESS programming at GTE/Automatic Electric.

DIJ72A Dijkstra, E. W. The Humble Programmer. Comm. ACM 15, 10(Oct. 1972), 859-866.
Evils of complexity; PL/I as the fatal disease.

DOL76A Dolotta, T. A., and Mashey, J. R. An Introduction to the Programmer's Workbench. Proc. Second Int. Conf. on Software Engineering, Oct. 13-15, 1976, 164-168.
Design philosophy (7.2): we argued a lot about this but it worked.

EDP71A Advanced Projects in Data Processing. EDP Analyzer 9, 11(Nov. 1971), 1-14.
Hardly anything works.

EDP77A Getting the Requirements Right. EDP Analyzer 15, 7(July 1977).
...is not easy to do.

ELS76A Elshoff, J. L. An Analysis of Some Commercial PL/I Programs. IEEE Trans. on Software Eng. SE-2, 2(June 1976), 113-120.
PL/I: 100K lines of code: most ``features'' are hardly ever used.

ELS77A Elshoff, J. L. The Influence of Structured Programming on PL/I Program Profiles. IEEE Trans. on Software Eng. SE-3, 5(Sept. 1977), 364-368.
Real data in place of faith in how wonderful it will be.

GAL75A Gall, J. SYSTEMANTICS How Systems Work and Especially How They Fail. Quadrangle/New York Times Book Company, New York, 1975.
Systems work poorly or not at all: a light treatment.

HOR75A Horowitz, E., Ed. Practical Strategies for Developing Large Software Systems. Addison-Wesley, Reading, Mass., 1975.
Articles by many; see esp. Royce and Schwartz.

JOY71A Joyce, W. B. Organization of Unsuccessful R&D Projects. IEEE Trans. on Engineering Management Vol. EM-18, 2(May 1971), 57-65.
Most R&D projects fail; planning for failure saves money.

KER76A Kernighan, B. W., and Plauger, P. J. Software Tools. Addison-Wesley, Reading, Mass., 1976.
Good tools built before your eyes; déja vu for UNIX folks.

KER79A Kernighan, B. W., and Mashey, J. R. The UNIX™ Programming environment. ~~Software-Practice and Experience 9, (1979), 1-15.~~ COMPUTER, APRIL 1980.
See especially "UNIX AND MODERN PROGRAMMING METHODOLOGIES".

KNU71A Knuth, D. E. An empirical study of FORTRAN programs. Software-Practice and Experience 1, (1971), 105-133.
Human intuition about programs and languages is terrible.

MYE76A Myers, G. J. Software Reliability. Wiley-Interscience, New York, 1976.
good thoughts on programming in general; balanced (if conventional) view.

MYE77A Myers, S., Sweezy, E. E. Why Innovations Falter and Fail: A Study of 200 Cases. Cunningham, D. E., Craig, J. R., Schlie, T. W. Eds. Technological Innovation, Westview Press, Boulder, Col. 1977, 83-99.
Nothing works; 58 ideas to yield one successful product.

NAU69A Naur, P., Randell, B. (eds.) Software Engineering. Scientific Affairs Division, NATO, Brussels 39, Belgium, Jan. 1969.
Required reading, if you can find a copy. See McIlroy's comments.

PAP73A Papanek, V. Design for the Real World. Bantam Books, Toronto, 1973.
Friends of Bucky Fuller will like this one.

PAP77A Papanek, V., and Hennessey, J. How Things Don't Work. Pantheon Books, New York, 1977.
Good thoughts for designers of anything; see Ch. 7 especially.

RIT77A Ritchie, D. M. The UNIX Time-Sharing System -- A Retrospective. Proc. of Tenth Hawaii International Conf. on Systems Science, Honolulu, Hawaii, Jan. 1977.
Many good thoughts and thought-provoking history.*

ROS76A Ross, D. T. Homilies for Humble Standards. Comm. ACM 19, 11(Nov. 1976), 595-600.
A rational view on the role of standards; read before imposing any.

SCH73B Schumacher, E. F. Small is Beautiful. Harper & Row, New York, 1973.
No mention of programming, but the philosophy is applicable.

STR72A Strunk, W., Jr., and White, E. B. The Elements of Style, 2nd ed.. MacMillan, New York, 1972.
Everyone should read this once a year and write code that reads like this.

THO75A Thompson, K. In Structured Programming, The UNIX Command Language. Infotech State of the Art Report, London, Mar. 10-12, 1975. 375-384.
Good minimalist philosophy; features that fill much-needed gaps.*

* For newer versions, see Bell System Technical Journal 57, No. 6, Part 2 (July-August 1978), an entire issue on UNIX and its offspring.
($2, BSTJ Circulation Group, Bell Labs, Whippany Rd. Whippany, N.J. 07981)

Richard T. Ace	New York Institute of Technology
H. R. Adams	Hughes Aircraft Company
Allen Akin	Georgia Tech
Brian H. Anderson	Mini Systems Division
Karl Auerbach	Interactive Systems Corporation
Cyrus Azar	AMPEX
William H. Baker III	Jamestown College
R. L. Baldridge	Mostek Corporation
T. Roger Bane	General Instrument Corporation
Jim Barclay	Digital Equipment Corporation
Edward D. Bard	The University of Texas
Edward Barkovsky	Human Computing Resources
John Bass	Fortune Systems
John L. Bass	Fortune Systems
Mark T. Bell	University of South Carolina
Thomas Beres	RLG Corporation
Richard R. Berlint	Sandia National Laboratories
Scott Bertilson	Rosemount Inc.
Cathy Beste	The Robert Wood Johnson Foundation
Roger Bielfeld	Case Western Reserve University
Bradford E. Blasing	University of Minnesota
Bonnie Bonnell	Interactive Systems Corporation
Reidar Bornholdt	Columbia University
David L. Boudreau	Dictaphone Corporation
Joan S. Bowden	National Bureau of Standards
Jeff A. Bowles	University of Oklahoma
Ed Bradford	Bell Labs
John M. Bradley	Univerity of Texas
Richard Bratt	BBN Computer
Jack Brennan	JMI Software Consultants, Inc.
Jerry Brenner	Digital Equipment Corporation
Kenneth L. Brody	Codex Corporation
Ronald L. Broersma	N O S C
Walter E. Brown	Moravian College
Michael A. Brzustowicz	Bell Telephone Laboratories, Inc.
Robert Buaas	N O S C
Douglas Burks	Quotron Systems
Brian C. Burton	The University of Texas
Robert H. Burton	Martin Marietta
William G. Cage	Mostek
Alfredo J. Campos	The University of Texas
Keith Carlson	Texas Instruments
David Carver	University of Texas
Michael Cerda	University of Texas
Jay W. Chalmers	HAO/NCAR
Brownwell Cahlstrom	Bolt, Beranek, and Newman, Inc.
Patrick C. Chen	Exxon Production Research
Albert Chu	System Industries
David Clemans	Systems Development Corporation
David H. Cleveland	Lawrence Berkeley Laboratory
James R. Clifford	Los Alamos National Laboratory
J. Cloutier	Bell Laboratories
Michele Cloutier	Naval Underwater Systems Center

Phil Cohen	University of California
Clement T. Cole	Tektronix, Inc.
Richard Colella	National Dureau of Standards
Ronald Coll	Ford Aerospace and Communications Corp.
R. L. Cock	Bunker Ramo
Ralph P. Cooley	Western Electric
Alfred Correira	The University of Texas
Lano Cox	University of Texas
Peggy Craft	The MITRE Corporation
Ron Crawford	Interactive Systems Corporation
Gary Crockett	Bell Telephone Laboratories
Susan Dahlberg	CompuThink
Marini Daniele	University of Milan
Bruce Dawson	Digital Equipment Corporation
Dale DeJager	Bell Labs
Mike DeMoney	Purdue University
P. O. Deckowitz	ITT -- BCD
John L. Donnelly	NCAR
Dan L. Downum	Geotronics Corporation
Gene Dronek	/usr/doc/services
Stephen P. Dyer	Harvard University
Dennis Ebron	AES Data Ltd.
H. W. Egdorf	Los Alamos National Laboratory
David L. Egle	Texas A & M University
Steve Emmerich	Bolt, Beranek, and Newman, Inc.
Amanda Evans	CACHE Corporation/UT
Kyle Evans	National Instruments
Gregory Felton	American Health Systems
Gale C. Fenoglio	N O S C
Melvin Ferentz	Rockefeller University
William H. Firestone	Burroughs Corporation
William G. Fish, Jr.	Analysis & Technology, Inc.
Dr. Roger R. Flynn	University of Pittsburgh
Robin E. Fogle	Analysis & Technology, Inc.
Craig C. Forney	Plexus Computers, Inc.
Chuck Forsberg	Computer Development Inc.
Dan Forsyth	Georgia Tech
Brian Foster	Caltech
Barbara Foushee	TRW Vidar
Adrian Freed	L.E.R.S.
David Galloway	University of Toronto
Alex L. Garcia	The University of Texas
Michael Ghesquiere	Microtel Pacific Research
Dan Gielan	BTL
Frank Gilmurray	Stanford University
Steve Glaser	Tektronix
Tom Glinos	University of Western Ontario
George Goble	Purdue University
Steve Goiwgs	Mostek
Robert Golden	Quotron Systems
Ron Gomes	Human Computing Resources
Kenneth Goodwin	New Jersey State Medical Underwriters
Gary Gorgen	The Wollongong Group

D. J. Gorman	Speery Systems Management
Thomas Philip Gould	Whitesmiths, Ltd.
Marvin L. Graham	Texas Instruments
Michael P. Green	Emerson Electric, Doric Scientific Div.
Gene Grell	University of Texas
Carol Gross	T A S C
David Grossman	Washington University
Neil Groundwater	Analytic Disciplines, Inc.
Nil Grove	Air Force Data Services Center
Douglas A. Gwyn	Geotronics Corporation
Carolyn A. Hahn	Department of Transportation
Robert Haken	U. S. Geological Survey
G. K. Hama	RCMP P. Directorate
Joe Hamlin	Air Force Institute of Technology
Diane Hancharick	The Robert Wood Johnson Foundation
Walter J. Hansen	Olivetti Advanced Technology Center, Inc
Jeff Hardy	University of California, San Diego
Ronald Hare	U. S. Air Force
Ernie Harkins	University of Colorado
Ancell Harold	Massachusetts Institute of Technology
Scott Harris	The University of Texas
Stephen B. Harsh	Michigan State University
Ray Hausman	Extel Corporation
Erik Hedberg	Stanford University
Joseph Hefner	Naval Underwater Systems Center
Michael Heinz	Polytechnic Institute of New York
Dale K. Hensley	TRW/DSSG
David Herington	University of Texas
Ann L. Herrington	Hughes Aircraft Company
Gregory Hill	Trigraph Ltd.
Edward S. Hirgelt	Plantronics/Zehntel
Jeffrey L. Hobson	U.S.G.S.
Donald J. Hoffman	Bunker Ramo
Robert C. Hoffmann	Lawrence Berkeley Laboratory
Kathy Holle	The University of Texas
Clyde Hoover	University of Texas
Robert P. Hopkins	The Cooper Union
James B. Houser	Department of Defense
Nat Howard	Davis, Polk, & Wardwell
Jessie C. Howell	ABA Corp. of VA
Ralph Howenstine	University of Oklahoma
Lewis A. Hughes	South Central Bell
Doug Hunt	Microtel Pacific Research
Sterling Huxley	Tektronix
Jerry Isdale	Interactive Systems Corporation
Peter Ivanov	University of New South Wales
Mary Ann Jackson	M. A. Jackson & Associates
George D. Jenkins	The University of Texas
Donald W. Jennings	Oklahoma City Air Logistics Center
Kathy Jennings	University of Oklahoma
Roya Johnson	The University of Texas
Jeffrey Jongewald	Boeing Aerospace Company
Mildred Joseph	University of Texas

Bev Josephs	BBN Computer Corporation
James Joyce	International Technical Seminars
Charles A. Judge	TISOFT, Inc.
Nathan Kalowski	BBN Computer Corporation
Mark Kampe	Interactive Systems Corporation
Barry A. Kaplan	Raytheon Data Systems
Gordon Kass	Olivetti Advanced Technology Center, Inc
Lou Katz	Columbia University
Luther Keeler	University of Texas
Tom W. Keller	University of Texas
Christopher A. Kent	University of Cincinnati
David N. Kern	The University of Texas
Stuart Kern	American Physical Society
Karen Kielblock	University of Illinois
Richard Kiessig	Intelligent Decisions, Inc.
Douglas P. Kingston, III	US Army, ARADCOM
Dale W. Kirmse	University of Florida
Bill Knapp	General Instrument Corporation
Joshua W. Knight, III	Stanford University
John W. Knox	University of Texas
Margaret H. Knox	University of Texas
Margaret Knox	The University of Texas
Jeffrey Kodosky	National Instruments
George A. Koehler	The University of Texas
John Kornatowski	Rhodnius, Inc.
John Z. Kornatowski	Rhodnius, Inc.
Kim Korner	University of Texas
Glenn Kroeger	Stanford University
Dan Ladermann	The Wollongong Group
John S. Langford	The University of Texas
P. Langston	Davis, Polk, & Wardwell
Thomas W. Lanzatella	
Lewis A. Law	Harvard University
Bob Lawson	University of Toronto
Paul J. Leach	Apollo Computer Inc.
Bill Lee	The University of Texas
Samuel J. Leffler	Sytek, Inc.
B. M. Legg	Hughes Aircraft Company
Joe Lepreau	University of Utah
Craig Alan Levin	BBN Computer Corporation
Oded Lion	Albert Einstein College of Medicine
Carl Lizza	Air Force Institute of Technology
Walt Loew	University of Caifornia
Daniel Lorenzini	Philips Laboratories
Dwayne Lowry	Interactive Systems Corporation
David L. Luce	Analysis & Technology, Inc.
Dr. Robert C. Lummis	Albert Einstein College of Medicine
Tom Lyon	Amdahl Corporation
Joel Magid	Digital Equipment Corporation
Rosemary Malino	U. S. Naval Underwater Systems Center
Ken Mandelberg	Emory University
Leon Mandrona, Jr.	Bunker Ramo
Jerry Manitius	Northern Telecom, Canada

