



VAX11 780



SYSTEMS AND OPTIONS SUMMARY

APRIL 1978

digital

Contents

INTRODUCTION TO VAX-11/780 SYSTEMS	1
VAX-11/780 SYSTEMS	
Dual RK07 disk-based system	2
RM03 disk/TE16 magnetic tape-based system	4
RP06 disk/TE16 magnetic tape-based system	6
VAX-11/780 PROCESSOR & MEMORY OPTIONS	
Processor Options	8
Expansion Memory	8
VAX-11/780 MASSBUS OPTIONS	
VAX-11/780 Expansion Cabinet	9
VAX-11/780 MASSBUS Peripherals	
Disk Pack Subsystems	9
Add-On Disk Pack Drives	10
Dual Access & Upgrade Options	11
Disk Packs	12
Magnetic Tape Subsystem	13
Add-On Magnetic Tape Drive	13
VAX-11/780 UNIBUS OPTIONS	
VAX-11/780 UNIBUS Options Cabinet	14
VAX-11/780 Extension Mounting Box	14
System Unit Expansion	14
VAX-11/780 UNIBUS Peripherals	
Cartridge Disk Subsystems	15
Add-On Cartridge Disk Drives	16
Dual Access Options	16
Cartridge Disks and Accessories	17
Asynchronous Multiplexers (Programmed I/O)	18
Single Line Synchronous Interfaces	19
VAX-11/780 INPUT/OUTPUT DEVICES	
Line Printers	21
Card Reader	22
Terminals	22
LANGUAGES & UTILITIES FOR VAX-11/780 SYSTEMS	24
INDEX	26

This summary was designed, produced and typeset
by DIGITAL's Sales Support Literature Group
using an In-House text-processing system
operating on a DECSYSTEM-20.

Introduction to VAX-11/780 Systems

VAX-11/780 is DIGITAL's new 32-bit computer system designed for interactive environments and high throughput applications.

The three basic systems described in this product summary combine the powerful VAX/VMS virtual memory operating system with a selection of system disks and backup/load devices. Also included are descriptions and configuring information for VAX-11/780 add-on options.

LEGEND

System Code/Option Code

The first entry is the order number for the system/option, with 115 Vac, 60 cycle power. The second entry, shown immediately below in *italics* is used for 230 Vac, 50 cycle power.

The basic features and specifications of each system/option are included in the description. More complete hardware and software descriptions can be found in handbooks and Software Product Descriptions.

SU	System Unit. Unit of space in chassis for mounting pre-wired backplanes(s) which can accept Hex- or Quad-sized modules.
SU 1-2	Defined here as the first two system units in a BA11-K UNIBUS expander box.
SU 3-5	Defined here as the last three system units in a BA11-K UNIBUS expander box.
Hex slot	Space in pre-wired backplane which will accept a 15.604 inch (39.634cm) high module.
Quad slot	Space in pre-wired backplane which will accept a 10.437 inch (26.510cm) high module.
MBA	MASSBUS adapter

+5V Power Available/Drawn

The +5V current available in or drawn from the system.

System UNIBUS Loads Available/Drawn

The number of UNIBUS loads remaining on or drawn from the UNIBUS. There can be a total of 20 loads or 50 feet (15.2 meters) of UNIBUS cable before a UNIBUS repeater (DB11) is needed.

Mounting Code

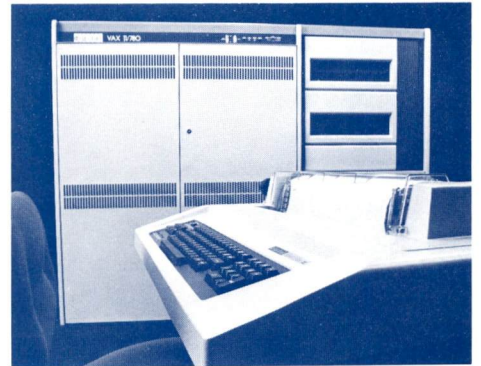
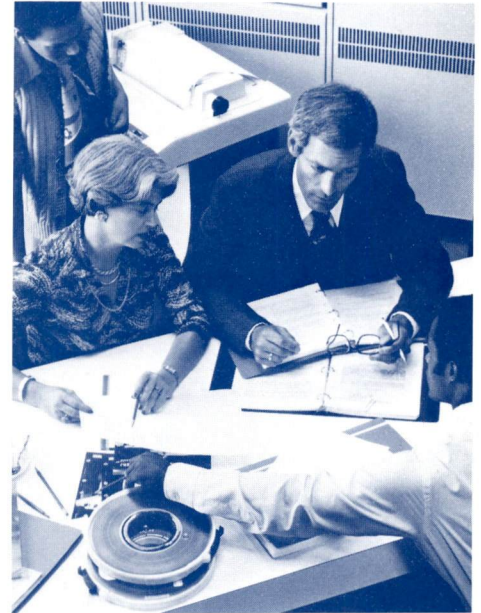
The mounting code indicates how the option mounts into the system.

CAB	Cabinet mounted.
Dedicated CAB	Option is packaged in its own cabinet.
FS	Free standing unit.
TT	Table top unit.
PAN	Panel mounted. Front panel height is 10.5 inches (26.7cm).
SM PAN	Small panel. Front panel height is 5.25 inches (13.3cm).

Support Category

The software product includes in the license fee, at a minimum, the service specified for the category indicated:

- A On-site installation, one year Software Performance Reporting (SPR) Service, remedial service within the first 90 days.
- B One year SPR service.



Dual RK07 disk-based VAX-11/780 System

System Code

SV-AXHHA-LA

SV-AXHHA-LD

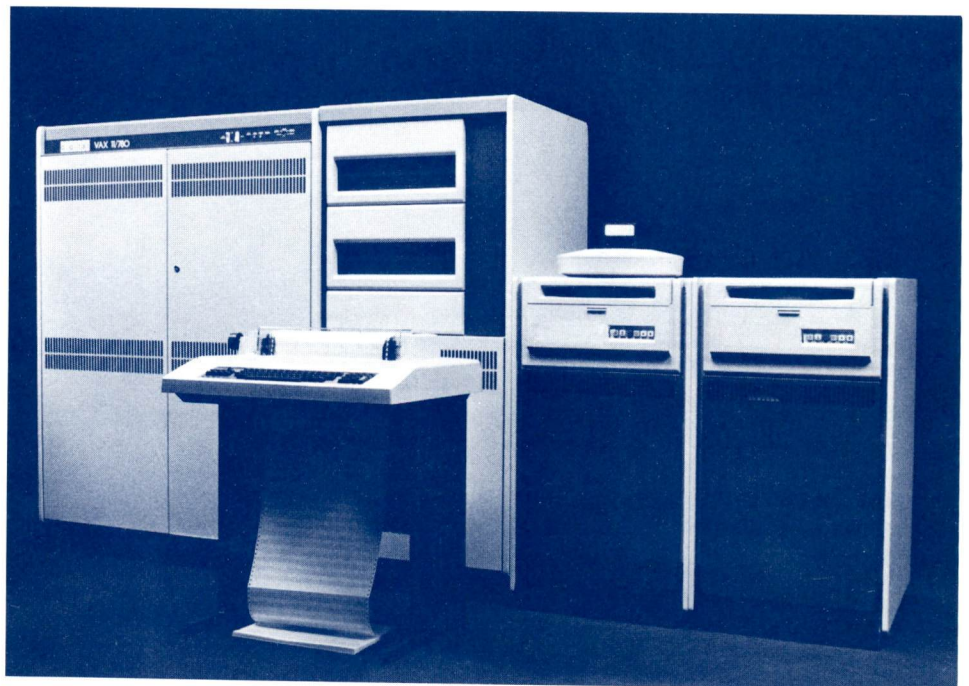
This VAX-11/780 interactive computer system, featuring dual RK07 disk-based storage devices, is designed to provide a high-performance, general-purpose, multiprogramming environment by combining DIGITAL's powerful VAX-11/780 central processor and the fully supported VAX/VMS operating system.

The system is configured with 256K bytes of ECC MOS memory, an RK711 UNIBUS controller with two top loading RK07 cartridge disk drives for a total of 56 megabytes of on-line storage, one DZ11-A asynchronous multiplexer providing eight EIA terminal lines, and an LA36 DECwriter II console terminal.

Basic equipment for this system includes the VAX-11/780 central processing unit with virtual memory management, bootstrap loader, standard instructions for packed decimal, floating and fixed point arithmetic, and character and string manipulations, 8K byte parity bipolar cache memory, high precision programmable real-time clock, time-of-year clock (with battery backup), and 12K bytes of writable diagnostic control store.

Also included as standard equipment is an integral diagnostic console subsystem, for use in both local and remote operations, which consists of an intelligent micro-computer (LSI-11 with 16K bytes read/write memory and 8K bytes read only memory) to which an RX01 floppy disk and the LA36 DECwriter II are connected.

This VAX-11/780 configuration is arranged in a 60"(H) x 48"(W) x 30"(D) (152.4cm x 122cm x 76.2cm) double-width high-boy cabinet with power supplies, two free-standing dedicated cabinet units, and one single-width high-boy UNIBUS expansion cabinet which includes a BA11-K extension mounting box, one DD11-DK backpanel mounting unit, and a DZ11-A distribution panel.



EXPANSION CAPABILITY

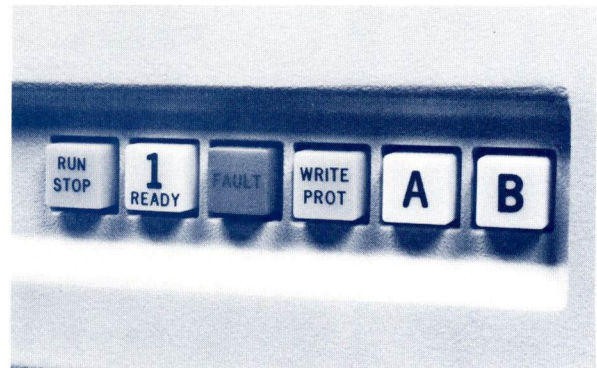
Expansion space for this system is available in three separate areas:

- Pre-designated space in the VAX-11/780 CPU Cabinet
- UNIBUS Expansion Cabinet
- VAX-11/780 Expansion Cabinet (not included)

CPU Cabinet

The CPU cabinet itself has pre-designated space available to accept the following options:

- FP780-AA(AB) high-performance floating-point accelerator with power supply.
- 12K bytes writable control store (KU780).
- An additional 768K bytes of ECC MOS memory.
- Memory battery backup (H7112) for up to one million bytes of memory.
- Serial line unit for remote diagnosis. (Remote diagnosis is only available to those customers under the terms and conditions of a current DIGITAL Field Service contract.)
- Plus, space for two MASSBUS adapters. (Adapter logic is bundled in with the REM03, REP05, REP06, and TEE16.)



UNIBUS Expansion Cabinet

UNIBUS expansion specified for the VAX-11/780 systems can be added in the system's single-width, high-boy UNIBUS expansion cabinet in the same way that conventional UNIBUS options are added to PDP-11 computers. This cabinet can accept a total of two BA11-K extension mounting boxes and three DZ11 distribution panels.

For UNIBUS expansion beyond the limits of this cabinet, an H9602-DF(DH) "add-on" UNIBUS Options Cabinet may be ordered.

Please note, however, that only those UNIBUS options described in this summary are supported on VAX-11/780 systems.



Box	Expansion Space	+5V Power Available	UNIBUS Loads Available
BA11-K			18
SU 1 - 2	6 Hex slots, 2 Quad slots	22.8	
SU 3 - 5	2 Hex slots, 1 Quad slot, 1 SU	10.0	

VAX-11/780 Expansion Cabinet

A single-width, high-boy VAX-11/780 expansion cabinet, H9602-HA(HB), (not included with this system) is available and provides mounting space for an additional one million bytes of memory with control, space for two MASSBUS adapters, and one memory battery backup option (H7112-A/B).