David R. Mark	TRW
Mike Marsh	Purdue University
Darrell B. Martin	Hughes Aricraft Company
Dave Martin	Hughes Aircraft Co.
Dr. Frank E. Martin	Michigan State University
William G. Martin	US Army Development & Readiness Command
John Mashey	Bell Labs
Henry Matthes	Paradyne Corporation
Richard L. Maus, Jr.	Bell Labs
E. A. Mazzari	Hughes Aircraft Company
Doug McCallum	University of Texas
Howard C. McCausland	Harvard University
Joel C. McClung	University of Oklahoma
R. D. McCulloch	Union Carbide Corporation
Michael McDade	Compugraphic Corporation
Michael McFarland	University of Oklahoma
D. E. McIntosh	Hughes Aircraft Company
Wayne A. McLaren	Data General Corporation
James McMullen	Bell Northern Research
Richard Mendez	Bell Labs
John Menges	University of North Carolina
Robert S. Miles	U. S. Army Ballistic Research Laboratory
Jack Miller	NCAR
Richard J. Miller	Veterans Administration/Self-Employed
Bill Mitchell	N. C. State University
Marguerite Moreno	The University of Texas
Arthur S. Morse	Raytheon Data Systems
David Mosher	University of Caifornia
John R. Mullen	The MITRE Corporation
Dennis L. Mumaugh	Department of Defense
John F. Naglee	AT & T
Norman W. Naugle	Texas A & M University
Roger W. Nestle	US Gov./Wright State University
Le H. Nguyen	University of Florida
Steven Carl Nichols	University of California, Davis
Charles Niemeier	BNR, Inc.
J. Gregory Noel	NPRDC
Gregory O'Brien	Digital Equipment Corporation
Michael T. O'Brien	The Rand Corporation
Mike O'Dell	Lawrence Berkeley Laboratory
Suzanne O'Dell	Lawrence Berkeley Laboratory
Steven Oki	Lawrence Berkeley Laboratory
Stephen M. Padgett	University of Texas
Gary E. Page	Docutel Corporation
Kevin Patterson	Computer Automation, Inc.
Bob Paver	The University of Texas
David M. Phillips	University of Texas
John W. Pierce	University of California, San Diego
Jane Poplawski	University of Texas
Samuel H. Poraul	ITT
Rich Ptak	Digital Equipment Corporation
Isaac Rabinovitch	University of California, Santa Cruz
Ernie Rael	ONYX

John Rainey	Ford Aerospace
Edward Rathje	JMI Software Consultants, Inc.
Cathryn L. Ratzloff	TRW Vidar
Brian E. Redman	BTL
Dennis F. Reed	Storage Technology Corporation
Keith G. Regan	COMSAT Corporation
Thomas C. Reinertson	TISOFT, Inc.
James Rhodes	National Instruments
William M. Richards	ABACUS
Dan Ritchie	Department of Defense
Chris Robertson	University of Toronto
Gary A. Roediger	Corporate Computer Services
Ed Rupp	
Jerome Sable	RCA Laboratories
Von Sagrillo	Martin Marietta
Jonathan Haim Saks	MIT -- Lincoln Laboratory
Louis Salkind	New York University
Stanley S. Schenker	Codex Corporation
Debbie Scherrer	Lawrence Berkeley Laboratory
Jeff Schriebman	Jeff Schriebman Consulting
Michael Schucking	University of Texas
German Schumacher	Cerro Tololo Interamerican Observatory
Edward W. Scott, Jr.	Computer Consoles, Inc.
James Seagraves	Bell Labs
John Seamons	Lucasfilm, Ltd.
Norm Seethoff	John Fluke Mfg. Co.
Bill Shannon	Digital Equipment Corporation
Scott Sibley	Ford Aerospace
Greg Siebers	CACHE Corporation
Roger J. Sippl	Relational Database Systems, Inc.
Chester G. Skapczynski	TRT Telecommunications Corporation
Tom Slezak	Lawrence Livermore National Lab
Steven Small	Uniq Computer Corporation
Dennis R. Smith	USC - Information Science Institute
Henry Spencer	University of Toronto
Earl Stahl	Bell Northern Research
Barbara Staudt	Moravion College
Jeff Stearns	John Fluke Mfg. Co.
Armando P. Stettner	Digital Equipment Corporation
Peter Stevens	Bunker Ramo Corporation
W. Richard Stevens	Kitt Peak National Observatory
Pam Stevenson	MicroTrol Systems, Inc.
John Stier	State University of New York
James Stolpa	The University of Texas
James Stolpa	University of Texas
Daniel R. Strick	University of Pittsburgh
Tom Strong	Strong Consulting
Daniel M. Sunday	Johns Hopkins University
Kenneth P. Sympson	Computer Consoles, Inc.
Andrew Tannebaum	WEC/BTL
Cheryl Taub	American Institute of Physics
Bruce M. Taylor	Advanced Business Communications
Judy Terrill	USARMY DARCOM

Dean Thomas	Santa Cruz Operation
Spencer Thomas	University of Utah
Carolyn R. Thompson	Johns Hopkins University
Nancy Thornton	Geotronics Coporation
Michael D. Tilson	Human Computing Resources Corporation
Abigail Tischler	Columbia University
Daniel P. Tkach, Jr.	Uniq Computer Corporation
Fred Toth	Washburn Press
James Truchard	National Instruments
Richard D. Tuck	Hewlett-Packard
Lisa Lynn Turboff	The University of Texas
David R. Turner	EG & G, Idaho
Ray Turner	Genrad
Richard G. Turner	USARMY DARCOM
Bill Tuthill	University of California
Robert W. Tuttle	General Electric Co.
Ronald W. Underwood	Bell Labs
R. Uthurusamy	Exxon Production Research Co.
Shirley Verrett	The University of Texas
Carl Vesper	Fischer & Porter
Roger Vossler	TRW
Sue Wainer	JMI Software Consultants, Inc.
Justin C. Walker	Interactive Systems Corporation
Charles R. Waltrip	Johns Hopkins University
Dan H. Ward	MicroTrol Systems, Inc.
Charles H. Warlick	University of Texas
Kirk Webb	University of Texas
Wally Wedel	The University of Texas
Peter G. Weiner	Interactive Systems Corporation
Gilbert Weingarten	AM Varityper
R. L. Weis	Hughes Aircraft, Ground Systems Group
Howard S. Weiss	Department of Defense
Michael L. Wendel	General Instrument Coporation
Wentworth	Ford Aerospace & Communications Corp.
Anna M. West	BBN Computer Corporation
Dorothy Anne Whitaker	The MITRE Corporation
Dr. James C. Whitney	Dictaphone Corporation
Norman Wilson	Caltech
David G. Wiseman	University of Western Ontario
Robert Worls	Bolt, Beranek, and Newman, Inc.
Amanda Wright	System Industries
Joseph S. D. Yao	Science Applications, Inc.
Brian Yasaki	I. M. S.
Chris Yeung	Microtel Pacific Research
David Yost	
Alan Zaring	University of
P. A. Zinn	Hughes Aircraft Company
Christine Zuest	USARMY DARCOM
Mike Zuhl	Tektronix, Inc.

RELATIONAL
DATABASE
SYSTEMS, INC.

MARATHON

Nearly all computer users store information on disks, modify it regularly, and wish to query it and have reports written from it. If the application is very common, such as payroll, a packaged program may be available to perform these tasks. However, most user's requirements have a great deal of uniqueness. For example, a blood bank's supply tracking needs probably won't be satisfied by the capabilities of a ready-made inventory control package. In fact, many companies inventory control requirements are not met by the capabilities of any pre-programmed packages. Some customization is usually needed, and that means programming. If no package is available at all, the amount of programming needed to start from scratch is often prohibitive.

Many consultants, system designers and programmers are turning to UNIX to increase their custom programming productivity. UNIX provides not only a pleasant environment for software development, but also provides a wide variety of programs that are already written, and designed to work together. Programmers are using these off-the-shelf components to build custom systems without starting completely from scratch. By doing this they are cutting their costs and increasing their productivity enormously.

MARATHON is a collection of such programs designed to aid the building of single or multi-user applications on UNIX, UNIX look-a-like and UNIX-compatible operating systems. MARATHON's components include an interactive query language, an interactive data entry and maintenance program, several report writers, audit trail and recovery programs and all the utilities needed to create, optimize and modify databases.

This set of tools is so flexible and complete that many simple applications need only be configured, and not programmed at all. For situations where detailed customization is a must, powerful interfaces to standard programming languages are naturally part of the system.

MARATHON is the first commercial DBMS for UNIX. It is not based upon any particular academic system or commercial prototype, but rather it draws from the best features of all of them. The sole reason for its existence is the commercial UNIX user, and the design and future directions of the product reflect this.

Good documentation, error messages, enhancements, host language compatability, training and support are all expected by commercial users. RDS is committed to helping system designers and programmers cut the costs and heighten the reliability and expandability of their applications. The MARATHON system and its report writers are the base for that help.

No application should be built for UNIX starting from scratch. Any data intensive software system should include the data manipulation and maintenance functions provided by MARATHON. Systems designers and applications programmers owe it to their users to start building an application with a firm foundation.

1208 Apollo Way, Suite 503
Sunnyvale, California 94086
Telephone 408/746 0982

Relational DBMS

MISTRESS

A Relational Database Management System
For small to medium applications

MISTRESS

A NEW RELATIONAL DATABASE MANAGEMENT SYSTEM
DESIGNED BY THE CREATORS OF MRS

MISTRESS

READILY CUSTOMIZED, FULLY MAINTAINED,
EXTENSIVELY SUPPORTED

MISTRESS

WRITTEN AND IMPLEMENTED USING
FULL SOFTWARE ENGINEERING PRINCIPLES –
MODULAR DESIGN, TOP-DOWN PROGRAMMING,
TRANSPORTABLE CODE

MISTRESS

DOWNWARD-COMPATIBLE WITH THE MRS
QUERY LANGUAGE

GENERAL FEATURES:

- a superset of the English-like Query Language used in MRS
- a host-language interface via "C" callable procedures
- restrictions on the numbers of tuples and attributes removed
- an ongoing program of enhancement and development

- more data types:
  several varieties of character strings
  3 kinds of integers: 8, 16, and 32-bit
  2 kinds of floating point: 32 and 64-bit
  3 formats of date data type
  dollar data type for commercial users

APPLICATIONS PACKAGES:

A number of applications packages are currently under development. These include:

- report generator
- screen oriented forms interface
- transaction processing system

Other packages are being designed for future release. Custom applications and other services will be undertaken on request at negotiable rates.

MISTRESS is designed to run on most existing PDP-11 and LSI-11/23 configurations under the Unix™ operating system (V7, PWB or V6). In addition, it is targeted for other configurations and operating system environments.

MISTRESS is sold under both standard and educational licences. A standard licence includes binaries, documentation, and one year's support. Ongoing support is available for a yearly fee. An educational licence, at a special price for non-profit academic institutions, includes only binaries and documentation, with no support.

Interested in acquiring a MISTRESS? Write to:

RHODNIUS INC.,
Product Information,
P.O. Box 1, Station D,
Scarborough, Ontario
CANADA M1R 4Y7

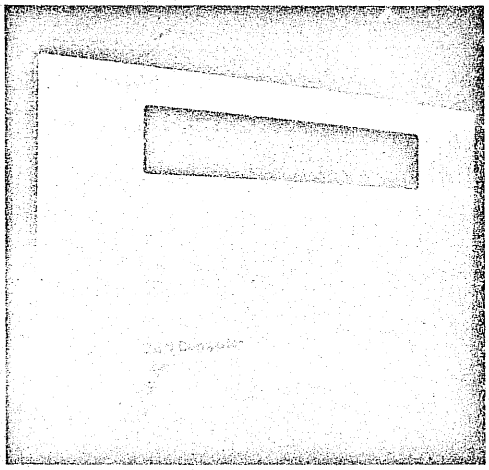
For prompt service, telephone

John Kornatowski
(416) 922-1743

Unix is a registered trademark of Western Electric

C/30 Packet Switch Processor

Features



- **130 packets/second full duplex communication line throughput**
- **Plus 200 packets/second full duplex host computer throughput**
- **ARPANET network technology**
- **135 ns powerful microprogrammable architecture**
- **Supports up to 6 X.21, X.21 (bis), Bi-Sync or ARPANET Sync Ports**
- **Also supports up to 4 1822/ARPANET Host Ports**
- **Also supports up to 64 Medium-speed Async/Sync Terminals**
- **Self-configuring on start-up**

Summary

The C/30 packet switch processor was designed specifically for high-speed communications environments. A very fast, powerful, microprogrammed CPU architecture provides raw speed and basic communication instruction sets.

Much of the complicated I-0 logic is performed by the central microprocessor which both reduces I-0 component count by a factor of 3 or 4 and allows one basic I-0 device to be microcode configured as many different devices. For example, the basic serial communications-line device can handle Asynchronous devices from 50 to 19,200 baud, with or without modem control, Synchronous, or Binary Synchronous devices up to 56 kbaud, all by simply changing microcode.

I-0 devices are supplied with several microcode routines to allow them to handle a variety of devices. A basic micro-assembler and micro-loader are available as well.

Optionally, the C/30 is supplied with the ARPANET Interface Message Processor software, or the ARPANET Terminal Interface Processor software. Tens of man-years have been invested in this code to allow it to operate the world's biggest packet switch network.

Central Processor

The C/30 CPU is based around a fast microprogrammed CPU with an instruction set designed for communications environments. The basic elements are a 1k x 20-bit register file, a 512 x 32-bit microcode ROM containing the loader, debugger and console logic, and a microcode memory in sizes of 2k, 4k or 8k x 32 bit which contains the macro-instruction set and I-0 emulation. Microcode has the ability to run in either the 16-bit or 20-bit mode at the macro instruction level. Microcode memory is loaded from a microcassette holding up to 100kb.

The BASIC C/30 also includes 2 serial async I-0 ports, (one for the console terminal, the other for the micro-cassette loader), a 4-slot chassis with battery backup power supply, and 32k words of 20-bit 405ns semiconductor memory. An Error Detection and Correction system is also standard.

An EXPANDED C/30 provides an 8-slot chassis and larger power supply system.

C/30 systems are provided with microcoded emulation of a minicomputer's instruction set, providing simple access to ARPANET packet switch network software.

I-O System

The I-0 system is greatly simplified compared to most current minicomputers. Most C/30 I-0 devices consist of only voltage conversion devices, serial to parallel conversion, and a variably sized FIFO (first-in, first-out) buffer. All complicated logic is performed in the CPU.

This approach produces low-cost I-0 devices because very little hardware is actually used, and one hardware design can function as many different devices by changing microcode.

For example, a single design of serial communications port can support any kind of Asynchronous Device, with speeds from 50 baud to 19,200 baud; Synchronous Device, or Binary Synchronous Device with speeds to 56k baud. Different microcode is required, but this is kept in RAM memory (loaded from microcassette) and allows the system's OEM to use a very small number of interface designs for a very large number of different applications.

Most C/30 I/0 devices use small daughter boards to customize the voltage conversion for different line disciplines, for example, RS-232, Bell 303, RS-422, MIL-188-114, etc.

Memory

The C/30 includes the 32kw of 20-bit memory. Additional memory is available in sizes of 32kw. All memory is RAM, 405ns access time. Maximum memory is 128k words.

Error Detection and Correction (EDAC) is standard. This provides 6 additional bits of checking, allowing detection of all single- and double-bit failures, and correction of all single-bit failures.

C/30 Memory is currently based on 16k memory ICs.

Communications I-O Devices

The C/30 has four basic communication I-0 designs:

- Medium-speed Async/Sync Interfaces
- High-speed Async/Sync Interfaces
- High-speed HDLC/SDLC Interfaces
- High-speed 1822/ARPANET Interfaces

Each design has a number of daughter-boards, which provide voltage conversion as required and associated microcode to handle the interface for a specific line protocol.

The following currently exist:

- For Async Interface Design:
  - 20-ma current loop
  - EIA interfacing to terminals (modem control not included)
  - EIA interfacing to modems (modem control included)

  All of the above can handle devices at speeds of 50, 75, 110, 134.5, 300, 600, 900, 1200, 2400, 4800, 9600, 19,200 baud. Full duplex, half duplex, 5/6/7/8 bit data bytes, 1, 1½ and 2 stop bits, even/odd/none/marking parity are all supported.

  The microcode supplied includes the ability to support auto-speed detection. IBM 2741 terminals can also be supported.
- For High-speed Async/Sync Design
  - Bell 303 modern support
  - RS-232C modem support
  - RS 422/423 modem support
  - MIL-188-114 support
  - V.28 support
- For High-speed HDLC/SDLC
  - X 21 support
- For 1822/Arpanet
  - Local 1822 Host support
  - Distant 1822 Host support
  - Very Distant Host (VDH) support (same as Bell 303).

The various communication line designs are available on 14x18 logic cards in a variety of combinations:

5401: 4 1822 Ports, 6 High-speed Async/Sync
5410: 32 Async Ports, 1 1822 Port, 1 Sync/SDLC/HDLC Port.

For complete details, refer to the price list.

Other I-O Devices

Also available is a disk controller designed to handle up to 2 Storage Module Disk (SMD) drives. Many manufacturers offer a variety of disk drives that interface to the SMD bus.

Power

The BASIC C/30 requires 300 watts of power, which can be supplied at 115v or 230v, or 50 or 60 Hz.

Physical Construction

The BASIC C/30 fits in a standard 19-inch-wide rack and uses 12.5 inches of vertical space (20 inches deep).

The microcassette occupies 3½ inches of vertical space (12 inches deep). Most communication I-0 boards include connector panels that use 3½ inches of vertical space in the rear of the cabinet, normally behind the microcassette.

The EXPANDED C/30 requires 17¾ inches of vertical space. All other dimensions are the same.

Environmental

Operating: 2°C to 32°C, humidity 0-90% (noncondensing)
Storage: -15°C to 65°C, humidity 0-98% (noncondensing),
alt: 0-7000 ft

BBN Computer Corporation 33 Moulton Street Cambridge, Massachusetts 02238 (617) 491-1850 Ext.3765

Plexus

P/40 Computer System Hardware Architecture

Plexus P/40 is a 16-bit minicomputer system designed specifically to operate with the popular UNIX® operating system. The P/40 features a large main memory, high-performance peripherals, and built-in error checking and correction for faster, more reliable operation.

Communications processors to offload terminal and other serial I/O functions enhance system performance further. In addition, the P/40 features the industry-standard Multibus® I/O bus, allowing users to tailor systems to particular applications with a wide variety of auxiliary devices.

The P/40 UNIX system is ideal for technical or business environments where demanding professionals require a powerful yet easy-to-use computing facility. A typical P/40 configuration gives 5 to 15 users the memory and disk capacity they expect to find only on the largest minicomputers, but at a cost per user that is competitive with small microcomputer systems.

Features

- Multiprocessor architecture links as many as eight 16-bit microprocessors for high-performance computing and I/O.
- Main memory, available in 256 Kbyte and 1 Mbyte increments, ranges from 256 Kbytes to 4 Mbytes.
- Multibus-compatible I/O bus allows ready attachment of peripheral controllers for a wide range of specialized hardware configurations.
- Modular intelligent communications processors support 8 terminals or communications lines per module. Up to 3 communications processors can be configured, for a total of 24 lines.
- Intelligent disk controllers support 72 to 580 Mbytes of error-correcting Winchester disk storage.
- Standard 9-track magnetic tape drive doubles as a high-speed streaming drive for disk backup.
- The P/40 system supports V7, the Plexus adaptation of the UNIX operating system (version 7).
- P/40 software is compatible across the entire Plexus product line.

High-Performance Main Processor

The P/40 main processor includes a 16-bit CPU, floating-point processor, battery-operated clock/calendar, and vectored interrupt controller.

The CPU (based on the Z8001) has more than 110 distinct instructions that operate on 7 data types, including bits, 32-bit words, and strings. For easy programming, it has 16 general purpose registers and supports separate instruction and data spaces. The CPU also supports privileged instructions.

The floating-point processor performs single-precision (32-bit) or double-precision (64-bit) addition, subtraction, multiplication, and division. It is compatible with the IEEE format.

A battery-operated real-time clock gives the CPU continuous access to the actual date and time of day. The clock remains operational even when AC power is removed from the system.

Multilevel interrupt vectoring saves the CPU from having to identify each interrupt. Sixteen major interrupt levels include I/O transfer complete, memory error, interval timer, page fault and privileged instruction violation.

Burst-multiplexed direct memory access (DMA) is used to transfer data between devices and main memory at rates up to 3 Mbytes per second. Since each peripheral controller has its own DMA, intelligence, and buffer storage, the main processor is interrupted only after an entire I/O operation has taken place.

The system is designed for flexible configuration. All boards plug into a 12-slot backplane. With standard peripherals, the backplane has 4 slots for memory boards and 5 open slots for additional Multibus-compatible I/O controllers.

Efficient Memory Management

Memory management in the P/40 is optimized for high performance computing with the UNIX operating system. The large amount of memory available for operating processes, combined with separate data and instruction spaces, give the system attractive throughput characteristics.

Data transfers between the peripheral controllers and main memory take place through the memory manager. Transferring data directly into the logical address space of any process executing on the main processor significantly increases throughput.

Address translation takes place through high-speed bipolar RAMs that form 16 process spaces. Each process space contains separate instruction and data maps. Each map can contain up to 128 Kbytes of memory – 64 Kbytes of instruction and 64 Kbytes of data. Because each instruction map has read-only access, all the code generated is sharable, reentrant code for efficient memory use. Also, the large number of process spaces reduces the context switching time necessary to run multiple processes.

High-speed memory is available in increments of 256 Kbytes (16K dynamic RAMs) or 1 Mbyte (64K dynamic RAMs), up to a maximum of 4 Mbytes. Memory is divided into pages of 2 Kbytes each.

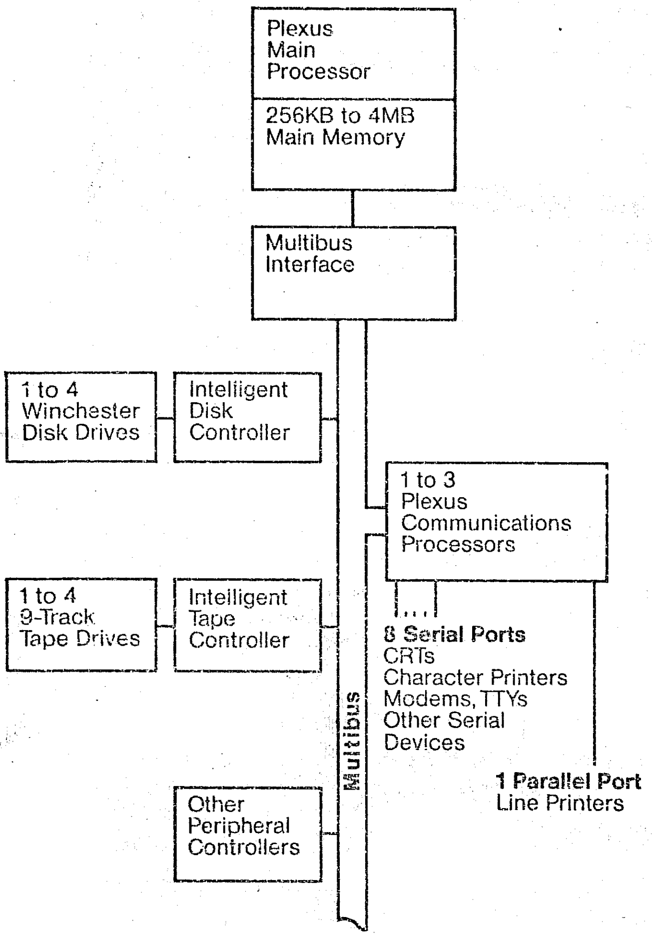
Memory cycle time is 600 nsec. Cycle time includes translation of the logical address to the physical address, memory access, and error detection and correction.

Single-bit memory error correction and double-bit error detection are provided for all accesses to main memory. Data is stored in 22-bit words logically organized as 16 data bits and 6 error correcting bits.

Up to 16 Kbytes of PROM and 2 Kbytes of static RAM are also in the CPU. PROM and static RAM are used for self-test and diagnostic routines.

Standard Multibus Interface

P/40 architecture provides a high-speed synchronous path between the main processor and memory, yet makes available to the user the many Multibus-compatible controllers available in the marketplace.



Plexus P/40 Computer System Block Diagram

Communications Processor Handles Terminals

All serial communication tasks are handled by the Plexus Communication Processor (PCP). It contains a 16-bit processor, memory, 8 serial ports, 1 parallel port and 9 DMA channels. Typically, the PCP is programmed to provide all buffering and protocol conversion necessary to support terminals, modems, and printers. Each full-duplex serial channel supports asynchronous, bisync, or HDLC protocols at software-selectable rates up to 19.2 Kbaud. Each channel has its own DMA, so all 8 can operate at the maximum baud rate simultaneously.

The PCP has 32 Kbytes of RAM with parity and up to 16 Kbytes of PROM. This large memory allows segments of UNIX or customized communications and terminal handling programs to be downloaded to the PCP and executed locally.

The 8 serial ports are RS-232C compatible and have the modem control lines necessary to support standard asynchronous and synchronous modems. The parallel port is designed to support a line printer with a standard (Centronics-compatible) parallel interface.

Intelligent Disk Controller

P/40 has an intelligent disk controller that supports up to 4 industry-standard SMD drives. The controller contains a 16-bit processor, 10 Kbytes of local memory, and a DMA channel. The controller performs multiple sector transfers that span cylinders, automatic error recovery, and transparent error correction for burst errors up to 11 bits long.

The standard Winchester-type disk is a 14-inch fixed disk with either 72 or 145 Mbytes of formatted capacity. The drive has a 35 msec average access time and transfers data at 1.01 Mbytes per second.

Dual-Mode Tape Drive

An intelligent 9-track magnetic tape controller supports up to 4 industry-standard drives. The horizontally mounted unit features automatic threading and can be used in two ways.

In streaming mode, which is used for disk backup, the drive stores 46 Mbytes of data in 4.7 minutes on a 10.5 inch reel of tape. In conventional mode, the ANSI/IBM compatible 1600 bpi drive provides a convenient method for exchanging data with other computer systems.

Built-In Reliability

P/40 minimizes the two most common causes of downtime in small computer systems: memory errors and disk drive failure. The P/40 main processor automatically detects and corrects memory and disk errors.

In addition, the state-of-the-art P/40 system, with its extensive use of LSI circuitry – and reduced parts count – is inherently more reliable than earlier designs.

P/40 was designed from the outset for ease of service. At the time the system is turned on, each processor automatically performs a complete self test and displays the results on the main processor console and on arrays of light-emitting diodes on the boards. The main processor has a diagnostic port that connects a terminal or modem directly to the main processor. This port can be used from a remote location to isolate many hardware and software problems.

All P/40 circuit boards are easy to access and replace. Mean time to repair is under 30 minutes with no special tools.

Designed for Offices

P/40 is designed to operate in an office environment. It requires no special environmental control or power conditioning. The system, including the disk and tape drives, is packaged in an enclosure 42 inches high.

Specifications

Size	
Height	42.0 in. (106.7 cm)
Width	23.0 in. (58.4 cm)
Depth	30.0 in. (76.2 cm)
Weight	295 lb (134 kg)
Power	
115 vac 60 Hz	10 amp
230 vac 50 Hz	5 amp

Environmental
Operating temperature: 5°C-38°C (41°F-100°F)
Relative humidity: 20-80% without condensation

Plexus Computers Incorporated

3375 Scott Blvd.
Suite 340
Santa Clara, Ca 95051

408 · 988 1755

2K681 Printed in U.S.A. AA01

Hints on Configuring VAX* Systems for UNIX†
Revised: May 4, 1981

Bob Kridle

Computer Systems Support Group
U. C. Berkeley
kridle@berkeley, ucbvax!kridle

Bill Joy

Computer Systems Research Group
U. C. Berkeley
wnj@berkeley, ucbvax!wnj

ABSTRACT

This document reflects our experiences and opinions in configuring four existing VAXes and two more that are on order to run UNIX†.

Our prime considerations in choosing equipment are:

- Cost
- Reliability
- Maintainability and Maintenance Cost
- Delivery Time
- Redundancy of the system

We consider components individually and then describe several system packages built from these components, emphasizing independently single-source systems, minimization of cost, and maximal expansion capability.

† UNIX is a trademark of Bell Laboratories.
* VAX, VMS, MASSBUS, and UNIBUS are trademarks of Digital Equipment Corporation.

DISCLAIMER

This documents reflects our *personal opinions*. We are responsible for software and hardware support of VAX systems, and the recommendations we give reflect what we would do. We are careful to note the equipment that we recommend but are not using; we recommend no second-vendor equipment that is not known to be in use successfully at several UNIX sites. In any case you may get a lemon, no matter what you buy. All we promise is that this is what we believe. Let us know what you find out. Prices are constantly changing and may be inaccurate.

We have little familiarity with VMS. Recommendations made here should not be construed to be applicable to any operating system other than UNIX. We have often adapted UNIX to these devices in a way that may not be possible with other operating systems.

Finally, note that we have written this document to share the knowledge we have acquired with you. We have written it because we do not have the time to talk to everyone who needs this information. Please do not call us to confirm the information here or to ask questions about our opinions. We would like to hear of your experiences, or learn of mistakes in this document or products that we know nothing about, but do not have time to chat about the information that is given here. We do welcome electronic mail sent to our addresses as shown on the first page.

OVERVIEW

We first discuss components, listing the alternatives we have tried and sometimes a few we haven't, and then discuss system packages. We buy a substantial portion of the equipment we use from vendors other than DEC. The reasons for choosing second vendor equipment are usually some combination of more current technology, lower cost for equivalent equipment or shorter delivery time.

We do not consider devices that have proven unreliable or whose performance we consider inadequate. In addition, there are many devices that we have no experience with. As a general rule, every new peripheral has required a non-trivial amount of leg work to get up to speed. We suggest using only peripherals that have been previously used successfully on *the type of VAX you are configuring* (780 or 750) or demanding a substantial (50-100 percent) discount for being a guinea pig. Be especially careful of UNIBUS† interfaces. Almost every manufacturer of a UNIBUS widget now includes the VAX as a machine on which his device will work. Few of these devices have been tested in this situation. Many of them will not work without substantial modification.