RM03 disk/TE16 magnetic tape-based VAX-11/780 System

System Code

SV-AXTVA-LA

SV-AXTVA-LD

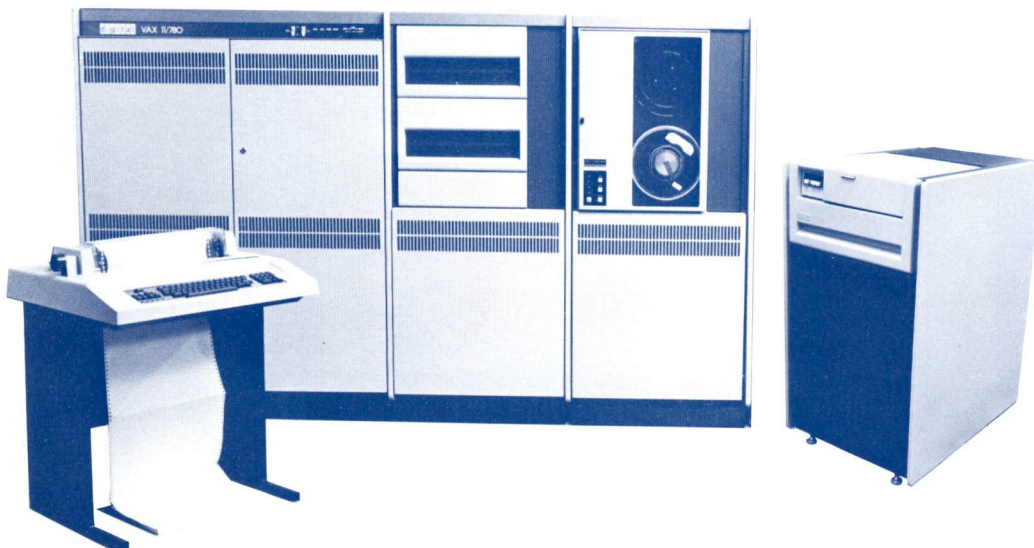
This VAX-11/780 interactive computer system, featuring RM03 disk and TE16 magnetic tape-based storage devices, is designed to provide a high-performance, general-purpose, multiprogramming environment by combining DIGITAL's powerful VAX-11/780 central processor and the fully supported VAX/VMS operating system.

The system is configured with 256K bytes of ECC MOS memory, an REM03 single access 67 million byte disk drive with MASSBUS adapter, a TEE16 magnetic tape transport unit (45 inches/second) with MASSBUS adapter, one DZ11-A asynchronous multiplexer providing eight EIA terminal lines, and an LA36 DECwriter II console terminal.

Basic equipment for this system includes the VAX-11/780 central processing unit with virtual memory management, bootstrap loader, standard instructions for packed decimal, floating and fixed point arithmetic, and character and string manipulations, 8K byte parity bipolar cache memory, high precision programmable real-time clock, time-of-year clock (with battery backup), and 12K bytes of writable diagnostic control store.

Also included as standard equipment is an integral diagnostic console subsystem, for use in both local and remote operations, which consists of an intelligent micro-computer (LSI-11 with 16K bytes read/write memory and 8K bytes read only memory) to which an RX01 floppy disk and the LA36 DECwriter II are connected.

This VAX-11/780 configuration is arranged in a 60"(H) x 48"(W) x 30"(D) (152.4cm x 122cm x 76.2cm) double-width high-boy cabinet with power supplies, one free-standing dedicated disk cabinet, one dedicated single-width high-boy tape transport cabinet, and one single-width high-boy UNIBUS expansion cabinet which includes a BA11-K extension mounting box, one DD11-DK backpanel mounting unit, and a DZ11-A distribution panel.



EXPANSION CAPABILITY

Expansion space for this system is available in three separate areas:

- Pre-designated space in the VAX-11/780 CPU Cabinet
- UNIBUS Expansion Cabinet
- VAX-11/780 Expansion Cabinet (not included)

CPU Cabinet

The CPU cabinet itself has pre-designated space available to accept the following options:

- FP780-AA(AB) high-performance floating-point accelerator with power supply.
- 12K bytes writable control store (KU780).
- An additional 768K bytes of ECC MOS memory.
- Memory battery backup (H7112) for up to one million bytes of memory.
- Serial line unit for remote diagnosis. (Remote diagnosis is only available to those customers under the terms and conditions of a current DIGITAL Field Service contract.)

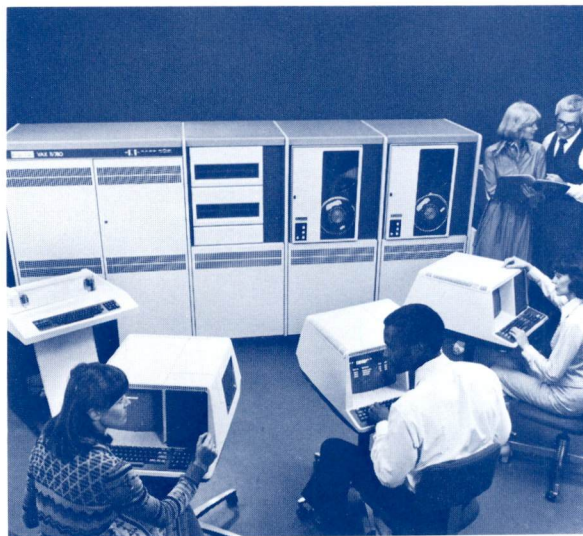
UNIBUS Expansion Cabinet

UNIBUS expansion specified for the VAX-11/780 systems can be added in the system's single-width, high-boy UNIBUS expansion cabinet in the same way that conventional UNIBUS options are added to PDP-11 computers. This cabinet can accept a total of two BA11-K extension mounting boxes and three DZ11 distribution panels.

For UNIBUS expansion beyond the limits of this cabinet, an H9602-DF(DH) "add-on" UNIBUS Options Cabinet may be ordered.

Please note, however, that only those UNIBUS options described in this summary are supported on VAX-11/780 systems.

Box	Expansion Space	+5V Power Available	UNIBUS Loads Available
BA11-K			19
SU 1 - 2	6 Hex slots, 2 Quad slots	22.8	
SU 3 - 5	3 SUs	25.0	



VAX-11/780 Expansion Cabinet

A single-width, high-boy VAX-11/780 expansion cabinet, H9602-HA(HB), (not included with this system) is available and provides mounting space for an additional one million bytes of memory with control, space for two MASSBUS adapters, and one memory battery backup option (H7112-A/B).

RP06 disk/TE16 magnetic tape-based VAX-11/780 System

System Code
SV-AXCVA-LA
SV-AXCVA-LD

This VAX-11/780 interactive computer system, featuring RP06 disk and TE16 magnetic tape-based storage devices, is designed to provide a high-performance, general-purpose, multiprogramming environment by combining DIGITAL's powerful VAX-11/780 central processor and the fully supported VAX/VMS operating system.

The system is configured with 512K bytes of ECC MOS memory, an REP06 single access 176 million byte disk drive with MASSBUS adapter, a TEE16 magnetic tape transport unit (45 inches/second) with MASSBUS adapter, one DZ11-A asynchronous multiplexer providing eight EIA terminal lines, and an LA36 DECwriter II console terminal.

Basic equipment for this system includes the VAX-11/780 central processing unit with virtual memory management, bootstrap loader, standard instructions for packed decimal, floating and fixed point arithmetic, and character and string manipulations, 8K byte parity bipolar cache memory, high precision programmable real-time clock, time-of-year clock (with battery backup), and 12K bytes of writable diagnostic control store.

Also included as standard equipment is an integral diagnostic console subsystem, for use in both local and remote operations, which consists of an intelligent micro-computer (LSI-11 with 16K bytes read/write memory and 8K bytes read only memory) to which an RX01 floppy disk and the LA36 DECwriter II are connected.

This VAX-11/780 configuration is arranged in a 60"(H) x 48"(W) x 30"(D) (152.4cm x 122cm x 76.2cm) double-width high-boy cabinet with power supplies, one free-standing dedicated disk cabinet, one dedicated single-width high-boy tape transport cabinet, and one single-width high-boy UNIBUS expansion cabinet which includes a BA11-K extension mounting box, one DD11-DK backpanel mounting unit, and a DZ11-A distribution panel.



EXPANSION CAPABILITY

Expansion space for this system is available in three separate areas:

- Pre-designated space in the VAX-11/780 CPU Cabinet
- UNIBUS Expansion Cabinet
- VAX-11/780 Expansion Cabinet (not included)

CPU Cabinet

The CPU cabinet itself has pre-designated space available to accept the following options:

- FP780-AA(AB) high-performance floating-point accelerator with power supply.
- 12K bytes writable control store (KU780).
- An additional 512K bytes of ECC MOS memory.
- Memory battery backup (H7112) for up to one million bytes of memory.
- Serial line unit for remote diagnosis. (Remote diagnosis is only available to those customers under the terms and conditions of a current DIGITAL Field Service contract.)

UNIBUS Expansion Cabinet

UNIBUS expansion specified for the VAX-11/780 systems can be added in the system's single-width, high-boy UNIBUS expansion cabinet in the same way that conventional UNIBUS options are added to PDP-11 computers. This cabinet can accept a total of two BA11-K extension mounting boxes and three DZ11 distribution panels.

For UNIBUS expansion beyond the limits of this cabinet, an H9602-DF(DH) "add-on" UNIBUS Options Cabinet may be ordered.

Please note, however, that only those UNIBUS options described in this summary are supported on VAX-11/780 systems.



Box	Expansion Space	+5V Power Available	UNIBUS Loads Available
BA11-K			19
SU 1 - 2	6 Hex slots, 2 Quad slots	22.8	
SU 3 - 5	3 SUs	25.0	

VAX-11/780 Expansion Cabinet

A single-width, high-boy VAX-11/780 expansion cabinet, H9602-HA(HB), (not included with this system) is available and provides mounting space for an additional one million bytes of memory with control, space for two MASSBUS adapters, and one memory battery backup option (H7112-A/B).

VAX-11/780 Processor & Memory Options

PROCESSOR OPTIONS

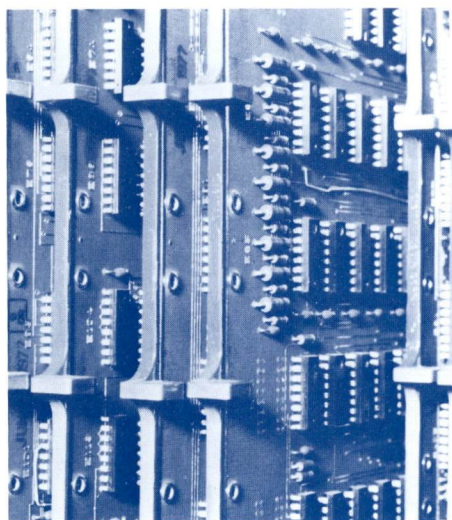
Please note that there is adequate power and prewired mounting space in the CPU cabinet to add the following options to VAX-11/780 systems.

FP780-AA
FP780-AB High-performance floating-point accelerator for single- and double-precision floating-point instructions plus POLY, EMOD and MULL. Power supply is also included.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
Prewired CPU slots	N/A	N/A

KU780 12K byte writable control store. Please note that this option is not supported by VAX-11 system software.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
Prewired CPU slots	N/A	N/A



EXPANSION MEMORY

MS780-AA
MS780-AB 128K byte ECC MOS memory with controller. Expandable to a total of one million bytes with the addition of seven MS780-BAs or other combinations of expansion memories listed below. This option must be ordered for expansion beyond one million bytes of CPU cabinet-mounted memory. (Please note that one MS780-AA(AB) is included with each VAX-11/780 system to accommodate the first million bytes.)