All prices quoted are *undiscounted, quantity one*, unless stated to be otherwise and are accurate to the best of our knowledge at this writing. We usually receive at least an educational discount and sometimes a quantity discount. These typically reduce the list prices from ten to forty percent. Since discounts vary from area to area and prices change frequently, you should always request quotations from local vendors.

System buyers without ready access to an in-house hardware staff should consider carefully the option of buying as much DEC equipment as possible. If you have the money and time required to do this, there are some strong advantages. Our DEC equipment has, in general, proven somewhat more reliable than the equivalent alternate vendor equipment. Time from equipment delivery to running system is also usually shorter. DEC field service in our area is excellent. Outside service available for non-DEC peripherals is spotty at best.

For smaller installations this option should be carefully considered. It is easier if you can call one party for all your problems, if you can afford it. At Berkeley, we are well past the inventory level where self maintenance begins to pay off even on all DEC systems, so this is not a consideration. One of us (Kridle) manages our local hardware support group.

Unfortunately, the limited selection of configurations currently available sometimes make the all-DEC choice difficult. This is especially true of the smaller configurations as DEC's bottom end peripherals are less satisfactory for UNIX. We say this not just for monetary reasons; functionally and aesthetically we would prefer to have neither the RK07 disk nor the TS11 tape unit in any system we have to deal with.

We recommend getting field service at least on your CPU for the first year. It has paid off for us in the cost of parts alone. You can drop the contract after the engineering changes have tapered off and most of the infant failures have occurred. DEC requires a certain amount of its peripheral equipment on the machine to qualify for field service. We understand that it is company policy not to provide a maintenance contract for a system without a DEC mass storage peripheral. If you intend to purchase a maintenance contract, be certain that your local field service is willing to support at least the DEC equipment you buy.

BANDWIDTH CONSIDERATIONS

Evaluation of the data transfer capacities between the various parts of VAX systems is a complex task that plays a critical part in system configuration. Unfortunately, there is a tremendous amount of misinformation available on this subject and little useful hard data. We have made a few measurements and intend to make more. What we currently know follows.

The 11/780 UNIBUS adapter is the device most frequently shrouded in confusion. DEC documents variously give the bandwidth at between 1.2 MB/sec and 1.5 MB/sec when transferring through a buffered data path. We are not aware of any specifications for the unbuffered data path but have not been able to use it with some devices as slow as 50 KB/sec. One experiment we conducted involved examining the UNIBUS protocol lines with a scope while constantly transferring from a disk drive. We observed that while the drive was transferring at an average rate of about 1.2 MB/sec the UNIBUS was close to one hundred precent busy. This test was conducted on an otherwise idle system. No other devices were active on the UNIBUS and large disk transfers (cylinders) reduced any register set up time to a minimum. We conclude from this that 1.2 MB/sec is the *absolute maximum* transfer rate possible through a 11/780 UNIBUS adapter. Our observations showed that the largest delays while transferring data occurred while the buffered data path was being loaded or unloaded from the SBI. Since the UBA is controlled by a micro sequencer that is also involved in other UBA activities such as processing interrupts, we suspect that on an active UBA this bandwidth *may* be somewhat reduced.

It is also our conclusion that UBA bandwidth is not an important limitation in most applications. Most modern disk controllers have at least 1536 bytes or three sectors of internal buffering. This means two things: first, they can be designed to never produce *data late* error conditions regardless of the transfer size. Second, they can complete an entire track transfer without losing a revolution in all but the most adverse circumstances. UNIX rarely makes single transfers larger than a single disk track.

There are troublesome devices that cannot buffer enough data to guarantee that the maximum size record can always be transferred (6250bpi tape drives) or do not buffer an adequate amount of data (RK07 disk controller). To handle these devices UNIX provides a software interlock mechanism that prevents excess UBA contention.

The MASSBUS adapters are specified to have a higher potential bandwidth of 2.5 MB/sec. Since they are selector channels that allow only one device to transfer data at a time, the realized bandwidth is limited to the rate of the fastest device. The fastest devices currently available from DEC for 11/750 systems or 11/780 systems with a single memory controller transfer at 1.3 MB/sec. Large 11/780 systems with two memory controllers and interleaved memory may run RP07 disk drives that then transfer data at 2.2 MB/sec.*

We have measured the average throughput of disks with a maximum transfer rate of 1.2 MB/sec (RM05 equivalent) using three different interfaces on an idle 11/780 system. The interfaces included a DEC MASSBUS adapter, a foreign vendor SBI interface, and a foreign vendor UNIBUS interface. In all three cases the throughput was within a few percent of the maximum rate for the disk drive.

Our conclusion is that for most UNIX applications, the UNIBUS interface is adequate. It is usually equivalent in performance to MASSBUS interfaced devices and less expensive. The long history of foreign vendor UNIBUS interfacing has created a rich and competitive market of devices. If necessary, a second UNIBUS adapter can be added at a cost similar to that of a MASSBUS adapter and is fully supported by UNIX.

A more interesting bandwidth limit may be established by the memory controller. We suspect that the CPU may be slowed considerably by memory contention when two disk channels are being used simultaneously. This should be alleviated by using interleaved memory controllers. We have a second memory controller, not yet installed, and intend to attempt measurements soon.

The appendix to the *VAX Hardware Handbook* titled "System Throughput Considerations" seems to bear out these impressions and should be read carefully by anyone hoping to understand the consequences for VAX applications involving high bandwidth input or output. If we had data intensive applications we would seriously consider the use of RP07 disks (and interleaved memory controllers) because of the resultant higher burst transfer rate; this will be discussed further below.

* On machines with only one memory controller the RP07 hardware is arranged to transfer at 1.3 MB/sec. It is our understanding that RP07's will work on 11/750's with a single memory controller at 2.2 MB/sec, but RP07s are not currently sold for 11/750s.

MEMORY

All VAXes are sold with some minimum amount of DEC memory adequate to run diagnostics. Additional memory is the lowest risk alternate vendor choice. We buy the remainder of our 780 memory from National Semiconductor or Trendata.** We are currently paying about $5,000 per megabyte. This area is extremely competitive and there are at least 6 possible vendors. By all means, ask for competitive quotes. Assure yourself, however, that you are not the first customer for a new vendor.

Add in memory for the VAX 750 is a newer item and prices are not as low. However, this memory is almost identical to the 11/70 MK11 memory and several vendors have managed to build this product by modifying their previous 11/70 add-in product. Trendata 750 memory is currently listing at $2300 per 256 Kbyte board. A fifteen percent educational discount is available. A substantial GSA discount is also available.

Small quantities (one to two megabytes) are usually available off the shelf from Trendata or National. Large quantities (4 megabytes and up) have taken less than 30 days.

For the 11/780 memory, the RAM chips are socketed, and two replacement chips per board are supplied by both Trendata and National; these are the only two second vendors whose memory we are currently using. You can pull out the board and replace the chip at your leisure. Since single bit errors are corrected this has never involved any unexpected down time for us. There is at least a one year return to factory agreement on the boards, included in the purchase price. Out of warranty repairs are said to typically cost less than $300. We have returned only one board to the plant in about 30 board years.

When purchased from DEC, memory is $19,800 per megabyte in 1 megabyte quantities or $15,950 per megabyte in 2 megabyte quantities for 780s. Maintenance on DEC memory is $231 per megabyte per month with board replacement through field service. The boards are not socketed. Delivery times on memory from DEC have typically been substantially longer than times from second vendors.

If you are going to have more than 4 megabytes of memory to your 780 you will need a CPU expansion cabinet at $4,460 and a second memory controller for $26,600 that includes a second half-megabyte of DEC memory. It is not currently possible to put more than 2 megabytes on a 750†. This reflects both the maximum addressing capacity of the controller, and the lack of space for an additional controller.

DISKS

There are three alternatives for disks. The first is to buy the disk hardware from DEC. The available equipment includes 67M byte removable disks (RM03), 252M byte removable disks (RM05) and 124M byte non-removable disks (RM80), and a new and aggressively priced 504M byte non-removable disk (RP07) that can be made to run at 2.2 MB/sec on 11/780's with interleaved memory.*

** A list of second vendors and their phone numbers is given at the end of the document.
† We have heard that a recent DEC field change order prevents use of more than 1.75 Megabytes of memory on an 11/750.
* Disk sizes shown throughout this document are in bytes of formatted space available.

DEC also sells RK07 cartridge disks that we don't consider a viable option for UNIX for several reasons. They will not work concurrently with other UNIBUS DMA devices because of insufficient buffering (we mitigate this somewhat through software interlocking on the UNIBUS) and they are small and slow. If you get a system with these in it you can sell a drive and keep the other to have a DEC mass storage peripheral for maintenance reasons. It is currently possible to get roughly $7,500 for a single RK07.

The viable DEC disks run on MASSBUS adapters, and are, except for the RM80, other manufacturers' drive hardware with some DEC electronics added. The DEC disks are significantly more expensive than the alternatives from other vendors. The second-vendor UNIBUS controllers use bit-slice technology and microcode, while most of the older DEC controllers use discrete logic. The following summarizes the cost of the drives available from DEC including two packs for each drive where applicable; sizes shown are formatted space available. First drive prices include a MASSBUS adapter.

Type	Size	First	Maint	Add-on	Maint
RM03	67MB	$30,935	$179	$20,935	$147
RM05	256MB	$45,450	$310	$35,450	$240
RM80	124MB	$29,900	$132	$19,900	$100
RP07	504MB	$48,000	$240	$38,000	$180

The second alternative is to get a MASSBUS† look-alike. Systems Industries will sell you a modified 9400 controller and storage module CDC 256MB removable pack drives or Fujitsu 134MB Winchesters. These setups are less expensive than the same capacity from DEC, but are currently available only for 11/780's as far as we know. We currently do not use this equipment at Berkeley but are aware of several satisfied users running UNIX. For disk I/O intensive systems, having the extra disk channels offered by MASSBUS adapters or multiple UNIBUS adapters is a definite advantage. Costs from Systems Industries including two packs per removable media drive are:

Type	Size	First	Maint	Add-on	Maint
CDC 9766	256M	$38,500	$225	$19,400	$166
Fujitsu*	134M	$22,000	$173	$9,500	$114
CDC 9775	576M	$39,900	$226	$28,500	$167

The final alternative, and the one that we prefer for all but the largest 11/780 systems, is to get a UNIBUS storage module disk controller and some storage module drives. We run most of our disks on the UNIBUS, and have had good luck doing so. Each of our VAXes has an Emulex SC21/V controller with one or more Ampex 9300-CD 253MB drives. The Emulex controller is a single card in the

† DEC considers at least some MASSBUS emulating interfaces to infringe on their hardware patents and are pursuing legal action against selected vendors. We are not clear about the implications of this for users of these devices.

* If you put Fujitsu drives on a Systems Industries controller, the drives normally appear to be multiple RM03's to the system. This causes problems in that the system thinks it can do parallel seeks on drives that actually are pieces of the same drive. Some sites have modified the drivers to alleviate this problem, but these modifications are not part of the standard system. There is also an option to cause the Fujitsu's to appear as single large drives, but the drive type register still says that the drive is an RM03. We are talking to SI about making it possible to discriminate the devices in the software, but this is not yet in the standard release.

UNIBUS box and handles up to 4 storage module drives. We get a better price from Ampex than from CDC, whose 256MB drives seem about equivalent. Emulex and CDC have an agreement for service through CDC organizations of SC21's with CDC drives. We maintain our Ampex drives in house. The Emulex SC-21/V lists for $6,000. Quantity one list price for the Ampex 9300-CD is $14,000. Cables run about $250 per drive. We buy Dysan error free 300MB packs for about $1,000 each (2 for each drive). Thus the total cost for a 253MB drive and its packs is $22,250. Non-discounted CDC prices add about $2,000. Below we show package prices including cables and 2 packs.

Another good storage module drive that is running on VAXes (11/750's) at Bell Laboratories with the SC-21V controller is the Fujitsu 134MB Winchester Drive. We have had extremely good reports on this drive, although we don't have any ourselves. The Emulex storage module packages we suggest are thus:

Type	Size	First	Maint	Add-on	Maint
Fujitsu	134MB	$13,750	???	7,750	???
Ampex	253MB	$22,250	???	$16,000	???
CDC	256MB	$24,250	$300	$18,000	$224

TAPES

We use Emulex TC-11/P UNIBUS tape controllers that list for $3,600 and Kennedy model 9300-3 800/1600 BPI 125 IPS transports at $7,200. You also need a rack to mount the tape drive and cables. An assembled system lists for about $12,000 excluding discounts. The Kennedy transport comes with a 15 month factory warranty. Our distributor exchanges/repairs the cards in the controllers based on a local diagnostic mode in the transport. After the warranty period, card swaps cost about $75. For transport mechanical failures the transport is returned to the factory in Monrovia, California, or we fix it ourselves.

George Goble at Purdue is using a 6250 tape system with UNIX. It includes a Telex 6253 drive (800/1600/6250 BPI) 125 IPS with a TELEX Formatter and an Aviv 1 board UNIBUS interface. The UNIBUS interface has 4KB of buffering, to help with bus latency problems, and it really seems to be necessary. The whole system costs him about $32,000. It emulates a TU10 similarly to the Emulex NRZ/PE controller. 6250 at 125 IPS and uses the entire UBA. UNIBUS access can be arbitrated by the operating system to allow a 6250 BPI tape drive and a disk controller to coexist on one UBA if necessary. N.B.: The driver for this controller/transport combination is not currently included in the standard 4.1bsd system.

Name	Speed	Densities	Cost	Maint	
Kennedy	125ips	800/1600	$12,000	???	
Telex	125ips	800/1600/6250	$33,000	???	(No driver in 4.1bsd)

Our original VAX system came in a package with a DEC TE16 on its own MBA. The TE16 is reliable but slow. The DEC TU45 is faster, but fraught with problems as the high maintenance cost reflects. The DEC TU77 is a good transport, but the auto-loading features don't seem to work well, and it is expensive. Finally, there is a new product from DEC, a 1600/6250bpi 125ips tape drive, the TU78. This is the same transport as the TU77. We do not have a driver for this in the current distribution (4.1bsd), but expect to have one soon. As 6250bpi tapes are desirable in systems with large amounts of disk storage, we expect that this will be an attractive tape drive to include in systems with large amounts of disk storage.

The UNIBUS tape, the TS11, is included in packages for the 11/750 except for the RK07 package system. It does not have a vacuum column, and is thus hard on tapes. It is a problem to load and seems to be unreliable.

Name	Speed	Densities	Cost	Mtce	
TS11	45ips	1600	$13,396	$75	(Not recommended)
TE16	45ips	800/1600	$22,000	$147	
TU45	75ips	800/1600	$25,424	$265	(Not recommended)
TU77	125ips	800/1600	$30,453	$235	
TU78	125ips	1600/6250	$52,000	$340	(No driver in 4.1bsd)

TERMINAL INTERFACES

With a VAX you get 8 lines of DZ-11 that provide some modem control but are not DMA. We use the Able DH-11 emulator, the DMAX/16, that gives 16 lines of DMA output ports without modem control for $4,500, or with modem control for $5,300. We have been getting about 30 day delivery on the these. We are currently testing new products from both Able and Emulex that have lower costs per port and consume less backplane space and power. We expect to finish evaluating these units soon. The current alternative from DEC is more DZ-11's, that cost $2,050 for 8 additional lines, that can be accommodated in the same distribution panel, then $4,300 for each additional 16 lines.

Both the DZ's and the DH's have input silo's that UNIX can use to reduce interrupt load on input. The DMA output of the DMAX is especially important for graphics applications where high-volume and continuous output occurs.

MODEMS

We buy Vadic modem racks for the system end. Vadic offers modems that can talk both standard 300 baud and also the two commonly used 1200 baud styles: Bell 212 and Vadic. It is handy to have these "triple" or "3-way" modems in your machine.

Here are the list prices of various components:

Chassis	$450	(Provides 16 slots)
Power Supplies	$200	(You can use two for redundancy)
3-way Answer	$850	(Takes 2 slots)
Autodialer	$500	(Takes 1 slots, and need 300 and/or 1200 stuff below)
RS232 to RS466	$500	(Converter for autodialer)
300 Orig/Ans	$450	(Takes 1 slot, for autodialer)
1200 Orig/Ans	$740	(Takes 1 slots, for autodialer)

A rack with 5 3-ways and no auto-dialers would cost about $5,100 and would have 6 free slots for additional modems. A rack containing 5 3-way modems, and a 300 baud and 1200 baud dialer would thus have only one free slot, and would cost about $7,250. This equipment has had some infant failure problems.

The Vadic home-end originate answer 300/1200 modems that can also talk to Bell 212 modems are $750 when purchased with Voice/Data switches. These plug directly into modular phones. We have had trouble with initial reliability on these; about 1/3 don't work when received. We are currently buying Ventel modems for home use and reliability seems improved. They too are about $750. These are only two way, 212 std. for 1200 baud and 103 std. for 300 baud but since we have no vadic only answer modems, this is not a problem for us.

We have heard of a new DEC product in modems: the DF03. It is a 300/1200 baud modem using a chip to do the Bell 212 1200 baud standard. It is attractively priced, the DF03-AA is $950 and a unit that provides auto-calling, the DF03-AC, is $1,350. These have been used successfully by the UNIX group at DEC for some months now. We are hoping to try some of these soon.

PRINTERS

We have been using some Printronix 300 line per minute dot-matrix printers. They do point-plotting at 60 points per inch. They are not outstandingly cheap ($7,000), but are ruggedly built.

The new Data Products B-600-1 is a 600LPM band printer. This sells for $7,500 with controller. Although we don't have one of these, Tom Ferrin at UCSF has been using one for four months and is satisfied.

PLOTTERS

Electrostatic printer/plotters that are capable of 200 dots/inch are usable both as printers and as output devices for *troff*. We have an old model Varian that requires considerable care and feeding; newer models are said to be less of a headache. A new Versatec 11" model sells for about $8,000. The objections to these guys are that the paper tends to be wet sometimes, stinky, and more expensive than line printer ($20 per 1000 sheets). These are high maintenance items as are all heavily used hardcopy output devices we are familiar with.*

LOCAL NETWORK INTERFACES

We are currently getting experience with an Ungermann/Bass NET/ONE local network interface, which uses an Ethernet-technology coaxial network. We are also getting some DMR-11 interfaces from DEC, that allow point-to-point networking at up to 1Mbit/sec. The older DMC-11 interfaces can be made to work at high speeds only by modifying the hardware and should probably be avoided.

We are hoping to learn much more about local network products in the next few months and would like to hear any UNIX user experience with different commercially available devices.

* Note added in proof: some new laser printers have recently been announced. There are at least two devices now commercially available: one from Canon lists for about $23,000 and comes with either RS232 or DR-11 interfaces, and another we have heard of lists for about $8,000 in small quantities; look in recent DATAMATION issues for details.

SYSTEM PACKAGES

We now present some sample system packages. Each represents a balanced system for timesharing use under UNIX. People often ask us how many users can be supported under UNIX in these configurations. In the absence of specific information about applications to be run this is an unanswerable question. The amount of load presented to the system by different applications varies widely. We mention with each system the count of interactive users typical of our university research environment that could be supported by the system.

We first present systems based on 11/750s and then systems based on 11/780s. With each system we suggest similarly functional systems configured in at least two different ways: first with as much equipment as possible from DEC and second with the best equipment known to us.

Various people have measured the speed of 11/750's and 11/780's and found that the 11/750 executes at roughly 60 percent of the speed of an 11/780. By comparison, an 11/70 runs at roughly 75 percent of the speed of an 11/780 on the same benchmarks, that involve no floating point, no 32 bit arithmetic on the 11/70 and also no system calls. For time sharing UNIX usage we believe that the 11/750 should have better performance than an 11/70. The additional tuning and performance enhancements to the VAX kernel and larger address space of the VAX provide the boost for the 11/750.

The first system we consider is a small 11/750. We are fond of the 11/750, as it is cost effective for small systems. The problems with starting with an 11/750 are that it is not as expandable as the 11/780, and that there is currently no hardware floating point unit for the machine. We present an expansion of the 11/750 into a larger system.

The second base system is a small 11/780. We show how it can be built from a DEC RM80/TU77 package system, and two different ways to build it from a DEC RK07 package using the best equipment that we are aware of, emphasizing either lowest cost or capacity for expansion to a system with high i/o bandwidth.

The small systems we suggest start with a single disk and tape controller and some memory. For time-sharing applications we configure our VAX systems allowing 256k bytes of memory for the kernel and roughly an additional 100k bytes of memory per active user.* Memory is cheap, especially for the 11/780, so we don't skimp on it.

With more than a few users, it is critical that more than one disk arm be present in the system. Thus all but the smallest systems include more than one disk. As the active user count rises, having more than one disk controller is also a good idea. The large system packages include more than one disk controller. For large and i/o intensive systems we recommend the fast new RP07 drive from DEC as it runs at 2.2 MB/sec, twice as fast in burst rate as the other drives; this requires running with interleaved memory.

It is desirable on all UNIX systems to have at least 100MB of disk space so that all the source for the system and all the standard programs may be kept on line with some room for locally developed programs. The amount of space required by user programs varies per installation; we manage to run many of our instructional/research machines using about 300 megabytes of space actively, although slightly more than this would be desirable.

* These numbers work reasonably well in an environment typical of University work (course work, paper preparation, debugging programs, developing applications for microcomputers, etc.) More demanding applications could require substantially more memory per user.

Our large research machine runs with 1 Gigabyte of disk storage, with 2 disks on a UNIBUS and 2 disks on MASSBUS adapters. The weakest point in this system is that it has only a 45ips TE16 tape drive for backups. For even the smallest systems, 45ips will soon seem slow. We therefore recommend starting with a 125ips 1600bpi tape drive. As full 2400 foot tape reels hold only 30MB at 1600bpi, large systems would be a good idea to include a 6250bpi tape drive.

Finally, in all but the smallest systems we include modems and printers. 300 baud modems are painfully slow, so all our systems include 300/1200 modems. Given an auto-call unit you can hook your machine into a loosely coupled network of UNIX machines called the *uucp* network. As such auto-call units are inexpensive, adding one can be a valuable way of tying into the UNIX community.

VAX 11/750 PACKAGES

We want to put together a small 11/750 system capable of supporting about 8 time-sharing UNIX users, and a larger 11/750 system for roughly 16-24 users. We need a minimum of 100 megabytes of space for the small system and a reasonable tape drive, preferably a 125ips drive so that tape operations can be done in a reasonable amount of time; if the system is to include only non-removable disks, we consider the faster tape system to be important. For the larger system, we wish to add disk space to give the system a minimum of 250 megabytes of space, and have more than one disk arm.

Small system

When buying an 11/750 the most difficult problem is the sales packaging of the systems. DEC currently packages only 3 11/750 systems, and we are not happy with the tape equipment in any of these systems. The small DEC system includes 2 RK07 disk systems. These are not adequate to support our desired configuration, since they supply only 28 megabytes of space each. To build a system from all-DEC equipment we would thus start with the the RM80/TS11 configuration, SV-BXWAA-AA; this lists currently for $120,000, less $5,000 for VAX/VMS license only. The TS11 tape drive is a 45ips 1600bpi only tension arm tape drive, and does not meet all the requirements for our system, but is the only tape drive available from DEC for the 11/750. We would prefer to have a 125ips tape transport but the only 125ips transports available from DEC are far too expensive to be reasonably included in this system, even if DEC sales packaging rules would let you buy them.

You can consider building a system using equipment from vendors other than DEC. You can buy the cheapest currently available packaged configuration, the SV-BXHHA-AA dual RK07 system; this lists for $89,900 less $5,000 for dropping VMS support, or $84,900 list for most UNIX users.[9] This package represents the most cost effective way for non-OEMs to currently purchase a CPU, although the RK07's are not useful. A single RK07 drive is required for DEC field service. The second drive lists for $12,000, and if you can sell it for $7,500, you can bring the foundation cost down to about $77,400.

[9] If you qualify as an OEM, you may be able to buy a bare CPU for about $48,000 list from DEC; we have not been able to buy the naked CPU even given the quantity of hardware we buy from DEC.

To this we would then add:

0.5M Trendata memory	1 MB total	$4,600
Emulex SC-21V/Fujitsu	134 MB	$13,750
Emulex TC-11P/Kennedy 9300	125 ips	$12,000
Total starting from DEC RK07 package		$107,750

Thus, summarizing the possibilities, we can start by buying an RM80/TS11 system; this will give us 124M bytes of disk space and a 45ips tension arm drive for $115,000 list price or we can buy a RK07 system, sell a RK07 drive, and add a Emulex disk controller and a 134M bytes Fujitsu drive and an Emulex tape controller and a 125ips Kennedy tape drive, to a total cost of $107,750.

Medium system.

To expand this basic system to support more users and get additional disk space, we would add additional lines, disk storage, modems and memory. To add this equipment to the small all-DEC system we would add another RM80, another Megabyte of memory and a DZ-11E:

RM80/TS11 base system		$115,000
MS750-AC 1MB memory	2MB total	$19,800
RM80-AA 124MB disk	268MB total	$19,900
DZ11-E 16 lines	24 total	$4,300
Vadic Modems	5 300/1200	$5,100
Data products B-600-1	600lpm	$7,500
All-DEC Medium 11/750 cost		$171,600

To make this system from the RK-07 package using non-DEC equipment we would add the following equipment to the small system:

RK07 base system		$107,750
1M Trendata memory		$9,200
1 134M Fujitsu disk drive		$7,750
Able DMAX	24 lines total	$4,500
Vadic Modems	5 300/1200	$5,100
Data products B-600-1	600lpm	$7,500
Add-on cost		$33,700
Total starting from DEC RK07 package		$141,450

The differences in the hardware functionality between the two medium 11/750 systems are: The DMAX is a DMA device more suitable for high-speed output, while the DZ11 is interrupt per character. The system not all from DEC includes a 125ips vacuum tape drive while the DEC system includes a 45ips tension arm drive. The all-DEC system has a MASSBUS disk drive, on a different bus than the tape. The non-DEC system includes a single RK07.

VAX 11/780 PACKAGES

For a system with more growth possibilities than an 11/750, faster processing (especially floating point), and higher i/o bandwidth, we recommend starting with a small 11/780. Our goal here is to start with a system capable of supporting 8-16 timesharing users and expanding the system to be capable of supporting rougly 24 users. We also consider a large expansion of this system, to a system that might support 32 to 40 terminal users to the exhaustion of available CPU cycles.*

Small system

For our small system we require 100 Megabytes of disk storage and a 125ips 1600bpi tape drive that will be capable of handling file backups if the system is eventually expanded. In our first expansion of this small system, we wish to add to the available space to a minimum of 250 Megabytes of disk storage, acquire at least two disk arms, and add additional terminal lines, a printer and modem equipment for dial-in lines. In a large expansion of this system we include more terminals, an additional disk controller to get at least two separate disk channels, and an additional 500 Megabytes of storage for a minimum of 750 Megabytes.

To build a small system from all DEC equipment, we would start with the RM80/TU77 based system, the SV-AXWBA-CA. This system includes 8 terminal lines, 1 Megabyte of memory, a 124 Megabyte disk drive and a 125ips tape. The system lists for $225,000, and we can save $5,000 by getting a system without VAX/VMS support making the basic all-DEC system list price $220,000. To this we would add .5 Megabyte of memory, the MS780-DB for $13,900, and a floating point accelerator for $10,600 to bring the total all-DEC system price to $244,500.

As we did for the 11/750, we use an RK07 based system for the nucleus of our system built using second-vendor equipment. We start with the SV-AXHHV-CK package consisting of:

- 11/780 CPU
- 512K memory
- 8 lines of DZ-11
- LA120 console
- 2 RK07 Disks and UNIBUS Controller
- VAX/VMS license only

This system lists for $138,000. We would immediately sell a RK07 drive. If we get $7,500 for this drive and add a floating point accelerator we get a list price for the equipment from DEC of $141,100.

* Using systems similar to the largest shown here, in an environment consisting of small student programming some sites have reported running up to 70 interactive users; cpu cycles are the critical resource with this many users.

To form a system with the performance we desire, while minimizing the cost, we make the following package:

RK07 base system		$141,100
1M memory	1.5MB total	$5,000
Emulex SC-21V/Fujitsu 134MB	172MB total	$13,750
Emulex TC-11P/Kennedy 9300	125ips	$12,000
Add-on cost		$32,750
Total starting from DEC RK07 package		$173,850

The list price difference between these two systems is $55,250; much of the additional cost in the all-DEC system is in the two MASSBUS adapters for the RM80 and the TU77. The bandwidth of these adapters is not critical in this small system. Also, the TU77 is an autoloading tape drive, while the Kennedy 9300 is not. We do not consider these differences significant.

To form a system with the emphasis on handling of data-intensive applications, and to emphasize total growth of the system, we would add an REP07-AA 504MB disk drive and MASSBUS controller to the basic system instead of adding the Emulex SC-21V and Fujitsu drive. To do this, however, we must expand the physical packaging of the cpu package that we based the system on.