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
H9602-HA(HB)	N/A	N/A

MS780-BA 128K byte ECC MOS expansion memory.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
MS780-AA(AB)	N/A	N/A

MS780-BB 256K byte ECC MOS expansion memory.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
MS780-AA(AB)	N/A	N/A

MS780-BC 512K byte ECC MOS expansion memory.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
MS780-AA(AB)	N/A	N/A

H7112-A
H7112-B MOS memory battery backup. Powers up to one million bytes of memory for at least 10 minutes, or less memory for longer than 10 minutes.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
VAX-11/780 CPU CAB or H9602-HA(HB)	N/A	N/A

VAX-11/780 MASSBUS Options

VAX-11/780 EXPANSION CABINET

H9602-HA
H9602-HB

Single-width, high-boy expansion cabinet. 60"(H) x 28"(W) x 30"(D) (152.4cm x 71.2cm x 76.2cm). Designed to provide mounting space for up to one million bytes of memory with control, one memory battery backup option (H7112-A/B), and space for two MASSBUS adapters (which are bundled in with the REM03, REP05, REP06, and TEE16).

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
FS Cabinet	N/A	N/A

VAX-11/780 MASSBUS PERIPHERALS

Note 1: Each VAX-11/780 system has adequate power available to add the following MASSBUS peripherals. For this reason, the power drawn by these options is stated as not applicable.

Note 2: Average access time is defined as the sum of the average seek time plus the average latency. All average access times given below are stated as worst case. All capacities are formatted.

The nominal positioning time for the following disk drives is 28 msec. However, the positioning time for any disk drive varies slightly in a statistical distribution around the nominal value. Therefore, the worst case average positioning time is stated at 30 msec.

Disk Pack Subsystems

The dual access capability of disk subsystems is not supported by DIGITAL operating system software nor diagnostics.

REM03-AA
REM03-AD

Single-access 67 megabyte removable disk pack drive and VAX-11/780 MASSBUS adapter. Expandable to a total of 8 single access RM03 drives. One RM03-P disk pack is included. 1.2 megabytes/second peak transfer rate, 38.3 msec average access time.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
FS Drive Predesignated MBA space in CPU CAB or H9602-HA(HB)	N/A	N/A

REM03-BA
REM03-BD

Dual-access 67 megabyte removable disk pack drive and two VAX-11/780 MASSBUS adapters. Expandable to a total of 8 dual access RM03 drives. One RM03-P disk pack is included. 1.2 megabytes/second peak transfer rate, 38.3 msec average access time.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
FS Drive 2 predesignated MBA spaces in CPU CAB or H9602-HA(HB)	N/A	N/A



VAX-11/780 MASSBUS Options



REP05-AA
REP05-AB

Single-access 88 megabyte removable disk pack drive and VAX-11/780 MASSBUS adapter. Expandable to a total of 8 single access RP drives (RP05, RP06). One RP04-P disk pack is included. 806K bytes/second peak transfer rate, 38.3 msec average access time. Field upgradable to the RP06.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
FS Drive Predesignated MBA space in CPU CAB or H9602-HA(HB)	N/A	N/A

REP05-BA
REP05-BB

Dual-access 88 megabyte removable disk pack drive and two VAX-11/780 MASSBUS adapters. Expandable to a total of 8 dual access RP drives (RP05, RP06). One RP04-P disk pack is included. 806K bytes/second peak transfer rate, 38.3 msec average access time. Field upgradable to the RP06.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
FS Drive 2 predesignated MBA spaces in CPU CAB or H9602-HA(HB)	N/A	N/A

REP06-AA
REP06-AB

Single-access 176 megabyte removable disk pack drive and VAX-11/780 MASSBUS adapter. Expandable to a total of 8 single access RP drives (RP05, RP06). One RP06-P disk pack is included. 806K bytes/second peak transfer rate, 38.3 msec average access time.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
FS Drive Pre-designated MBA space in CPU CAB or H9602-HA(HB)	N/A	N/A

REP06-BA
REP06-BB

Dual-access 176 megabyte removable disk pack drive and two VAX-11/780 MASSBUS adapters. Expandable to a total of 8 dual access RP drives (RP05, RP06). One RP06-P disk pack is included. 806K bytes/second peak transfer rate, 38.3 msec average access time.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
FS Drive 2 predesignated MBA spaces in CPU CAB or H9602-HA(HB)	N/A	N/A

Add-On Disk Pack Drives

RM03-AA
RM03-AD

Single-access 67 megabyte removable disk pack drive. One RM03-P disk pack is included. 1.2 megabytes/second peak transfer rate, 38.3 msec average access time.

Prerequisite: REM03-AA(AD) disk subsystem.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
FS Drive	N/A	N/A

RM03-BA
RM03-BD

Dual-access 67 megabyte removable disk pack drive. One RM03-P disk pack is included. 1.2 megabytes/second peak transfer rate, 38.3 msec average access time.
Prerequisite: REM03-BA(BD) disk subsystem.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
FS Drive	N/A	N/A

RP05-AA
RP05-AB

Single-access 88 megabyte removable disk pack drive. One RP04-P disk pack is included. 806K bytes/second peak transfer rate, 38.3 msec average access time. Field upgradable to the RP06.
Prerequisite: REP05-AA(AB) disk subsystem.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
FS Drive	N/A	N/A

RP05-BA
RP05-BB

Dual-access 88 megabyte removable disk pack drive. One RP04-P disk pack is included. 806K bytes/second peak transfer rate, 38.3 msec average access time. Field upgradable to the RP06.
Prerequisite: REP05-BA(BB) disk subsystem.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
FS Drive	N/A	N/A

RP06-AA
RP06-AB

Single-access 176 megabyte removable disk pack drive. One RP06-P disk pack is included. 806K bytes/second peak transfer rate, 38.3 msec average access time.
Prerequisite: REP06-AA(AB) disk subsystem.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
FS Drive	N/A	N/A

RP06-BA
RP06-BB

Dual-access 176 megabyte removable disk pack drive. One RP06-P disk pack is included. 806K bytes/second peak transfer rate, 38.3 msec average access time.
Prerequisite: REP06-BA(BB) disk subsystem.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
FS Drive	N/A	N/A



Dual Access & Upgrade Options

The dual access capability of disk subsystems is not supported by DIGITAL operating system software nor diagnostics.

REM03-DA
REM03-DB

RM03 dual access conversion kit. Contains RM03-C, VAX-11/780 MASSBUS adapter and power supply to convert REM03-A to REM03-B.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
REM03-A	N/A	N/A

RM03-C

RM03 dual access kit containing drive logic and cables to convert RM03-A to RM03-B.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
RM03-A	N/A	N/A

VAX-11/780 MASSBUS Options

REP05-DA
REP05-DB

RP05 dual access conversion kit. Contains RP05-C, VAX-11/780 MASS-BUS adapter and power supply to convert REP05-A to REP05-B.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
REP05-A	N/A	N/A

RP05-C

RP05 dual access kit containing drive logic and cables to convert RP05-A to RP05-B.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
RP05-A	N/A	N/A

REP06-DA
REP06-DB

RP06 dual access conversion kit. Contains RP06-C, VAX-11/780 MASS-BUS adapter and power supply to convert REP06-A to REP06-B.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
REP06-A	N/A	N/A

RP06-C

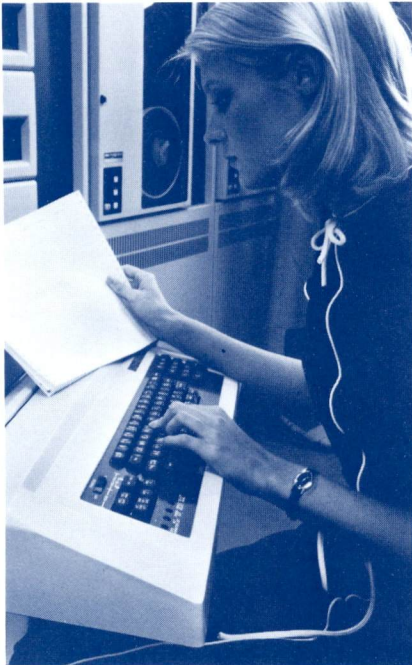
RP06 dual access kit containing drive logic and cables to convert RP06-A to RP06-B.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
RP06-A	N/A	N/A

RP06-U

RP05 to RP06 upgrade kit. Includes drive upgrade parts and RP06-P disk pack.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
RP05	N/A	N/A



Disk Packs

RM03-P

67 megabyte removable disk pack for RM03.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
RM03	N/A	N/A

RP04-P

88 megabyte removable disk pack for RP05.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
RP05	N/A	N/A

RP06-P

176 megabyte removable disk pack for RP06.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
RP06	N/A	N/A

Magnetic Tape Subsystem

TEE16-AE
TEE16-AJ

Program selectable 800 or 1600 bpi, 9-track, 45 inches/second, magnetic tape transport and VAX-11/780 MASSBUS adapter. Industry compatible. Expandable to total of eight TE16 transports.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
Dedicated CAB plus pre-designated MBA space in CPU CAB or H9602-HA(HB)	N/A	N/A

Add-On Magnetic Tape Drive

TE16-AE
TE16-AJ

Program selectable 800 or 1600 bpi, 9-track, 45 inches/second, magnetic tape transport unit. Industry compatible.
Prerequisite: TEE16 magtape subsystem.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
Dedicated CAB	N/A	N/A



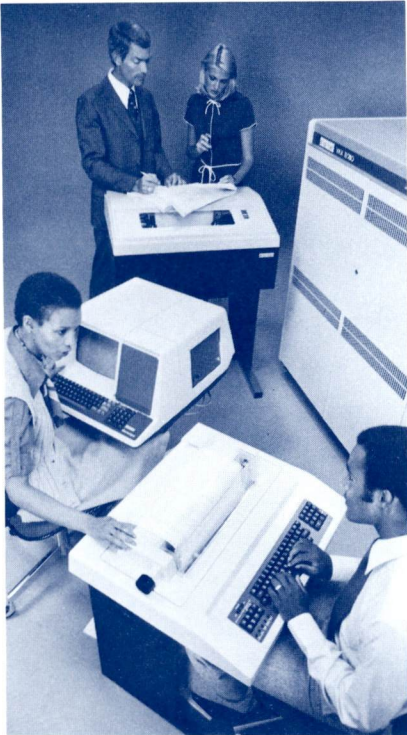
VAX-11/780 UNIBUS Options

VAX-11/780 UNIBUS OPTIONS CABINET

H9602-DF
H9602-DH

Single-width, high-boy "add-on" UNIBUS expansion cabinet with single-phase power control. Provides space for an additional two BA11-K boxes and three DZ11 distribution panels. Backpanel mounting units must be ordered separately. 60"(H) x 28"(W) x 30"(D) (152.4cm x 71.2cm x 76.2cm).

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
FS Cabinet	0.0	0



VAX-11/780 EXTENSION MOUNTING BOX

BA11-KE
BA11-KF

Rack-mountable extension mounting box. Provides mounting space for five system units. SUs 1-2 together, and SUs 3-5 together, each have 25.0 amps of power available @ +5V. One BA11-K is already included with each VAX-11/780 system.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
PAN space	50.0 total available	0

SYSTEM UNIT EXPANSION

DD11-CK

Backpanel mounting unit. Provides space for 2 Hex and 2 Quad slot modules.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
1 SU in BA11-K	N/A	N/A

DD11-DK

Backpanel mounting unit. Provides space for 7 Hex and 2 Quad slot modules. One DD11-DK is included with each VAX-11/780 system.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
2 SUs in BA11-K	N/A	N/A

BB11

Blank mounting panel for custom interface design and mounting system units.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
PAN space 1 SU in BA11-K	N/A	N/A

VAX-11/780 UNIBUS PERIPHERALS

Please note that average access time is defined as the sum of the average seek time plus the average latency.

Cartridge Disk Subsystems

The dual access capability of disk subsystems is not supported by DIGITAL operating system software nor diagnostics.

RK711-EA
RK711-EB Single-access 28 megabyte disk drive and control unit. Expandable to a total of eight single access RK06 or RK07 drives. One RK07K-DC data cartridge is included. Average access time of 49.0 msec, peak transfer rate of 538K bytes per second.

NOTE: The RK711 controller requires 2 SUs of mounting space in a BA11-K and has 2 Hex slots and 1 Quad slot of additional UNIBUS expansion space.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
2 SUs FS Drive	15.0	1

RK711-FA
RK711-FB Dual-access 28 megabyte disk drive and two control units. Expandable to a total of eight dual access RK06 or RK07 drives. One RK07K-DC data cartridge is included. Average access time of 49.0 msec, peak transfer rate of 538K bytes per second.