We want to add another MASSBUS adapter that there is no space in the CPU cabinet for. There are two possible ways to solve this problem. The first is to rearrange/upgrade the CPU by removing the built in UNIBUS backplane and adding an extender cabinet buying the component parts individually. The second involves purchasing these same parts in an *Upgrade Kit* from DEC. This kit and its component parts, as we are aware of them, are described below:

DEC Part No.	Description	DEC List
H9602-DF	UNIBUS Expander Cab.	$3,360
BA11-KE	UNIBUS Expansion Mounting Box	$3,400
DD11-DK	9 Slot UNIBUS Backplane	$900
	Misc. Cables	~$500
Components Total		$8,160
H9604-BA	DEC Upgrade Kit	$25,000
Unaccounted for Cost		$16,840

Many phone calls to DEC have not produced any explanation of the unaccounted for $16,840 or any reason that you should not simply buy the component parts. On the other hand, our sales representative will not *guarantee* that a system configured with separately purchased parts will work. Our reluctant conclusion is that the missing $16,840 is part of DEC's marketing strategy aimed at discouraging building up large systems from an inexpensive base.

We are planning to build two systems this summer using the component parts hereafter referred to as the *Berkeley Upgrade Kit*. Since we have not yet done this and cannot *guarantee* they will work, you should add the $16,840 cost to these parts to be most conservative about the system package we are suggesting.

Thus to form a small system emphasizing expansion we add:

RK07 base system		$141,100
Berkeley upgrade kit		$8,160
1M memory	1.5MB total	$5,000
DEC REP07-AA	504MB total	$48,000
Emulex TC-11P/Kennedy 9300	125ips	$12,000
Add-on cost		$73,160
Total starting from DEC RK07 package		$214,260

Medium system

To expand this basic system to support more users and get additional disk space, we would add additional lines, disk storage, modems and memory. To add this equipment to the small all-DEC system we would add another RM80, another Megabyte of memory and a DZ-11E:

RM80/TU77 base system		$244,500
MS780-DC 1 Megabyte memory	2.5MB total	$19,800
RM80-AA 124 Megabyte disk	248MB total	$19,900
DZ11-E 16 lines	24 lines total	$4,300
Vadic Modems	5 300/1200	$5,100
Data products B-600-1	600lpm	$7,500
All-DEC Medium 11/780 cost		$301,100

To make this system from the package using inexpensive non-DEC equipment we would add the following equipment to that small system:

RK07 derived base system		$173,850
1M memory	2.5MB total	$5,000
Fujitsu 134MB disk	296MB total	$7,750
Able DH/DM	24 lines total	$4,500
Vadic Modems	5 300/1200	$5,100
Data products B-600-1	600lpm	$7,500
Inexpensive medium 11/780		$203,700

The differences in the hardware functionality between the two systems are:

1) The DH/DM is a DMA device more suitable for high-speed output, while the DZ11 is interrupt per character.
2) The tape drive in the all-DEC system is autoloading.
3) The DEC system has MASSBUS disks and tapes, while the non-DEC system has UNIBUS peripherals. We do not consider this to be a serious problem for a system of the size considered here.

Finally, to put together a system with maximum growth possibilities we would take our small growth system and add an RP07-AA instead of a Fujitsu drive; giving the following breakdown:

RK07 derived base system		$214,260
1M memory	2.5MB total	$5,000
RP07-AA 504 MB disk	1036MB total	$38,000
Able DH/DM	24 lines total	$4,500
Vadic Modems	5 300/1200	$5,180
Data products B-600-1	600lpm	$7,500
Growth medium 11/780		$274,360

Large system

To build a large system from the medium system we add additional memory and more disk storage.

To the all-DEC system we would add an RP07 504 Megabyte disk drive, 1 Megabyte of additional memory, and 16 additional terminal lines:

Medium RM80/TU77 system		$301,100
MS750-DC 1 MB memory	3.5MB total	$19,800
RP07-AA 504 MB disk	752MB total	$38,000
DZ11-E 16 lines	40 total	$4,300
All-DEC Large 11/780 cost		$363,200

We would put the RP07 disk on the same MASSBUS adapter as the TU77 tape drive, leaving the two RM80 disks on the other MASSBUS adapter.

To build a similar system from our inexpensive medium system we can either add an RP07 to the system, or add another UNIBUS adapter and get more redundancy by putting in another Emulex disk controller and two more drives; if either UNIBUS or disk controller were to fail, this system could run all disks on one controller on one UNIBUS.

To build a UNIBUS based large system we add an upgrade kit, an additional UNIBUS adapter, an additional megabyte of memory, a second Emulex disk controller, two 253MB AMPEX 9300 drives, and an additional 16 terminal lines:

Medium inexpensive system		$203,700
Berkeley Upgrade Kit		$8,160
Additional UBA	2 total	$12,300
1MB Memory	3.5MB total	$5,000
Emulex SC-21V/Ampex 9300-CD		$22,250
Ampex 9300-CD(add on)	802MB total	$16,000
Able DMAX	40 lines total	$4,500
Total large UNIBUS system cost		$271,510

To make the large VAX system adding a RP07 and a MASSBUS adapter instead of two UNIBUS adapters, we add an REP07-AA 504MB disk drive and MASSBUS adapter instead of the additional UBA, Emulex controller and AMPEX disk.

Medium inexpensive system		$203,700
Berkeley Upgrade Kit		$8,160
1 MB Memory	3.5 MB total	$5,000
REP07-AA 504 MB disk	800MB total	$48,000
Able DMAX	40 lines total	$4,500
Total large UBA/MBA system		$269,360

Both of these large systems include multiple disk controllers: in the all-DEC system both MASSBUS adapters are acting as disk controllers; in the all UNIBUS system, there are disk controllers on both UNIBUSes. In the UNIBUS/MASSBUS system there are disks on both. The differences between the all-DEC and the not all-DEC systems remain: DMA terminal i/o is lacking in the all-DEC system, and the systems that are not all-DEC have removable media: the RK07 and also the Ampex storage modules in the all UNIBUS system.

It may be desirable in a system of this size to have a 6250bpi tape drive. We have a system with 1 Gigabyte of online disk storage that has only a 45ips 1600bpi tape drive to do backups, and find them painful. With a 125ips drive we think that the backups would be tolerable, and given our lack of experience with the 6250bpi drives do not include one here.

With a system as large as these it may be desirable to have a second memory controller and to interleave memory. We have a second memory controller on the premises and intend to experiment with this soon. To run an RP07 at full speed a second memory controller is necessary. Systems with two memory controllers and interleaved memory are discussed next, in examining VAX systems for data intensive applications.

Thus we consider our final two systems, that flesh out the growth versions of the system. First we add memory and more terminals to build a system comparable to the other large systems, and a 6250bpi DEC TU78 controller for two reasons: to provide high speed backup, and to put a RP07 on its MASSBUS, since we as yet have only one disk channel.*

We form this system as follows:

Medium expansion system		$274,360
1 MB Memory	3.5 MB total	$5,000
TEU78-AA Tape	6250bpi	$52,000
Able DMAX	40 lines total	$4,500
Total large expansion system		$335,860

Recall that this system has two tape controllers and still has an RK07 controller and single drive; these could be sold now, since the RK07 was kept only for maintenance reasons and the the tape drive is replaced by the TU78.

* Note that there is no TU78 driver in 4.1bsd, although we expect to have one reasonably soon.

Finally, to expand this system to its full capacity and to make it possible to run the RP07's at 2.2 MB/sec, we add an additional memory controller to the system. We also flush out the system to the full 8MB of memory:

Large expansion system		$335,860
MS780CC mem ctlr	.5MB	$26,600
H9602-HA expander cab		$4,460
4 MB Memory	8 MB total	$20,000
Total full expansion system		$386,920

SYSTEM SUMMARY

In the summary that follows, "All-DEC" means that all mass storage, memory and terminal interface devices are DEC supplied; all systems include second vendor modems and printers.

Type	CPU	Mem	Disk	Tty	Tape	Modems	Lpr	Price	Ratio	"All DEC?"
Small	750	1M	124M	8	45ips	None	No	$115,000	1.00	Yes
Small	750	1M	172M	8	125ips	None	No	$107,750	0.94	No
Med	750	2M	248M	24	45ips	5	Yes	$171,600	1.00	Yes
Med	750	2M	306M	24	125ips	5	Yes	$141,800	0.83	No
Small	780	1.5M	124M	8	125ips	None	No	$244,500	1.00	Yes
Small	780	1.5M	172M	8	125ips	None	No	$173,850	0.71	No
Small	780	1.5M	532M	8	125ips	None	No	$214,260	0.88	No
Med	780	2.5M	248M	24	125ips	5	Yes	$301,100	1.00	Yes
Med	780	2.5M	306M	24	125ips	5	Yes	$203,700	0.68	No
Med	780	2.5M	1032M	24	125ips	5	Yes	$274,360	0.90	No
Large	780	3.5M	752M	40	125ips	5	Yes	$363,200	1.00	Yes
Large	780	3.5M	602M	40	125ips	5	Yes	$271,910	0.77	No
Large	780	3.5M	1032M	40	2*125ips	5	Yes	$335,860	0.92	No
Large	780	8M	1032M	40	2*125ips	5	Yes	$386,920	1.06	No

CONCLUSIONS

We have presented sample VAX systems over a wide performance range using both all-DEC and the best available second vendor equipment, emphasizing independently minimal cost and maximal expandability. Use this information wisely; price shouldn't always be the bottom line.

Consider the all-DEC system if you can afford it. If not, the second-vendor equipment in the packages here is all thought to work well on VAX hardware. You can reliably operate such a system if you can get maintenance for such equipment in-house, or through a organization associated with the vendor, or can stock some spares for non-redundant equipment.

VENDOR REFERENCES

Manufacturer	Product	Phone	Vendor contact
Able	Async. Mux	(714) 979-7030	Able Computer (Norm Kiefer)
Ampex	Disks	(408) 255-4800	Ampex (Jess Clark)
Aviv	Tape controllers	(617) 933-1165	John Connolly
CDC	Disks	(415) 820-2933	Eakins Associates (Joe Voelker)
Data Products	Printers	(415) 948-8961	MQI Associates (Avery Blake)
Dysan	Disk Packs	(408) 730-2145	Dysan (Sandy Foss)
Emulex	Controllers	(415) 820-2933	Eakins Associates (Joe Voelker)
Fujitsu	Disks	(415) 820-2933	Eakins Associates (Joe Voelker)
Kennedy	Tape Transports	(408) 245-9291	Electronic Marketing Specialists
National	Memory	(800) 538-8514	Don Johnson
Printronix	Printers	(408) 245-4392	Group III Electronics (Paul McCann)
Systems Ind.	Storage Systems	(408) 732-1650	John Leavitt
Trendata	Memory	(714) 540-3605	Trendata (Mike Jacobs)
Vadic	Modems	(408) 727-6491	Moxon Electronics (Lee Berlind)
Varian	Plotters	(408) 733-2900	(Ted Downs)
Ventel	Modems	(408) 279-8711	Backus Data Systems (Jeff Masi)
Versatec	Plotters	(408) 985-5810	(Jim Muszalski)

Software Support
of
Ubiquitous UNIX
on a
Multitude of Machines

Dave Horsfall
Computing Services Unit
University of NSW

(dave:unswcsu)

The above alliterative title refers to the author's task of supporting UNIX on a variety of machines around campus. This version of UNIX is basically V6, with local AUSAM extensions, pieces of PWB & V7, and other local modifications. The machines include six PDP-11/40's, two LSI-11/23's, three PDP-11/34's and a PDP-11/60; and are referred to as CIVIL, COMM40, COMM34, MECH, MATHS, CSU40, CSU60, LIBRARY, FOOD (School of Food Technology), PGES (Post Graduate Extension Studies), MANLY (Manly Vale Water Research Laboratory) and PHARM (Physiology & Pharmacology). In addition, a PDP-11/44 and a PDP-11/34 at Prince Henry Hospital will shortly come under the umbrella. The intention was for each of these sites to appear to be the same to the user, despite the fact that no two of them have the same configuration; and this implies having all systems generated from the one source data base. This data base is on CSU40, and various techniques are used to keep all the systems compatible. These techniques will be described for the benefit of other users who may find themselves in this situation.

It will be seen that particular circumstances dictated the particular methods used. Great emphasis is placed on _MAKE_, since modifying one source program will automatically invalidate all binaries of that program. Separate directories are used for each site, and the source is linked into each directory. All programs are compiled with their own _include_ directory, thereby establishing such things as site names, local hardware idiosyncracies etc. These directories are merely sub-directories of _/usr/include_, with their contents for the most part linked back to the parent directory. Only files such as _local-system_ are private. A run file in each site's source directory executes a common _makefile_ with a parameter passed to the C compiler specifying this _include_ directory.

Unfortunately, the current version of _MAKE_ does not allow dependencies of object files on SCCS files to be specified in a simple manner, since suffix transformation rules are used and SCCS usage involves adding an `s.' prefix. Various methods are used to get around this. The V7 version of _MAKE_ does allow this, but is yet to be implemented on V6 - if ever.

We turn now to a discussion of the techniques used, followed by the method by which the systems are kept up to date. Samples of the _makefiles_ are not included here, but their contents are rather obvious.

The first example is the resident kernel source arrangement. Here, all sites have their own directory tree, and the dependancies of the objects are specified explicitly to the common SCCS files. Although the compiled objects are kept, the source files are not; they are merely the result of an intermediate step - that of "getting" the file. Provision is made however for compiling and linking the system to test new modules without going through SCCS. In this case the source is retained for further modification. Makefiles in each directory set up the conditional compilation flags for local hardware differences, such as FPU, PAGEDKERNEL, STACKLIMIT etc. It is arguable whether these symbols would be better off kept in an include file, but it was done this way for historical reasons. In addition, most drivers include a file specifying the local site configuration, and various data areas are defined here. For example, the line printer driver will include a file conf.lp which would be unique to each site. This ensures that changes to the local configuration do not affect the driver proper - they have nothing to do with it. The makefile dependancies will ensure that the code gets recompiled. Another example is the KL driver - if dial-up lines are installed, or if it is driving a DLV-11E/F (on the LSI-11/23) then symbols must be defined to include the code to handle data-set interrupts and baud-rate changes respectively.

The second technique is the primary non-resident source - the shell, editor etc. Since there is very little difference between the various sites for this type of source, there are no separate directory trees. Instead, programs are merely "gotten" from the SCCS file in the main directory, and the makefile run, specifying the include directory on the command line by hand. Few programs need to have a specific include directory specified; they are GETTY, WALL and WRITE to get the local system name; and LPR and LPD to get the local printer configuration. Neither the objects nor the sources are kept since this type of source is basically static in nature, and usually consists of the one source module. Compare this situation with that of the resident kernel.

A third technique is used on sources such as the SUN network. Much the same method is used here as for the resident kernel, except there are no SCCS files, since the source is basically maintained by foreign users. Each subdirectory has the source linked into it as before, and a run file (called Make) merely executes the common Makefile with an argument specifying the include directory along with the rest of the argument list. One can then type "Make" instead of "make". Compiled objects are kept for the CSU40 system, but not for the others, since they only get built when the source has stabilized at CSU40.