NOTE: Each RK711 controller requires 2 SUs of mounting space in a BA11-K and has 2 Hex slots and 1 Quad slot of additional UNIBUS expansion space.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
2 SUs per controller and FS Drive	15.0 per controller	1 per controller

RK611-EA
RK611-ED Single access 14 megabyte disk drive and control unit. Expandable to a total of eight single access RK06 drives. One RK06K-DC data cartridge is included. Average access time of 50.5 msec, peak transfer rate of 538K bytes per second.

Note: The RK611 controller requires 2 SUs of mounting space in a BA11-K and has 2 Hex slots and 1 Quad slot of additional UNIBUS expansion space.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
2 SUs FS Drive	15.0	1

RK611-FA
RK611-FD Dual access 14 megabyte disk drive and two control units. Expandable to a total of eight dual access RK06 drives. One RK06K-DC data cartridge is included. Average access time of 50.5 msec, peak transfer rate of 538K bytes per second.

Note: Each RK611 controller requires 2 SUs of mounting space in a BA11-K and has 2 Hex slots and 1 Quad slot of additional UNIBUS expansion space.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
2 SUs per controller and FS Drive	15.0 per controller	1 per controller



VAX-11/780 UNIBUS Options

Add-On Cartridge Disk Drives

RK07-EA
RK07-EB Single-access 28 megabyte disk drive. One RK07K-DC data cartridge is included. Average access time of 49.0 msec, peak transfer rate of 538K bytes per second.

Prerequisite: RK711-E or RK611-E subsystem.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
FS Drive	0.0	0

RK07-FA
RK07-FB Dual-access 28 megabyte disk drive. One RK07K-DC data cartridge is included. Average access time of 49.0 msec, peak transfer rate of 538K bytes per second.

Prerequisite: RK711-F or RK611-F subsystem.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
FS Drive	0.0	0

RK06-EA
RK06-ED Single access 14 megabyte disk drive. One RK06K-DC data cartridge is included. Average access time of 50.5 msec, peak transfer rate of 538K bytes per second.

Prerequisite: RK611-E controller.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
FS Drive	0.0	0

RK06-FA
RK06-FD Dual access 14 megabyte disk drive. One RK06K-DC data cartridge is included. Average access time of 50.5 msec, peak transfer rate of 538K bytes per second.

Prerequisite: RK611-F controller.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
FS Drive	0.0	0



Dual Access Options

The dual access capability of disk subsystems is not supported by DIGITAL operating system software nor diagnostics.

RK711-C Dual-access kit containing drive logic and hardware, one controller and cables to convert an RK711-E to an RK711-F.
Prerequisite: RK711-E.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
2 SUs for controller	15.0 for controller	1 for controller

RK07-C Dual-access kit containing drive logic, hardware and cables to convert an RK07-E to an RK07-F.
Prerequisite: RK07-E.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
RK07-E	0.0	0

RK611-C Dual access kit containing drive logic and hardware, one controller and cables to convert an RK611-E to an RK611-F.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
2 SUs for controller	15.0 for controller	1 for controller

RK06-C Dual access kit containing drive logic, hardware and cables to convert an RK06-E to an RK06-F.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
RK06-E	0.0	0



Cartridge Disks and Accessories

RK07K-EF Error free 28 megabyte data cartridge for RK07 subsystems.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
RK07	0.0	0

RK07K-AC 28 megabyte alignment cartridge for RK07.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
RK07	0.0	0

RK07K-DC 28 megabyte data cartridge for RK07 subsystems.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
RK07	0.0	0

RK06K-EF Error free 14 megabyte data cartridge for RK06 subsystems.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
RK06	0.0	0

RK06K-AC 14 megabyte alignment cartridge for RK06.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
RK06	0.0	0

RK06K-DC 14 megabyte data cartridge for RK06 subsystems.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
RK06	0.0	0

VAX-11/780 UNIBUS Options

ASYNCHRONOUS MULTIPLEXERS (PROGRAMMED I/O)



DZ11-A

Asynchronous 8-line multiplexer for EIA/CCITT terminals or lines. Features programmable speeds (up to 9600 Baud) and formats on a per-line basis. Expandable to 16 lines. Includes data set control for use with Bell 103 or 113 modems or equivalent. BC05D cables are needed for modems. For local connect of EIA/CCITT terminals use BC03M-XX series of cables.

One DZ11-A is included with each VAX-11/780 system.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
Hex slot and SM PAN space	2.2	1

DZ11-B

Eight-line EIA/CCITT expansion multiplexer.

Prerequisite: DZ11-A.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
Hex slot	2.2	1

DZ11-C

Asynchronous 8-line multiplexer for 20mA current loop terminals. Features programmable speeds (up to 9600 Baud) and formats on a per-line basis. Expandable to 16 lines. Use BC04R-12 cables for Digital 20mA terminals.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
Hex slot and SM PAN space	3.0	1

DZ11-D

Eight-line 20mA current loop expansion multiplexer.

Prerequisite: DZ11-C.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
Hex slot	3.0	1

DZ11-E

Asynchronous 16-line multiplexer for EIA/CCITT terminals or lines. Features programmable speeds (up to 9600 Baud) and formats on a per-line basis. Includes data set control for use with Bell 103 and 113 modems or equivalent. BC05D cables are needed for modems. For local connect of EIA/CCITT terminals use BC03M-XX series of cables.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
2 Hex slots and SM PAN space	4.4	2

DZ11-F

Asynchronous 16 line multiplexer for 20mA current loop terminals. Features programmable speeds (up to 9600 Baud) and formats on a per-line basis. Use BC04R-12 cables for Digital 20mA terminals.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
2 Hex slots and SM PAN space	6.0	2

SINGLE LINE SYNCHRONOUS INTERFACES

DMC11-AR Network link DDCMP microprocessor module (remote). DDCMP protocol implemented in firmware for remote operation. Operates full or half duplex. NPR input and output transfers.
Prerequisite: DMC11-DA on DMC11-FA line units.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
Hex slot	5.0	1

DMC11-DA Network link remote line unit module. Interfaces to EIA/CCITT synchronous modems (Bell series 200 compatible) at speeds up to 19,200 bits/second. Operates full or half duplex. Includes data set control for switched network operations. Can be used to communicate over common carrier facilities to another DMC11 or to a synchronous interface with software implementation of DDCMP version 3.2. Includes 25 ft. (7.6m) modem cable.
Prerequisite: DMC11-AR.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
Hex slot (next to DMC11-AR)	3.0	0

DMC11-FA Network link remote line unit module. Interfaces to CCITT V.35/DDS synchronous modems (Bell 500A L1/5 or equivalent) at speeds up to 250,000 bits/second. Includes data set control for full or half duplex, private wire operation. Can be used to communicate over common carrier facilities to another DMC11 or to a synchronous interface with software implementation of DDCMP version 3.2. Includes 25 ft. (7.6m) modem cable.
Prerequisite: DMC11-AR.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
Hex slot (next to DMC11-AR)	3.0	0

DMC11-AL Network link DDCMP microprocessor module (local). DDCMP protocol is implemented in firmware for high speed NPR input and output transfers. One DMC11-AL operates at 1,000,000 bits/second in full duplex mode. Two DMC11-ALs operate at 1,000,000 bits/second in half duplex mode.
Prerequisite: DMC11-MA or DMC11-MD line units.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
Hex slot	5.0	1

DMC11-MA Network Link local line unit module 1,000,000 bits/second. Provides high speed connection to another local DMC11 using coaxial cable up to 6,000 ft. (1,829m) long. (Includes built-in modem). Operates full duplex with two cables and half duplex with a single cable. Cables not included.
Prerequisite: DMC11-AL.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
Hex slot (next to DMC11-AL)	3.0	0



VAX-11/780 UNIBUS Options

DMC11-MD

Network Link local line unit module 56,000 bits/second. Provides high speed connection to another local DMC11 using coaxial cable up to 18,000 ft. (5,487m) long. (Includes built-in modem). Operates full duplex with two cables and half duplex with a single cable. Cables not included.
Prerequisite: DMC11-AL.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
Hex slot (next to DMC11-AL)	3.0	0

BC03N-A0

100 ft. (30.5m) cable for DMC11 line units. (Use Belden cable type 8232 or equivalent for lengths greater than 100 ft. (30.5m). Refer to manual EK-DMCLU-MM-001 for cable connector details.)
Prerequisite: DMC11-MA or DMC11-MD.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
N/A	N/A	N/A



VAX-11/780 Input/Output Devices

LINE PRINTERS

LA11-PA
LA11-PD 132 column, 96 character matrix printer (DIGITAL's LA180 Line Printer) and control unit. 180 characters/second.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
1 Quad slot and FS	1.5	1

LP11-CA
LP11-CD 132 column, 64 character high speed printer and control unit. 900 lines/minute.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
1 Quad slot and FS	1.5	1

LP11-DA
LP11-DD 132 column, 96 character high speed printer and control unit. 660 lines/minute.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
1 Quad slot and FS	1.5	1

LP11-RA
LP11-RB Heavy duty line printer and control unit. 132 columns, 64 characters, 1250 lines/minute.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
1 Quad slot and FS	1.5	1

LP11-SA
LP11-SB Heavy duty line printer and control unit. 132 columns, 96 characters, 925 lines/minute.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
1 Quad slot and FS	1.5	1

LP11-VA
LP11-VD 132 column, 64 character printer and control unit. 300 lines/minute.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
1 Quad slot and FS	1.5	1



VAX-11/780 Input/Output Devices

LP11-WA
LP11-WD

132 column, 96 character printer and control unit. 240 lines/minute.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
1 Quad slot and FS	1.5	1

LP11-YA
LP11-YD

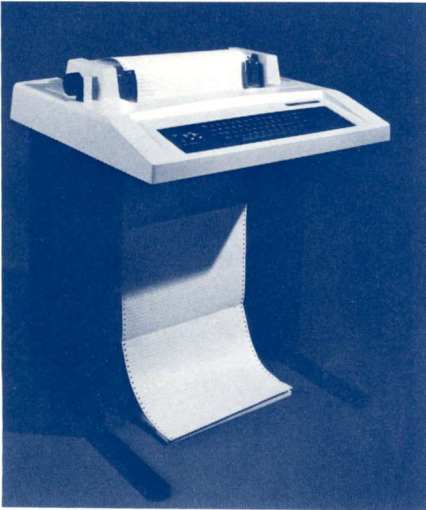
132 column, 64 character printer and control unit. 600 lines/minute.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
1 Quad slot and FS	1.5	1

LP11-ZA
LP11-ZD

132 column, 96 character printer and control unit. 436 lines /minute.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
1 Quad slot and FS	1.5	1



CARD READER

CR11
CR11-A

300 cards/minute reader and control unit. Reads 80-column punched cards.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
1 Quad slot and TT	1.5	1

TERMINALS

LA36-CE
LA36-CJ

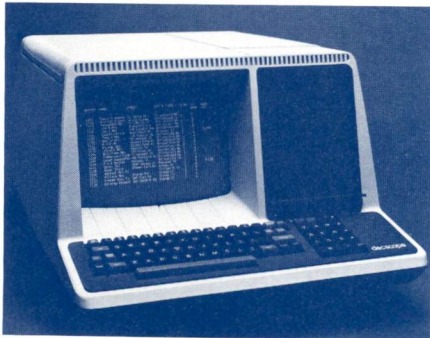
DECwriter II hardcopy terminal with numeric keypad. 30 characters/second, 96 characters, with 20mA current loop interface. *Prerequisite:* DZ11-C or DZ11-F.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
FS	N/A	N/A

LAXX-KG

EIA/CCITT adapter. Allows an LA36 to connect to an EIA/CCITT interface.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
N/A	N/A	N/A



VT52-AA
VT52-AB

Alphanumeric CRT terminal. Switch-selectable parity, 96-character keyboard, 80-column by 24-line display with cursor control. 20mA current loop interface.

Prerequisite: DZ11-C or DZ11-F.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
TT	N/A	N/A

VT52-AE
VT52-AF

Alphanumeric CRT terminal. Switch-selectable parity, 96-character keyboard, 80-column by 24-line display with cursor control. EIA/CCITT interface.

Prerequisite: DZ11-A or DZ11-E.

Mounting Code	+5V Power Drawn	UNIBUS Loads Drawn
TT	N/A	N/A



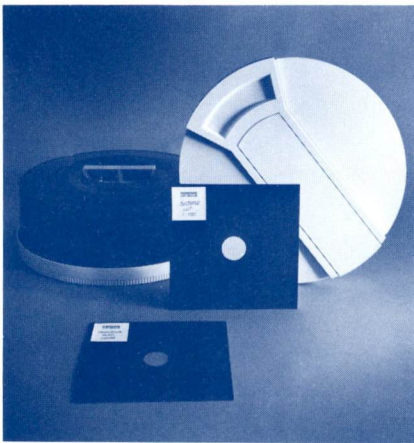
Languages & Utilities for VAX-11/780 Systems

VAX-11 FORTRAN IV-PLUS

DESCRIPTION: VAX-11 FORTRAN IV-PLUS is an optimizing FORTRAN compiler designed to achieve high execution speed. It is based on the ANS FORTRAN X3.9-1966 standard. It's generated code takes advantage of the floating point and character instruction set and the VAX/VMS virtual memory system.

SOFTWARE COMPONENTS: VAX-11 FORTRAN IV-PLUS compiler.

MINIMUM SYSTEM REQUIREMENTS: Any valid VAX/VMS system.



Option Number	Distribution Medium	Support Category
QE100-AY	Floppy Disk	A

PDP-11 COBOL-74/VAX

DESCRIPTION: PDP-11 COBOL-74/VAX is a language processor for business data processing. It is based on the ANS X3.23-1974 standard. The COBOL-74/VAX compiler generates code for compatibility mode.

SOFTWARE COMPONENTS: COBOL compiler and run-time system, report generator and reformat utility programs.

MINIMUM SYSTEM REQUIREMENTS: Any valid VAX/VMS system that includes an LP11 series line printer.

Option Number	Distribution Medium	Support Category
QE101-AY	Floppy Disk	A

PDP-11 BASIC-PLUS-2/VAX

DESCRIPTION: PDP-11 BASIC-PLUS-2/VAX is a superset of the RSTS/E BASIC-PLUS, BASIC-11 IAS-RSX, and Dartmouth BASIC languages. It includes CALL statements, COM or COMMON statements, record I/O, and interactive debugging. The BASIC-PLUS-2/VAX compiler generates code for compatibility mode.

SOFTWARE COMPONENTS: BASIC-PLUS-2 compiler and run-time system.

MINIMUM SYSTEM REQUIREMENTS: Any valid VAX/VMS system.

Option Number	Distribution Medium	Support Category
QE102-AY	Floppy Disk	A

VAX/RSX-11 Development Package

DESCRIPTION: A set of software tools for the development and limited execution of RSX-11M/S task images.

SOFTWARE COMPONENTS: RSX-11M/S SYSGEN, FORTRAN IV/IAS-RSX compiler and run-time system.

MINIMUM SYSTEM REQUIREMENTS: Any valid VAX/VMS system.

Option Number	Distribution Medium	Support Category
QE103-AY	Floppy Disk	A



DECnet-VAX

DESCRIPTION: DECnet-VAX allows a suitably configured VAX/VMS system to participate as a Phase II DECnet node in point-to-point computer networks.

DECnet-VAX offers task-to-task communications, network file transfer, and network resource sharing capabilities using the DIGITAL Network Architecture (DNA) protocols. DECnet-VAX communicates with adjacent nodes over synchronous communication lines.

SOFTWARE COMPONENTS: DECnet-VAX protocol manager.

MINIMUM SYSTEM REQUIREMENTS: Any valid VAX/VMS system with one of the following communication devices: DMC11-AR, -DA; DMC11-AL, -MD; or DMC11-AL -MA.

Option Number	Distribution Medium	Support Category
QED01-AY	Floppy Disk	A

Index

System Code	Page	Option Number	Page
VAX-11/780 SYSTEMS			
SV-AXHHA-LA(LD)	2	REP05-AA(AB)	10
SV-AXCVA-LA(LD)	6	REP05-BA(BB)	10
SV-AXTVA-LA(LD)	4	REP05-DA(DB)	12
Option Number			
VAX-11/780 OPTIONS			
BA11-KE(KF)	14	REP06-AA(AB)	10
BB11	14	REP06-BA(BB)	10
BC03N-A0	20	REP06-DA(DB)	12
CR11-(A)	22	RK611-EA(ED)	15
DD11-CK	14	RK611-FA(FD)	15
DD11-DK	14	RK611-C	17
DMC11-AL	19	RK06-EA(ED)	16
DMC11-AR	19	RK06-FA(FD)	16
DMC11-DA	19	RK06-C	17
DMC11-FA	19	RK06K-AC	17
DMC11-MA	19	RK06K-DC	17
DMC11-MD	20	RK06K-EF	17
DZ11-A	18	RK711-EA(EB)	15
DZ11-B	18	RK711-FA(FB)	15
DZ11-C	18	RK711-C	16
DZ11-D	18	RK07-EA(EB)	16
DZ11-E	18	RK07-FA(FB)	16
DZ11-F	18	RK07-C	16
FP780-AA(AB)	8	RK07K-AC	17
H7112-A(B)	8	RK07K-DC	17
H9602-DF(DH)	14	RK07K-EF	17
H9602-HA(HB)	9	RM03-AA(AD)	10
KU780	8	RM03-BA(BD)	11
LA36-CE(CJ)	22	RM03-C	11
LAXX-KG	22	RM03-P	12
LA11-PA(PD)	21	RP05-AA(AB)	11
LP11-CA(CD)	21	RP05-BA(BB)	11
LP11-DA(DD)	21	RP05-C	12
LP11-RA(RB)	21	RP04-P	12
LP11-SA(SB)	21	RP06-AA(AB)	11
LP11-VA(VD)	21	RP06-BA(BB)	11
LP11-WA(WD)	22	RP06-C	12
LP11-YA(YD)	22	RP06-P	12
LP11-ZA(ZD)	22	RP06-U	12
MS780-AA(AB)	8	TEE16-AE(AJ)	13
MS780-BA	8	TE16-AE(AJ)	13
MS780-BB	8	VT52-AA(AB)	23
MS780-BC	8	VT52-AE(AF)	23
REM03-AA(AD)	9	Option Number	
REM03-BA(BD)	9	LANGUAGES AND UTILITIES	
REM03-DA(DB)	11	QE100-AY	24
		QE101-AY	24
		QE102-AY	24
		QE103-AY	25
		QED01-AY	25

VAX-11/780 Systems & Options Summary

Insert for Prices & Index

VAX11 780

April 1978

PRICE

Purchase prices are stated in U.S. dollars, FOB DIGITAL plant, and apply only in the continental United States. Federal, state, and local taxes are not included. All prices and specifications are subject to change without notice. The fully supported systems described in the VAX-11/780 Systems & Options Summary, April 1978, include a Category A license for the operating system.

LICENSE FEE

The license fee for the software product on a single computer system, in U.S. dollars. Included are the binaries (and/or sources) on the specified medium, services, and documentation as specified in the DIGITAL Software Product Description.

FIELD SERVICE MONTHLY MAINTENANCE

This column lists the Field Service 8 hour/5 day monthly maintenance charge for a Basic Service Agreement. It includes all parts and labor required for system maintenance, plus scheduled preventive maintenance based on the specific needs of the equipment, and installation of engineering changes.

FIELD SERVICE INSTALLATION

This column lists the Field Service fixed price charge for performing field add-on installation of additional equipment to already installed systems.

PAGE

Refers to the corresponding page in the VAX-11/780 Systems & Options Summary, April 1978.

NOTE: (*) means to contact DIGITAL for prices.

digital

VAX11 780

System Code	Price(\$)	FS Monthly Maintenance	Page	Option Number	Price(\$)	Field Maint	Serv Instl	Page
VAX-11/780 SYSTEMS				REP06-AA(AB)	44,000	220	1,145	10
SV-AXCVA-LA(LD)	185,000	832	6	REP06-BA(BB)	56,600	270	1,165	10
SV-AXHHA-LA(LD)	128,600	692	2	REP06-DA(DB)	14,700	50	1,165	12
SV-AXTVA-LA(LD)	153,000	722	4	RK611-EA(ED)	11,500	108	380	15

Option Number	Price(\$)	Field Maint	Serv Instl	Page
---------------	-----------	-------------	------------	------

ADD-ON OPTIONS FOR VAX-11/780 SYSTEMS

BA11-KE(KF)	2,420	16	120	14
BB11	187	N/C	100	14
BC03N-A0	121	N/C	N/A	20
CR11-(A)	6,170	53	280	22
DD11-CK	330	N/C	100	14
DD11-DK	660	N/C	100	14
DMC11-AL	1,520	13	175	19
DMC11-AR	1,520	13	175	19
DMC11-DA	850	6	145	19
DMC11-FA	850	6	145	19
DMC11-MA	850	6	145	19
DMC11-MD	850	6	145	20
DZ11-A	2,310	25	195	18
DZ11-B	1,710	21	155	18
DZ11-C	2,310	25	195	18
DZ11-D	1,710	21	155	18
DZ11-E	3,740	46	275	18
DZ11-F	3,740	46	275	18
FP780-AA(AB)	9,900	45	335	8
H7112-A(B)	1,145	10	343	8
H9602-DF(DH)	2,300	N/C	N/A	14
H9602-HA(HB)	3,900	N/C	N/A	9
KU780	10,000	50	260	8
LA36-CE(CJ)	2,100	19	125	22
LAXX-KG	65	N/C	110	22
LA11-PA(PD)	3,770	55	125	21
LP11-CA(CD)	24,000	185	260	21
LP11-DA(DD)	25,700	185	260	21
LP11-RA(RB)	38,470	185	290	21
LP11-SA(SB)	42,900	185	290	21
LP11-VA(VD)	11,800	95	260	21
LP11-WA(WD)	14,050	95	260	22
LP11-YA(YD)	18,900	108	260	22
LP11-ZA(ZD)	20,500	108	260	22
MS780-AA(AB)	22,500	70	338	8
MS780-BA	8,000	30	281	8
MS780-BB	13,000	60	325	8
MS780-BC	22,000	120	377	8
REM03-AA(AD)	25,000	170	865	9
REM03-BA(BD)	33,000	215	1,110	9
REM03-DA(DB)	8,000	45	700	11
REP05-AA(AB)	40,950	220	1,145	10
REP05-BA(BB)	53,550	270	1,165	10
REP05-DA(DB)	14,700	50	1,165	12

RK611-FA(FD)	19,000	148	405	15
RK611-C	10,450	40	285	17
RK06-EA(ED)	7,500	78	335	16
RK06-FA(FD)	11,000	78	335	16
RK06-C	3,850	10	175	17
RK06K-AC	995	N/A	N/A	17
RK06K-DC	249	N/A	N/A	17
RK06K-EF	349	N/A	N/A	17
RK711-EA(EB)	14,500	145	425	15
RK711-FA(FB)	22,000	190	495	15
RK711-C	10,450	45	295	16
RK07-EA(EB)	10,500	115	380	16
RK07-FA(FB)	14,000	130	405	16
RK07-C	3,850	15	250	16
RK07K-AC	1,295	N/A	N/A	17
RK07K-DC	325	N/A	N/A	17
RK07K-EF	425	N/A	N/A	17
RM03-AA(AD)	19,000	140	495	10
RM03-BA(BD)	21,000	155	680	11
RM03-C	2,000	15	435	11
RM03-P	595	N/A	N/A	12
RP04-P	600	N/A	N/A	12
RP05-AA(AB)	31,400	190	655	11
RP05-BA(BB)	36,540	210	900	11
RP05-C	5,150	20	900	12
RP06-AA(AB)	34,000	190	655	11
RP06-BA(BB)	39,140	210	900	11
RP06-C	5,150	20	900	12
RP06-P	750	N/A	N/A	12
RP06-U	10,500	N/A	N/A	12
TEE16-AE(AJ)	18,850	120	660	13
TE16-AE(AJ)	11,290	60	315	13
VT52-AA(AB)	1,900	20	125	23
VT52-AE(AF)	1,900	20	125	23