Maintaining the source code is one thing, but ensuring that the sites are kept up to date is quite another. Currently, use is made of the OSCAR program (I've long since forgotten what the acronym stands for) to provide a list of all programs changed on a particular file system since the modification date on a "marker" file. The file systems specified are the root and the source. Most changes are picked up on the root file system, whereas the online manual is kept on the source file system. All sites have their own marker file, which is "touched" when a distribution is made to that site. Unfortunately this does not show which files have been deleted, only those that have been altered. This can be circumvented by keeping a list of all files on a distribution and executing COMM(1) to generate a list of deleted files, but it isn't worth the trouble of maintaining this list. Instead, every so often a fresh distribution is provided of the root system and the manual to all the sites, which effectively eliminates any rubbish that may have accumulated. It also has the beneficial side-effect of removing any "tampered" programs that may (and do) appear. I consider that this outweighs the disadvantage of the sites having to recover any programs that they may have installed. Of course, things like /etc/passwd, /etc/ttys and /etc/rc are saved first.

I hope the above may be of some use to UNIX gurus. These techniques were developed over a period of about three years, and have undergone considerable change. Further information such as directory listings, makefiles etc. may be obtained on request, or alternatively they can be inspected by anyone. I would be pleased to receive any comments, criticism etc of these techniques, since they are not as elegant as they could be. For example, the only way to generate a distribution from scratch is the brute-force technique of taking a copy of the CSU40 root file system and deleting by hand all the things not needed. Any mechanized technique would seem to imply some sort of list of files to copy, and this list would then become tedious to maintain on a per-site basis.

AUSTRALIAN UNIX USERS GROUP MEETING AT AGSM
March 16, 1981

FINANCIAL STATEMENT

INCOME

47 Registrations	470.00	
Cheque frcm Melbourne International Meeting	300.00	
Interest from Account S47069 (CSB at UNSW)	2.56	
TOTAL		$772.56

EXPENSES

2 Refunds (Non-attendance)	20.00	
AGSM Charges (Meals + Theatre)	337.50	
Bank charges (3 Cheques)	0.90	
Cheque to Brisbane Meeting	414.16	
TOTAL		$772.56

Lindsay Harris

Lindsay Harris
AGSM

From TELEMAIL Mon Apr 20 22:33:55 1981 remote from usa netmail from basser40
From ianj Mon Apr 20 22:32:48 1981 remote from mhtsa

pete,
welcome to the wonderful world of international mail !!
a bit premature BUT}i it is working
see important messages for you at waterloo
ian

From dave Wed Apr 22 10:01:21 1981 netmail from unswcsu
Subject: Organ folders

Hey mate - how about some sort of offer for folders for the various isssues of AUUGN that must be cluttering up everyones bookcase by now ?

From bob Thu Apr 23 22:48:45 1981 netmail from basser40

The following news item came from the U.S.A. Unix News net via Ianj and more networks than you can poke a stick at!
Bob Kummerfeld

######modem.news

This month marks the twentieth anniversary of the first appearance of the word "modem" in computing literature.

The word was inadvertently contributed to the language by W. W. Zysick, a Bell Labs engineer, in a paper that appeared in the April 1961 BSTJ. There, Zysick described "a modern device that facilitates the connection of automatic computing machinery to voice-grade telephone lines". When set in type, the slugs for "r" and "n" in "modern" ran together, producing "modem".

Unfortunately, shortly after submitting his manuscript to the BSTJ, Zysick was killed by a runaway fork-lift truck while touring WECo's Submarine Cable Shop in Baltimore. Thus, he never got to see the proofs, and the error went undetected to press.

Subsequently, a number of workers citing Zysick's paper independently "recognized" modem as an acronym for "modulator-demodulator", and dropped "device" as superfluous, thereby establishing the word in the language.

From dave Thu Apr 23 15:14:52 1981 netmail from unswcsu
Subject: computing practices

How about a page or so in AUUGN dedicated to tricks & hints that users may wish to pass on to their fellow hackers ? You could call it say "Computing Practices" as la CACM (if they don't sue), or perhaps "Qwik Tricks", "Dirty Deeds" etc. I know that Kev is anxious to pass on some of the tricks he knows with the Editor e.g. "gwa" to make multiple copies of a file (I think), or "gm0" (I think) to reverse the order of lines in a file! I myself find "<address,address>o/$/" useful

to compile a columnar list of things - you just enter in the next
line then hit ESC, then do it again for each column. Much less
tedious than typing it out in full. Perhaps we could call it
"Editor Tricks" with the disclaimer that not all editors may accept the
command.

From mhtsa!pwbcc!andrew Wed Apr 22 11:12:57 1981 remote from USA netmail from basser40
G'day mate:

just testing the new mail service.

andrew

From piers Wed Apr 22 19:45:43 1981 netmail from basservax
To: peteri:elecvax auugn:elecvax
Subject: USA - AUST link up?

From TELEMAIL Mon Apr 20 22:27:37 1981 remote from usa netmail from basser40
From ianj Mon Apr 20 22:23:43 1981 remote from mhtsa

piers,
welcome to the wonderful world of international MAIL
!!

this very evening I got my program going to pick up
those uucp files destined for australian distribution.

this mail was spooled by the local mail command, then
picked up by my new TMAIL program and collected into
one file mail.apr20 and then sent to TELEMAIL via TM.

what more can I say

cheers ian

From kre Sat Apr 25 05:24:44 1981 netmail from basservax
To: peteri
Subject: auugn ...

I think puting mail directly in AUUGN as it comes to you
is a good idea. Keep it up.

I also want to correct a couple of mis-statements I must
have made in my talk at the UUG meeting in March.

1) Vfork is not yet dead. Copy on write doesn't work on Comets
for some reason relating to their micro-code bugs. The system
has to work on those machines too, so vfork will stay till
the bugs are removed.

2) When copy on write is implemented, it will of course be
necessary to turn off write permission in the data & stack for both
parent & child, noyt just the child. It is also complicated

by multiple forks before anyone writes in the page ...
It would also be desirable to duplicate the bottom few stack
pages, as these are certain to be modified soon.

3) the better i/o library relates to F77.

4) A new tar is not new of course. It just has an improvement
(it keeps owner/modes of directories). Tapes produced by
this tar are entirely compatible with those of the standard
tar, except that if read by a standard tar, there will be
a diagnostic when the directory is attepmted to be created
for the second time (standard tar creates it twice if it
sees a directory on the tape) - Not really a "new tar"

I am also pleased to note that the UK people seem to
have done the sensible thing & turfed out the Bell tty driver
& created a useable one.
However I don't think much of making UNIX pretend to be a DEC
system, with all its crazy conventions. ^D for eof is quite
good enough for me (naturally it should be settable so
individual users can pick what they like - but leave the defaults
(other than '#') alone!)

From mhtsa!ianj Fri May 1 15:13:30 1981 remote from USA netmail from basser40
thanks for auugn
looks good
yet again

you still haven't mentioned (as far as I can tell)
that
1. unix 3.0 is to be licensed to outside world - get letters in soon
2. irma biren is no longer in charge of unix distributing ..
   so people should just write to computer information librarian ..

also
what are you travel type arrangements for texas ?

glad to see you and wally wedel have got in touch..
euug looks well organised ..

ian.

From mhtsa!ianj Tue May 5 23:11:08 1981 remote from usa netmail from basser40

```
                         IIIII BBBB  M    M
                      I    B    B MM MM
                      I    BBBB  M M M
                              I    B    B M    M
                    IIIII BBBB  M    M
```

New Updates and Fixes to IBM/VU

This is a notification of two patches to IBM/VU (Virtual
Universe) and a new release designed for users not requiring the full

capibilities of a virtual universe.

Several users have reported errors resulting from recursive calls to the Universe Creation Utility (UCU). This utility, called from IEBSAGAN, is used to initialize the virtual space which will hold the universe to be simulated. On occasion, the universe created by this routine will contain technologies capable of creating their 'own' virtual universe processors, which in turn call on the UCU. While the stack structure supporting the UCU was designed with this in mind, no system can handle unlimited recursion. Release 134 will contain a patch that will request user verification before a new level is created.

Another problem that has been experienced occurs during the use of black holes and neutron stars in the virtual universe. Although the mass storage media provided with the VU processor is of the highest quality, it can not handle storage at such a density. The highest density that is supported is 2.32E16 grams/cc. DO NOT EXCEED THIS LIMIT. Severe gravitaional effects have serious impact on the reliability of the system. V134 will also contain a program (IEBFORWARD) that can deal with these problems.

Due for release in April is IBM/VSS (Virtual Solar System). This is designed for the casual user, who does not require the full use of a universe. One possible use of this package is to provide uniform testing conditions for programs that take input regarding the phase of the moon. In addition, it is estimated that the cost of simulating the solar system and a Voyager flyby is 1/5 that of actually running such a mission.

IBM Software Division

From mhtsa!ianj Tue May 5 23:11:31 1981 remote from usa netmail from basser40
O listen all you hackers, to my sad tale of woe
I didn't check for table size when slots there were no mo'

I spent much weary looking, for esoteric quirks
Til finally the bug I found, and oh it really irks

For if at first I'd done it right, it wouldn't have been bad
I'd just have recompiled it once, and now would not be sad

So listen hackers, listen well, to my poor worthless rhyme
Do it good the first time through, and save yourself some time.

From mhtsa!ianj Mon May 4 20:20:38 1981 remote from usa netmail from basser40
Subject: bricks and watermelons...

An explanation of why digital communications is the way of the future (according to a recent article in Fortune magazine):

> ... more of the square-shaped digital signals can be transmitted over the same channel than analog signals, for the same reason that bricks, for example, can be fitted into a given space more compactly than watermelons ...

From root Mon May 18 14:00:35 1981 netmail from elec70

Dave: re cpio, the insistence on reading a list of file names from stdin is even worse when you realise how incredibly slow "find" is! What I find particularly revolting about cpio is that the end of a cpio dump is indicated by a zero-length file called "TRAILER!!!" - thus if someday some cretin creates one of these in his directory it will totally wreck the dump. There are a thousand better ways to indicate the end of a dump - one that pops immediately to mind is to store an illegal inode number (inodes are stored on the tape too (numbers that is) and it is easy to show that it is impossible to use inode numbers 0177771 to 0177777 because they wrap around). This is all then capped off by filling out the last block with whatever is left in the buffer (I hate such sloppiness!).
Re initsh: yes well you know how it goes.... I agree it is a pain, but the idea with catching @ and backspace was that a person might put these in his/her password thinking all was OK, when in fact all was not. I guess it is redundant given that initsh forces the password to be longer than 8 chars anyway.... Given the way it is fired up, it assumes that it will be followed by a getty (it does leap in and alter initial shells for example).
It does tend to ramble on, but then some students here are pretty thick.
Re iomove&mem: apart from making reads/writes on mem a zillion times faster (ps really flies down here now compared to how slow it used to be), it also had the desirable effect of make word operations do word operations and not behind-the-scenes byte operations. It does assume that you are using a mapped_buffers system i.e. ka5 is free and clobberable without restoring. If you are not using mapped_buffers than the changes you should make are obvious. You should also look at iomove - I altered it slightly to be more efficient on certain types of odd-byte reads/writes (there tended to be code in various drivers to iomove up to an even number and then do the last byte using cpass or passc - I moved this code to where it should have been in the first place!)

kev

From root Wed May 20 15:24:58 1981 netmail from basservax
To: gurus
Subject: CSIRO node name change

Please inform any CSIRONET users that the basser40 node name will change from "*mid" to "*suu" in about 2 weeks.

From mhtsa!ianj Wed May 27 16:25:53 1981 remote from usa netmail from basser40
Almost anything in software can be implemented, sold, and even used given enough determination. There is nothing a mere scientist can say that will stand against the flood of a hundred million dollars. But there is one quality that cannot be purchased in this way -- and that is reliability. The price of reliability is the pursuit of the utmost simplicity. It is a price which the very rich find most hard to pay.

C. A. R. Hoare

From judy Sun May 31 21:50:14 1981 netmail from basservax
To: ianj:usa piers chris peteri:elecvax jefft:elecvax
Subject: more Hoare

"Of course there are a lot of mistakes made in every branch of engineering, and one of the attributes of a good engineer, I hope, is that he avoids obvious mistakes.

Now there are two ways of avoiding obvious mistakes, which are open to every engineer: one way is to make things so simple that there obviously aren't any mistakes; and the other way is to make things so complicated that there aren't any obvious mistakes"

C.A.R.Hoare

From mhtsa!ianj Sun May 31 12:07:53 1981 remote from usa netmail from basser40

... when we recognize the battle against chaos, mess, and unmastered complexity as one of computing science's major callings, we must admit that "Beauty is our Business."

E. W. Dijkstra

From mhtsa!ianj Thu Jun 4 09:42:17 1981 remote from usa netmail from basser40

I conclude that there are two ways of constructing a software design: One way is to make it so simple that there are obviously no deficiencies and the other way is to make it so complicated that there are no obvious deficiencies.

C. A. R. Hoare

From mhtsa!ianj Sun Jun 7 20:49:03 1981 remote from usa netmail from basser40

I have just been examining my mail loopback statistics for last few weeks
(ie the mail I send myself via australia)
and am most happy to report that nearly always
I receive my mail back the VERY next day.
no mail has been lost since the NINE bug was fixed

this is just great !!

ian

From dave Fri Jul 03 12:26:50 1981 netmail from unswcsu
To: auugn:elec70
Subject: memory wanted

Could you please place a "want ad" in AUUGN ? I am looking around for 32K core memory to take our 11/40 up to 128K. I have no wish to scrap the lot & replace it with MOS - I just want extra core.

PS
Those quantities are in words, not bytes.

(Dave obviously wants CHEAP, eg second hand core..... ED)

From root Sat Jul 4 15:17:19 1981 netmail from elec70
Subject: Peter Dalwood

from UCC (Syd. Uni) on 692-3491 would like a copy of the paper delivered by Ross Gaylor at the last Unix Meeting (AGSM) - does AUUGN have a copy?

kev

(No, sorry I dont have one. Pester Ross...... ED)

From dave Thu Jul 9 11:22:44 1981
Subject: dd conv=swab

You may like to publish this note. If the file has an odd number of words, the last word never gets swabbed. The fix is just to remove the "& ~01" which some idiot had coded. The right shift automatically takes care of the case where there is an odd number of bytes.

>From bob Mon Jul 13 10:45:56 1981 netmail from basser40
Date: Mon Jul 13 10:44:59 1981
From: Bob Kummerfeld <bob at basser40>
To: augn at elec70

To: augn:elec70
Subject: A note for the next AUGN
Cc: peteri at elecvax

Peter,
This is a short note for AUGN to let people know what has been happening in the area of networks, the Sydney Unix Network in particular.

We now have 13 machines on the SUN as "full" members, i.e. running the network daemon and supporting file/mail transfer. The 11/44 at Sydney University Computing Centre has a working link into the net but hasn't yet got the software installed. There are another half-dozen machines connected to the network that can be "con"ed to from other systems, these are smaller machines or larger machines like Cybers.