Option Number	License Fee(\$)	Page
---------------	-----------------	------

LANGUAGES & UTILITIES

QE100-AY	3,300	24
QE101-AY	7,700	24
QE102-AY	4,400	24
QE103-AY	1,500	25
QED01-AY	2,700	25



DIGITAL EQUIPMENT CORPORATION, Corporate Headquarters: Maynard, Massachusetts 01754, Telephone: (617)897-5111 — SALES AND SERVICE OFFICES: UNITED STATES—ALABAMA, Huntsville • ARIZONA, Phoenix and Tucson • CALIFORNIA, El Segundo, Los Angeles, Oakland, Ridgecrest, San Diego, San Francisco (Mountain View), Santa Ana, Santa Clara, Stanford, Sunnyvale and Woodland Hills • COLORADO, Englewood • CONNECTICUT, Fairfield and Meriden • DISTRICT OF COLUMBIA, Washington (Lanham, MD) • FLORIDA, Ft. Lauderdale and Orlando • GEORGIA, Atlanta • HAWAII, Honolulu • ILLINOIS, Chicago (Rolling Meadows) • INDIANA, Indianapolis • IOWA, Bettendorf • KENTUCKY, Louisville • LOUISIANA, New Orleans (Metairie) • MARYLAND, Odenton • MASSACHUSETTS, Marlborough, Waltham and Westfield • MICHIGAN, Detroit (Farmington Hills) • MINNESOTA, Minneapolis • MISSOURI, Kansas City (Independence) and St. Louis • NEW HAMPSHIRE, Manchester • NEW JERSEY, Cherry Hill, Fairfield, Metuchen and Princeton • NEW MEXICO, Albuquerque • NEW YORK, Albany, Buffalo (Cheektowaga), Long Island (Huntington Station), Manhattan, Rochester and Syracuse • NORTH CAROLINA, Durham/Chapel Hill • OHIO, Cleveland (Euclid), Columbus and Dayton • OKLAHOMA, Tulsa • OREGON, Eugene and Portland • PENNSYLVANIA, Allentown, Philadelphia (Bluebell) and Pittsburgh • SOUTH CAROLINA, Columbia • TENNESSEE, Knoxville and Nashville • TEXAS, Austin, Dallas and Houston • UTAH, Salt Lake City • VIRGINIA, Richmond • WASHINGTON, Bellevue • WISCONSIN, Milwaukee (Brookfield) • INTERNATIONAL—ARGENTINA, Buenos Aires • AUSTRALIA, Adelaide, Brisbane, Canberra, Melbourne, Perth and Sydney • AUSTRIA, Vienna • BELGIUM, Brussels • BOLIVIA, La Paz • BRAZIL, Rio de Janeiro and Sao Paulo • CANADA, Calgary, Edmonton, Halifax, London, Montreal, Ottawa, Toronto, Vancouver and Winnipeg • CHILE, Santiago • DENMARK, Copenhagen • FINLAND, Helsinki • FRANCE, Lyon, Grenoble and Paris • GERMAN FEDERAL REPUBLIC, Cologne, Frankfurt, Hamburg, Hannover, Munich, Nuremberg, Stuttgart and West Berlin • HONG KONG • INDIA, Bombay • INDONESIA, Djakarta • IRELAND, Dublin • ITALY, Milan, Rome and Turin • IRAN, Tehran • JAPAN, Osaka and Tokyo • MALAYSIA, Kuala Lumpur • MEXICO, Mexico City • NETHERLANDS, Utrecht • NEW ZEALAND, Auckland and Christchurch • NORWAY, Oslo • PUERTO RICO, Santurce • SINGAPORE • SPAIN, Madrid • SWEDEN, Gothenburg and Stockholm • SWITZERLAND, Geneva and Zurich • UNITED KINGDOM, Birmingham, Bristol, Epsom, Edinburgh, Leeds, Leicester, London, Manchester and Reading • VENEZUELA, Caracas •

digital

COMPANY
CONFIDENTIAL

DRAFT

INTEROFFICE MEMORANDUM

TO: District Managers
cc: Regional Managers
Area Managers
Product Line Managers
Marketing Committee

DATE: December 8, 1977
FROM: Win Hindle
DEPT:
EXT: 2338
LOC/MAIL STOP: ML5/A53

SUBJ: VAX vs. DEC 10/20 POSITIONING

For years we have had a gap between the DECsystem-10 and the 11 families. VAX and the DECsystem-20 have closed that gap, but they overlap in price and performance. This overlap puts a special burden on marketing and sales management to make these products clear to our customers and sales representatives, so that we will offer the correct system to each user. The overlap occurs in systems with prices between \$130K and \$350K. Enclosed is a one page overview to help you direct our sales effort. This has been prepared by the product managers involved with these products and approved by the Marketing Committee.

We need your help to make sure we have the right sales representative proposing the right product to each customer. This will require your active and early involvement in situations where confusion could occur. I would like each of you to be involved personally to be sure that we act as one company.

The attached overview is our best attempt to help you, but it may not answer all of your questions. I will plan to issue this guideline again in the future as we get further suggestions and improve it. Please use the District Managers' report or communicate directly with me if you need further clarification of the guidelines. I will make sure that you receive quick, clear answers to the questions that arise. We want to surface the issues quickly and reduce any potential conflict. All of our energy should go to winning each sale.

At the Stockholders' Meeting, Ken was asked about the future of our large computer family. He commented that we are working on a small 20 and will announce it when the product is ready. He also said that our large computer family was started over 15 years ago and that we expected it to continue an equal number of years in the future.

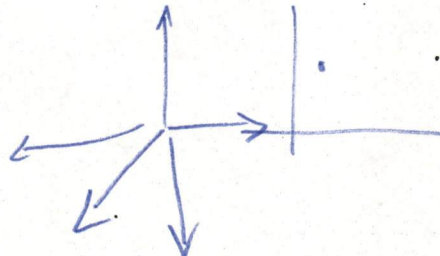
There have been benchmarks on VAX and the 20 which show the following:

1. Single program environments
 - FORTRAN: VAX is between the 2040 and 2050
 - COBOL: VAX - is 10% to 20% faster than IAS or RSTS/E
 - a factor of 4 slower than 2040
 - a factor of 8 - 16 slower than 2050
2. Multi-job environments

The above numbers tend to become more favorable for the DEC 10/20 with increasing number of simultaneous jobs.

bwf
Attachment

- APL
- Basic
- Cobol Bliss



mini ↔ gp (potential) ↔ gp realized

GUIDELINES FOR LEADS BETWEEN \$130K AND \$350K PER CPU

addressing space

1) CUSTOMER

- | | <u>PROPOSE
PDP-11 or VAX</u> | <u>PROPOSE
DEC 10/20</u> |
|---|----------------------------------|------------------------------|
| - Customer is familiar with PDP-11 Systems..... | X | |
| - Customer is familiar with DEC 10/20..... | | X |
| - Sophisticated hardware computer-nik..... | X | |
| - Interested in getting varying jobs done through automatic system optimization rather than having system manager optimize job mix..... | | X |
| - Primarily interested in Fortran, Basic and Assembler..... | X | |
| - Primarily likes Cobol and has an IBM background..... | | X |
| - Primarily speed and performance oriented..... | X | |
| - Comfort and functionality oriented..... | | X |
| - Interested in downward compatibility..... | X | |
| - Interested in expandability of system into network..... | X | |
| - Interested in expandability of his computer system upwards..... | | X |

2) APPLICATION

- | | | |
|--|---|---|
| - Real-time oriented application..... | X | |
| - Scientific application..... | X | |
| - Terminal intensive application..... | X | |
| - Dedicated application and/or predictable job mix..... | X | |
| - Fortran or Basic performance is key..... | X | |
| - First application of a network (with minis)..... | X | |
| - Transaction processing oriented..... | X | |
| - Heavily Cobol oriented application..... | | X |
| - Multi-job mix (<u>general purpose computing</u> with batch and time-sharing)..... | | X |
| - No dedicated applications..... | | X |
| - Conversion requirements from mainframe..... | | X |
| - Substantial APL or some PLI requirements..... | | X |

Cost / Performance oriented!
Perf.
OEM.

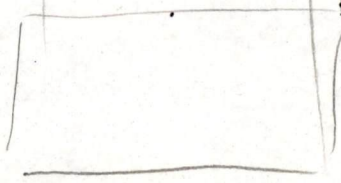
3) COMPETITION

- | | | |
|---|---|---|
| - Mainly mainframe..... | | X |
| - Mainly mini-computer companies..... | X | |
| - HP 3000..... | X | X |
| - Batch Cobol benchmarks needed..... | | X |
| - Fortran, Basic benchmarks needed..... | X | |

4) CUSTOMER FUTURE NEEDS

- | | | |
|--|---|---|
| - Customer plans to build network with minis..... | X | |
| - Customer plans to significantly expand system..... | | X |

- Off load scientific with mini
- Educ. - a mini or 10 or 20



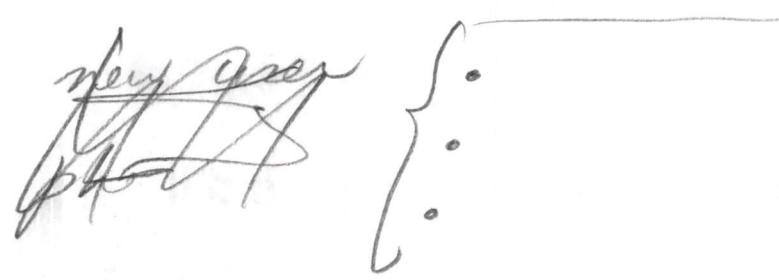
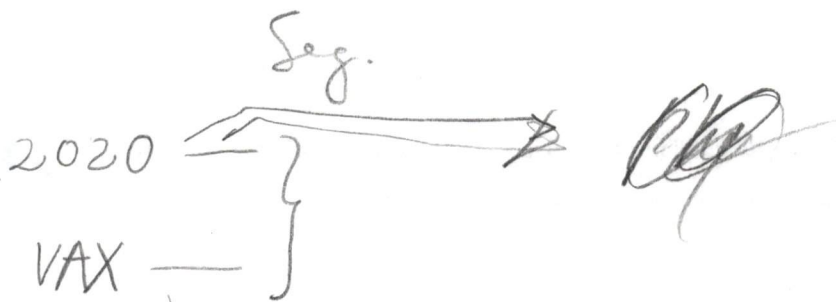
Profession-
based
tools

COMMUNICATIONS OF THE ACM *A Publication of the Association for Computing Machinery*

1133 AVENUE OF THE AMERICAS NEW YORK, NEW YORK 10036 212 265-6300



R. L. ASHENHURST, Editor-in-Chief
MARK S. MANDELBAUM, Executive Editor



- VAX
- ~~new users~~
 - off load 360 / CDC / unum. - large VM
 - features
 - 11's
 - 2020
 - 10 users
 - lots of SW

VAX	- 88K	/ 2Mb
2050	175	/ 2.3Mby
		= 512 Kw



→ GORDON

digital

for info

INTEROFFICE MEMORANDUM

TO: Ron Spinek
cc: Neil Federspiel
Bill Lewis
Roger Strickland
Bob Trocchi
SUBJ: VAX vs. 2020

Admunt

DATE: 27 MAR 1978
FROM: Dick Belmont *DB|nmj*
DEPT: EPG
EXT: 9688
LOC/MAIL STOP: PK3-1/M40

MAY 16 1978

It seems that many of us at EPG, and throughout DEC for that matter, have been spinning our wheels trying to position VAX and the 2020. The positioning of these two products have varied all over the spectrum depending on which article you read, when it was written, and who wrote it. So much so, that even the corporate sales office will ask "which position do you want us to take today?", when talking to a customer. No wonder our EPG sales force is confused. I have been personally involved in two of these confusing sales situations in the past month, both of which DEC now stands a good chance of losing, and I feel that it is about time that EPG takes a firm stand on the position of these products. (Right or wrong!!)

It does not seem like a difficult thing to do. The parameters are few, and the facts are quite clear. I have heard just about every parameter discussed, from delivery date to the color of the cabinet, in the past few weeks, but as far as I am concerned, there are only three things to consider. Price, performance, and functionality. If a prospect is satisfied with these three items, everything else is negotiable. ie: a September rather than a July delivery will not cost us an order if we meet the three criteria mentioned. Let us now discuss the three parameters and see what facts we know about them today.

Our competition loves that approach!

1. Price (See attached examples)

In a real world environment, the VAX system will be from 15% to 50% or more less expensive than a comparable 2020 hardware/software configuration.

Conclusion

If a 2020 is beyond the customer's budget, then VAX or 11/70 should be considered. No need to go any further.

2. Functionality (Software)

For the sake of simplicity, I will list only the major software functionality that VAX has to offer today.

FORTRAN IV PLUS, COBOL-11, SORT-11, BASIC PLUS II, MACRO, QUERY (DATATRIEVE-11), MULTI-KEY ISAM.

If the prospects functionality requirement go beyond those listed, ie: DBMS, APL, PL1, etc. then clearly the 2020 is the proper system to pursue.

For those items listed, VAX is now (FORTRAN IV PLUS, MACRO), or soon will be, vastly superior to the 2020. Even in compatibility mode, COBOL and BASIC are only slightly slower than the 2020, and as the number of users increase, the pendulum swings quickly over to VAX.

Conclusion

If it can be done on VAX, it can be done faster and less expensive than the 2020.

3. Performance

I do not believe that anything need be said about performance that has not already been said, ie: WHETSTONES, U.S. STEEL, TIME-TEST, etc. (See attached U.S. STEEL Benchmark)

Conclusion

VAX is a clear performance winner over the 2020.

Based on the parameters and/or facts mentioned above and in the the attachments, I feel confident in the following statement of positioning the two products.

Positioning Statement

Within the prospect's budget, whenever the 2020 provides customer required functionality not available on the VAX-11/780, then the 2020 is a clear winner. In all other cases, VAX or 11/70 should be the sales person's choice.

VAX vs. 2020 Price Comparison

Typical BASIC System

<u>2020</u>		<u>VAX</u>
CPU	}	CPU
128KW		256KB
RM03		RM03
TU45		TE16
DZ11		DZ11
TOPS-20		VAX-VMS
	149,500	153,000

Additions

RM03	19,000
128KW	30,000
COBOL	}
FORTRAN	
SORT	19,750
LP05	16,500
	\$234,750 .

Additions

RM03	19,000
256KB	30,000
COBOL	7,700
FORTRAN	3,300
SORT	3,300
LP11-VA	11,800
	\$207,800

Maintenance 1,952

Maintenance 743 P/M

VAX vs. 2020 Price Comparison

Typical Large System

2020

CPU
128KW
RP06
TU45
DZ11
TOPS-20 } \$164,500

VAX

CPU
512KB
RP06
TE16
DZ11
VAX-VMS } \$185,000

Additions

RP06 34,000
TU45 14,700
384KW 80,000

COBOL }
SORT } 19,750
FORTRAN }

LP20-D 35,385
(2)DZ11-AA 4,610

352,945

Maintenance 2,743

RP06 34,000
TE16 11,290
512KB 22,000

COBOL } 7,700
SORT }
FORTRAN } 3,300

LP-11ZA 23,500
DZ11-E 3,740

290,530

Maintenance 1,356

INTEROFFICE MEMORANDUM**TO:** Distribution**DATE:** 21 February 1978**FROM:** Marty Jack *WJ***DEPT:** TPS & Languages**EXT:** 4062**CC:** Software Eng. Technical Archive **LOC/MAIL STOP:** ML5-5/B35
ML8/B39**SUBJ:** U.S. Steel Benchmark vs. VAX-11 COBOL-79

The U.S. Steel Benchmark is a highly visible measure of COBOL system performance. Eleven code sequences are executed 100,000 times each. The run time in seconds is then normalized to produce a figure of merit, termed the "Productivity Index", which reflects the relative performance.

I have completed the hand coding and execution of this benchmark for the VAX-11/780. It is believed that the hand generated code is the same as will be produced by the VAX-11 COBOL-79 Release 1 compiler. I must acknowledge the considerable assistance provided by Rodger Blair and by Tom Kent of ESG.

The results are presented below. Data for competitive machines is taken from published U.S. Steel figures (August, 1977). VAX results were obtained on Pilot 1, under VMS 4.XB.

ddl

	<u>145</u>	<u>VAX</u>	<u>155</u>	<u>158</u>
1. DISPLAY arithmetic	95	53	24	15
2. COMPUTATIONAL arithmetic	67	29	24	16
3. Group compare	10	2	4	1
4. Elementary compare	10	2	4	1
5. Move 100 characters	12	4	6	2
6. Fill 100 characters	16	4	8	2
7. Subscripted move	21	11	8	4
8. Conditional string move	150	75	58	39
9. Iterative subscripted move	1,073	593	467	251
10. Straight move	39	3	15	8
11. Edit 132 character line	75	38	21	16
	<hr/>	<hr/>	<hr/>	<hr/>
TOTAL	1,578	816	636	351
"PRODUCTIVITY INDEX" 209408/TOTAL	132.7	256.6	329.3	596.6
Comparative		1.9x145	1.3x780	2.3x780

14.6 x 11/780 C/M

PDP-1 Chapter

G. Bell

The Lincoln Laboratory experimental, transistorized computer, TX-0, was one of the earliest computers to use transistors. TX-0 in turn was related to MIT's Whirlwind I, a computer which was operational in 1950. Whirlwind had a 16-bit word length and can be regarded as the forerunner to the modern minicomputer. This ancestry is uniquely Whirlwind's because it operated much faster (random access to the Williams tube; later the core memory versus having a drum primary memory) and--uniquely--had a short word length (16-bits) *as compared to the* versus having a 32- to 40-bit word length for scientific machines of the von Neumann era. The core memory was developed at MIT by Jay W. Forrester, head of the computer laboratory. The first core memory was exercised under control of a special computer, Memory Test Computer (MTC). Ken Olsen had responsibility for the design of MTC.

TX-0 was designed to test transistor circuitry and to verify that a 256 by 256 (65 Kword) core memory could be built. This work was part of the air defense project that Lincoln ran. It featured a 6 microsecond cycle time and resembled Whirlwind because of the large (12 inch) cathode ray tube and light pen console. TX-0 also had paper tape I/O with a flexowriter typewriter, and two toggle switch registers that the program could access (the first 16 memory words were toggle switches (for variables and/or program)).

Because of the experimental nature of TX-0, it was extremely

→ VAX

→ 2020?

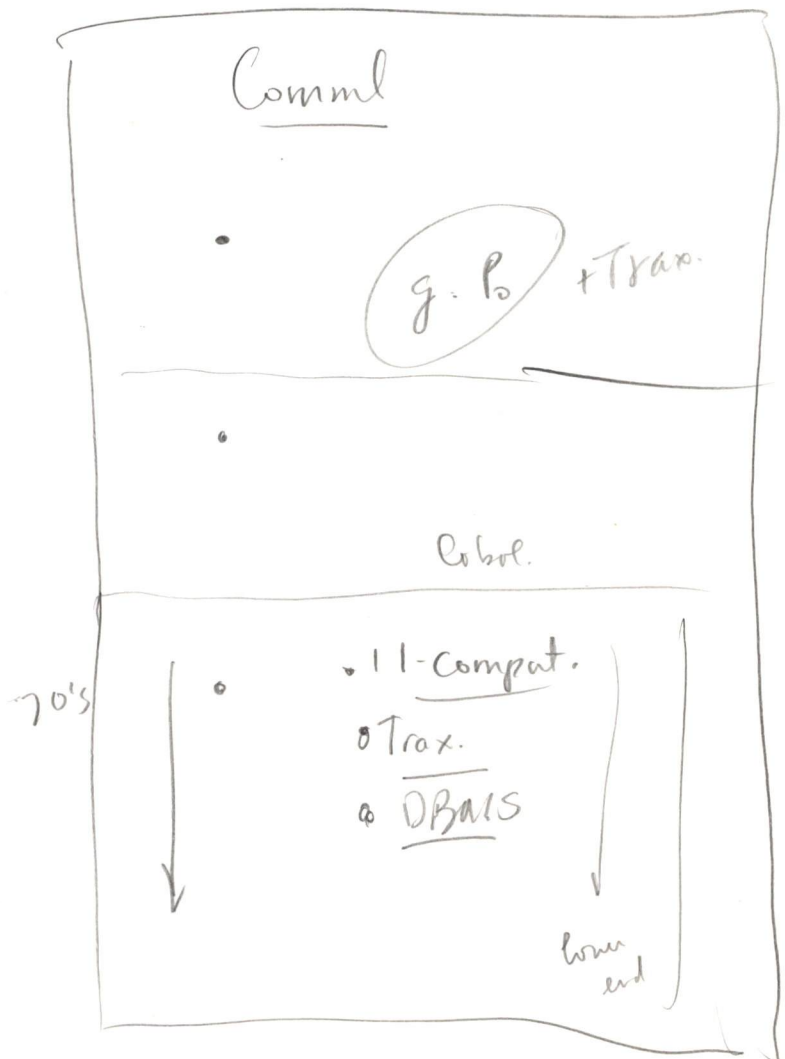
→ DECnet.

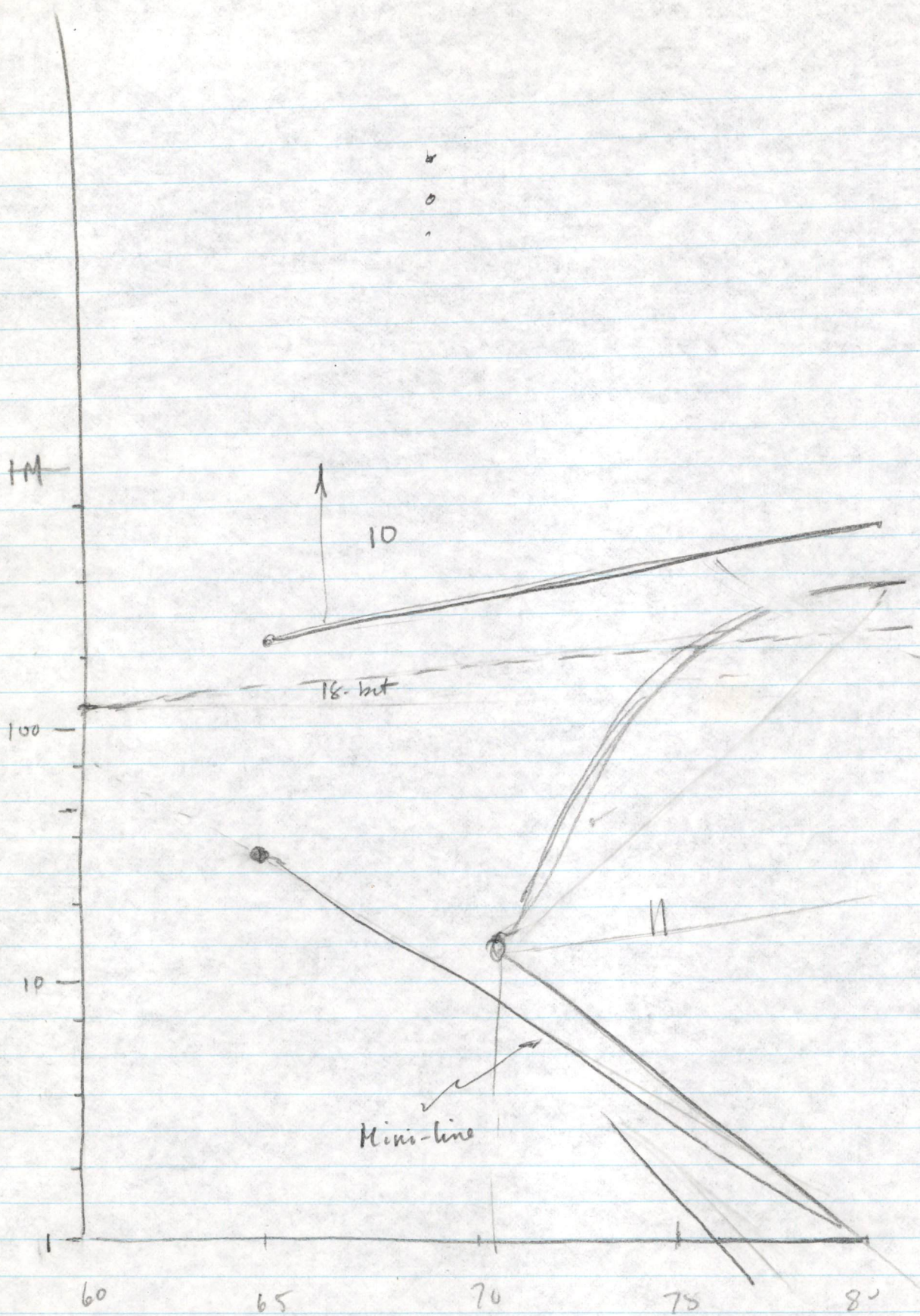
→ mixed WP & Computer.