The most important development in the network area has been the connection of the Basser 11/40 to the CSIRONET. This link operates with the "mx" protocol and allows people to get out into CSIRONET and connect to other hosts, it also allows people on CSIRONET to connect to the Sydney Unix Net. This software to make this connection is now officially supported by CSIRO DCR and is available to allow Unix systems anywhere in Australia to connect to CSIRONET. CSIRO have several Unix systems in the Division of Maths & Stats that will soon be connected. Unix systems in the University of Tasmania and Monash are to be connected to CSIRONET soon with other University Unix sites interested.

Piers is going to modify the existing SUN net daemon to take care of part-time connections via CSIRONET thus allowing mail/files to be transfered between Unix systems connected to CSIRONET. This will make communication between Unix users much easier.

The "mx" protocol has been enhanced for the CSIRONET connection to provide for call establishment, call clear, line mode change (allows a true raw mode across csironet!!) and full CRC error checking. These enhancements are not necessary for dedicated connections normally but CSIRO felt they were needed for connection to their net.

Another planned enhancement for the SUN software is the ability to multi-drop files/mail. This will allow a single copy of a mail item that was directed to many people on a remote machine to be passed around the net. Only at the final delivery stage will multiple copies be created and mailed to the destination users.

At the last AUG meeting Ian Johnston and I mentioned a mail relay system that we planned to construct that would allow mail from the SUN to be delivered to machines in the USA on UnixNet, uucpnet, ARPANET or any network connected to them. This system is now in experimental operation and is being used to keep in contact with people such as Bill Joy at Berkeley as well as Ian Johnston and Andrew Hume at Bell. Unfortunately the system cannot be released for general use at the moment since Basser is paying the trans-pacific network charges and can't afford much! The charge is 60cents per thousand characters - it doesn't take long to mount up.

Bob Kummerfeld

IN REPLY PLEASE QUOTE:

TELEPHONE: 692 3491
692 1122
EXT 3491

UNIVERSITY COMPUTING CENTRE
THE UNIVERSITY OF SYDNEY
NSW 2006

13 April 1981

WHY DOES DEC THINK THAT THE NUMBER 44 LIES BETWEEN 45 & 70.

The 11/44 is the latest in the range of PDP11 - perhaps the last if DEC turn all their efforts to designing VAX's. According to the processor handbook, the 11/44 has a shade more power than the 11/45, and the cost of an 11/34. Architecturally, the 44 is closer to the 70, and except for a few small differences, and the speed, looks identical to it. It has 22 bit addressing (Unibus maps), separate I & D space, supervisor mode, and an 8K cache. On the other hand, there is no Massbus.

Physically, the 11/44 comes in a cabinet about the size of an RK07 disc drive. It is, however, the most badly designed box ever. The only cable exit takes a Unibus cable and that is all! We have half our cabinet empty waiting for backplane, with no chance of leading cables out of it, if it was there.

Sydney University Computing Centre bought a minimum configuration from DEC for $28,000 - processor, 256Kb of memory, and two TU58 DECTAPE II cassette tape drives. Since then we have hooked on two RM02 drives from Digital Electronics, a TU10 look-alike (except it's dual density), a DZ11-E, a DU11 Synchronous interface, an optical mark reader, 3 printers and a card reader, most of which is plugged into a DD11-CK extension cabinet.

The processor comes on 4 hex cards, memory comes in 256Kb chunks of ECC mos memory, up to 1 Mb. There is an optional floating point processor, and also a rather groovy commercial instruction set processor which enables a number of instructions like block moves, character searches and translations, edits, and the like (just like a VAX).

Other than optional stuff, the processor executes all 11/70 instructions, plus a few others, just to give you the chance to write incompatible software. There is a CSM instruction that calls supervisor mode and MFPT instruction which returns a processor specific subcode--currently 0.

The TU58 is a weirdo too. It is a replacement for the old DECTAPE, so it is a random access tape. It uses a cassette a bit smaller than an audio cassette and is virtually fool-proof - DEC say that removing the cassette while writing may cause loss of data, otherwise it is safe. There are absolutely no switches or buttons to misposition or mispress. The best part is that it interfaces to the processor through an ordinary KL11 asynchronous terminal line. There is a handshaking protocol for accessing any of the 512 blocks on the cassette. In theory, you could run UNIX off them - how excruciating. These things are now standard on 11/44 systems--an improvement of the paper tape.

The console has a micro, allowing interactive examination and depositing, booting from different devices (you have to purchase different ROMs), single stepping and tracing, stepping through the microcode, self testing, examination of all the internal registers of the machine, loading from paper tape and dumping to paper tape. More civilized than flick switches!

So much for the hardware. The standard level 7 tape will not boot and requires a few small changes to bring up. Fortunately, the level 7 system that had run occasionally on the 11/34 at the UCC, booted and ran without change. Thereafter, the comedy begins. Online RESTOR will not load--it needs separate I & D. You can't load RESTOR off the tape (it gives strange error messages). You, therefore, have to boot standalone RESTOR from the disc (it had to know about RM02s any way so a driver had to be written) and copy the tape to disc.

Once a mountable file system is available, the following changes are needed:

a. Standalone boot has to know about RM02s. (change /usr/src/cmd/standalone/conf.c) and add a driver.

b. The 11/44 cache has a different format to the 11/70. For an 11/70, both mch.s and /usr/src/cmd/standalone/M.s move 3 to the cache register MSCR. For an 11/44 change this to 0201.

c. Since the 11/44 does not have a Massbus, all drivers have to do a call to MAPALLOC for raw I/O to set the Unibus map registers. Similarly, the standalone utilities (mainly boot) have to set them, otherwise the initial disc read flakes out with leftover rubbish.

11/70s do not have this problem, as Massbus disc controllers have 22 bit address registers. I can't see how to boot standard level 7 from RK05s on an 11/70.

d. /usr/sys/trap.c handles memory parity errors and prints out the cache registers. Some of these don't exist on the 11/44, so change it to print out the ones that do.

The other alternative is to bring it up as an 11/45 in 18-bit mode (OK if you have only 265Kb). This has the advantage of no messing about with Unibus maps. /usr/src/cmd/Ms sets MMR3 to 65 to enable 22 bit mapping and Unibus maps. Change this to 5 and b & c above, disappear. Similar code exist in mch.s, but if the normal boot sequence is followed, it is never executed.

After that, it is all fairly clear cut, converting the distributed system to a usable one, installing bug fixes (is there a list anywhere?), creating the floating point simulator library (the makefile fails!) and the C compiler, installing all the non-Bell drivers, etc. etc.

The system itself is stable, and only goes down when the discs fail (as they do regularly - Ampex discs on an Emulex controller - aarrgghh!) I gave up waiting for the much vaunted DEC-supplied Unix tape, and it only took about 50 lines of codes anyway.

My final comment is addressed to the unremembered (sorry) Queensland site who were going to run a similar system to us -- contact me for a disc copy or code fixes.

<u>GEOFF COLE</u>

Adrian Freed
Groupe Informatique,
L.E.R.S,
58 Rue de la Glaciere,
75013 PARIS
France

589 8929 ext 227
1/6/81

Dear Peter,

This is the first letter from the French UNIX connection and is intended to give readers of AUUGN a programmers perspective of what is happening at LERS, a commercial licence holder, where I now work. Ian Perry (my boss) has given a socio-historic account in a previous issue.

Incidently, there seems to be a lot of movement of UNIX people at the moment. I met Peter Lamb from Melbourne on the plane coming here. He is working in Switzerland. Richard Miller is currently working at Oxford.

1. Installation

Hardware:

1	PDP11/70 with remote diagnostics
1	RP04
1	TU10 (the tapes keep falling of the spindle!)
2	300 Mbyte CDC drives with System Industries Massbus controller
>30	Perkin-Elmer fox terminals
7	DZ/11's
3	DL/11's
1	KMC/11 (currently unused)
2	Anderson Jacobson high quality printers (Qume mechanism)
1	Benson pen plotter
3	Printronix printer/plotters
1	Matrox 512x512 video display device
1	Imlac Calligraphic terminal (not connected to 11/70)
1	Wang word processing system

Communications hardware including time division and statistical multiplexers, leased lines, dial-up lines, 300,1200,9600 and 72000 baud modems.

DEC have installed a remote diagnostic facility, which allows them to stick fingers in the CPU from a distance. I miss the dull red glow of the idle loop on the front panel, but thats the price of progress I guess. Also supported, on terminal connections, are about 12 LSI/11 based systems for real-time work. I say about 12, because some are still being configured from bits ordered from different DEC compatible suppliers. The various configurations are subsets of the following:

1 LSI 11/23
1 LSI 11/1
2 double density, double sided floppy disks
1 real-time clock
1 line-time clock
1 16 channel A/D, 2 channel D/A system
1 parallel I/O board
Dec, Intel and Dataram memory
Serial asynchronous interfaces
Work is also being done with a 20Mb Winchester disk system.

Software:

LERS runs a reasonably standard v7, with named pipes. The disk driver was developed locally. There is some use of ´ex´, which is quite pleasant as most of the terminals run at 9600 baud.

People:

The computer group consists of an Australian, a Cameroonian, a Dutchman, three Frenchman, an Englishman and a Scotsman.

2. Activities

2.1 Data bases

A stable, reliable and simple data base system, designed and implemented by Jeff Rottman forms the basis of 12 data bases.

2.2 Signal Processing

The LSI/11 systems are used to acquire, store and process data of various forms. Certain processing is done on the 70. The LSI systems have a remote directory on the 11/70 known to the micro as /70. File transfers are managed using the line-protocol. The data is from ECG, EEG, liquid scintillation counters, Behavioural control boxs and any other laboratory equipement, that happens to be around.

2.3 Word Processing

In French and English, using ´nroff´, unsophisticated macros. An interface between the Wang system and UNIX is being developed. The backspace key is not used to erase characters. It is used to overstrike accents. The ´del´ key is the erase character. ^C and ^P are the DEL and QUIT characters respectively. This can be a bit confusing to a new-comer.

2.4 Pert networks

Software is being developed for project planning to augment the simple PERT networks, which are currently used.

2.5 Miscellaneous

There is a statistical package (ISP) which is part of the statistical support provided by the group. There are programs for chemical naming, costing and inventory. There is an extensive graphics library and an interactive graphics language based on it (LIG).

3. /* social */

Of course, one of the great things about working at LERS is Paris. My first impression was that Paris is cold. This impression has gradually matured to the stage where I can say Paris ranges from bloody freezing to cold. However, if you can master the lookahead required on storage heaters it can be quite pleasant. A tee-shirt is about as useful as a pair of sunglasses or a can of Aerogard. Paris is famous for a lot of other things. It positively brims with priceless art collections, museums and architectural marvels and monstrosities. It seems that any excuse for an exhibition or trade fair suffices. There are at least 200 cinemas and you can see films from any country sub-titled or dubbed in French. Most films are also shown in their original form somewhere in Paris.

Food is of course very important to the French. The range of cheeses available is simply mind-boggling. My first impression was that 90% of them were like Camembert. Actually, the ones that aren't breathing and leaping at you are Camembert, an innocent friendly cheese. Its the ones that pulsate and grow, before your eyes, that you have to be careful with. This reminds me of a joke, which I will try to translate:

Indignant customer in a restaurant.

M: Waiter, is this coffee or tea?
W: What does it taste of, Sir?
M: Petrol!
W: Then, that would be the tea, Sir.
M: And Waiter what about this cheese, it practically walks?
W: Ah, that cheese comes from Normandie, its a long way.

Believe me, its funnier in French.

A decent sized steak is something that French people only dream about. You have to mortgage your house to buy one. The meat pie does not exist in France (sorry Kev). In fact, it is impossible to buy a pie.

The basic rule of French food is that they eat every part of anything that moves. This includes snails, frogs, goats, squirrels, rabbits, hare etc. I can live without snails, but frog's thighs (as the French call them) are rather good.

Actually, the hardest thing to get used to is wearing a suit to work (IRP: He doesn't very often).

If anyone in the UNIX community happens to be in Paris do get in touch.

Yours Faithfully,

A. D. Freed

Adrian Freed

THE UNIVERSITY OF NEW SOUTH WALES

P.O. BOX 1 • KENSINGTON • NEW SOUTH WALES • AUSTRALIA • 2033
TELEX AA26054 • TELEGRAPH: UNITECH, SYDNEY • TELEPHONE 663 0351
EXTN.

PLEASE QUOTE

SCHOOL OF ELECTRICAL ENGINEERING

8th July 1981

Peter Ivanov,
AUUGN Newsletter Editor,
Dept. of Computer Science,
University of N.S.W.

Dear Peter,

I am interested in making contact with any Australian Icon users or intending users, and I think that AUUGN is probably the most effective method. Icon has been suggested as an alternative to Snobol in our second year Programming I course so I am interested in finding out about other peoples' experiences with Icon, both for teaching and research. I can be contacted on the Sydney Unix network as davec:elecvax.

Yours sincerely,

D A Carrington

David Carrington,
Lecturer,
Dept. of Computer Science.

Position being advertised at U.N.S.W. for Analyst/Programmer

After the normal 3-month probationary period, the successful applicant will be graded according to the following scale:

Far Exceeds Job Requirements

Lifts buildings and walks under them.
Catches speeding bullets with his teeth and eats them.
Kicks locomotives off the track.
Freezes water with a single glance.
Gives policy to God.

Exceeds Job Requirements

Leaps tall buildings with a single bound.
Is faster than a speeding bullet.
Is more powerful than a locomotive.
Walks on water.
Talks with God.

Meets Job Requirements

Must take running start to leap over tall buildings.
Is just as fast as a speeding bullet.
Is stronger than a bull elephant.
Walks on water of indoor swimming pools.
Talks with the angels.

Needs Some Improvement

Can leap over short buildings only.
Wounds self with bullet when attempting to shoot.
Recognises locomotives two out of three times.
Swims well.
Talks with self.

Needs Much Improvement

Crashes into buildings when leaping.
Is not issued with ammunition.
Smells like the bull elephant.
Drinks water in emergencies.
Argues with self.

Does Not Meet Minimum Requirements

Cannot recognise buildings at all.
Wets himself with water pistol.
Says "Look at the choo choo".
Plays in mud puddles.
Loses arguments with self.

THE UNIVERSITY OF NEW SOUTH WALES
SCHOOL OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE
ANALYST PROGRAMMER (REF 377)

Graduate with good degree in Computer Science or equivalent, with a minimum of two years experience in operating systems or other large programming tasks and with experience in using Pascal-like languages is required. Duties include design, implementation and maintenance of systems and application software on VAX and PDP11 computers running the UNIX operating system in a teaching environment, evaluation of software from outside sources, and liaison and consultation with staff and users.

Experience with UNIX operating systems and programming VAX and PDP equipment as well as ability to work as a team member desirable.

Salary $18,796 range $20,[illegible]

Commencing salary according to qualifications and experience.

Applications close July 27, 1981.

Application form from General Staff Office (02) 66 [illegible] P.O. Box 1, Kensington N.S.W. 2033