→ TRAX.

→ low end

→ Part. Data

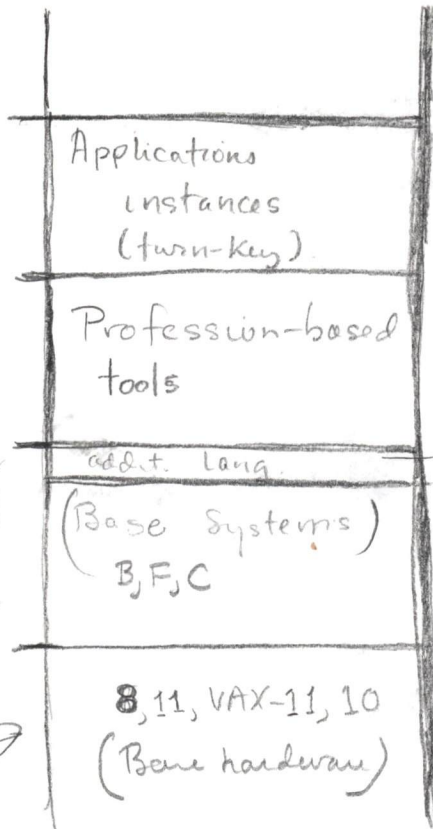




Silly manhood

DECnet

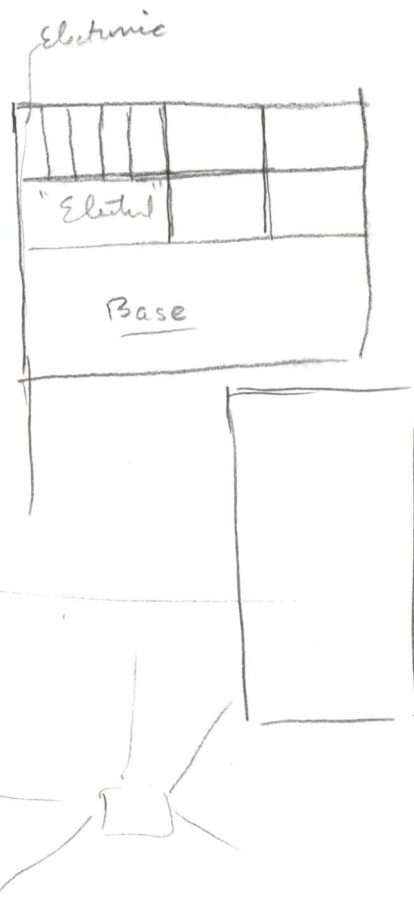
VT78



- APL
- DBMS ^{BLISS}
- Coat
- LISP
- Pascal
- PL/I

eg. DBMS.

Similar
CPMS



IBM's Concept

370

System 34

System 3-15

Series 11.

Discrete
Sub. Product
12 bit

Engineering

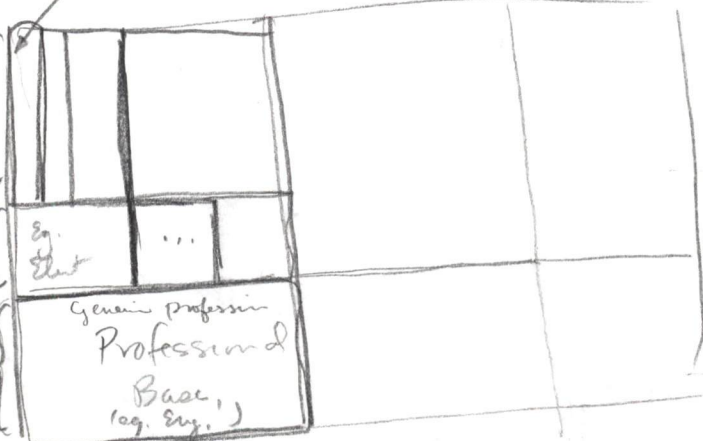
Scient
• Med
• Life science
Eng.

- Legal

- Business

• Arch

Specific product/service engineering
(eg. electronics, mfg, Conf.,
Civil, S)



• generic profession - eg. Engin

• Specific "

• ~~Specific product/service~~
• Industry ^{based} profession

COMMUNICATIONS OF THE ACM *A Publication of the Association for Computing Machinery*

1133 AVENUE OF THE AMERICAS NEW YORK, NEW YORK 10036 212 265-6300



R. L. ASHENHURST, Editor-in-Chief
MARK S. MANDELBAUM, Executive Editor

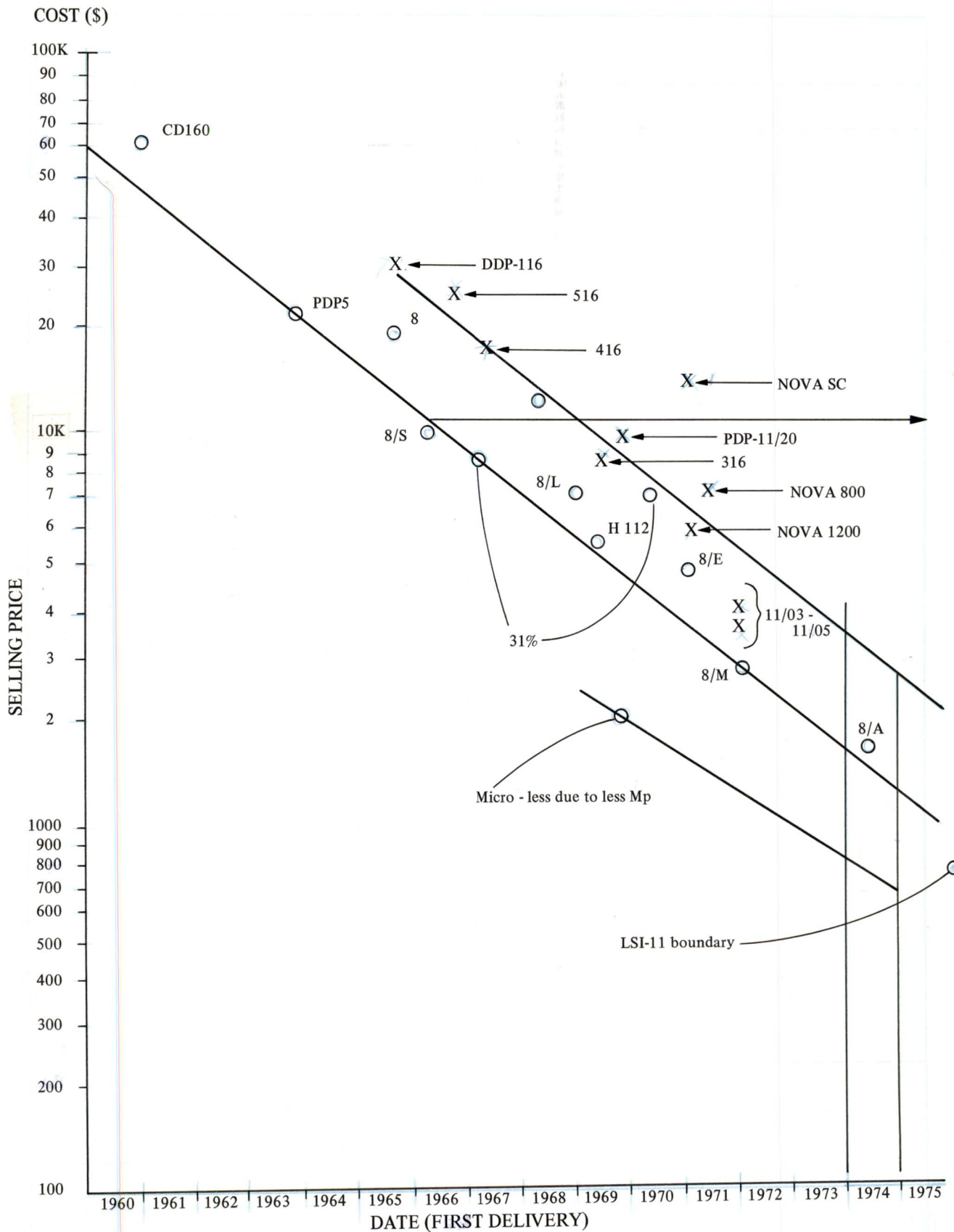


Figure 2 Selling Price versus First Delivery Date for Minicomputers

TO: Marketing Committee
OOD
Product Line Managers
Ulf Fagerquist
Bill Demmer
Jim Marshall
Bill Thompson

DATE: 15 February 1978
FROM: Paul Kampas/Terry Potter
DEPT: SYSTEMS ENGINEERING:
Systems Performance Analysis
EXT: 9649/9749
LOC: ML3-3/E67

CC: Pete van Roekens

SUBJ: 11/VAX/20 POSITIONING REPORT: Part II

FEB 16 1978

Attached to this memo is Part II of the 11/VAX/20 Positioning Series: "Single-User FORTRAN and COBOL Performance for the PDP 11/70, VAX 11/780 and DECsystem-2020, 2040 and 2050" computer systems.

The intent of this report is to provide, at a reasonable level of detail, an understanding of the relative single-user performance characteristics of these machines at THIS POINT IN TIME. This information can be used, for example, by development and/or product line people in such areas as product planning, market positioning, sales training, etc.

It is specifically NOT the intent of this report to make marketing strategy recommendations, to assess the short-term or long-term viability of the products or to predict performance for software or hardware products still in development.

Hardware and software factors contributing to measured performance are given when they are known. In some cases, especially in the area of disk I/O, only some of the factors have been identified. Every effort will be made in future editions to fill such gaps.

If you would like a more detailed presentation to you and your staff on this topic, please contact Paul Kampas at ext. 9649 in Maynard.

SYSTEMS ENGINEERING:
Systems Performance Analysis

Digital Equipment Corporation

Part II: 11/VAX/20 Performance Positioning Report

SINGLE-USER FORTRAN AND COBOL PERFORMANCE

for the

PDP 11/70
VAX 11/780
DECsystem-2020
DECsystem-2040
DECsystem-2050

ABSTRACT: This report presents comparative Single-User performance data for the 11/70 (RSX-11M, RSTS, IAS), VAX 11/780 (VAX/VMS), and DECsystem-2020, 2040 and 2050 (TOPS-20) computer systems. FORTRAN Computation, FORTRAN I/O, COBOL Computation and COBOL I/O measurements are included, each with an analysis of the hardware/software factors that collectively determine the measured performance profiles.

Paul Kampas
ML3-3/E67
Ext. 9649

Date: 6 Feb 78
Edition: 1.0
Expires: 1 May 78

This Report Should be Treated as

C O M P A N Y C O N F I D E N T I A L

POSITIONING OVERVIEW

It is important to note here that we are shooting at MOVING TARGETS! New software releases occur frequently (especially with new products such as VAX and 2020) and therefore performance information can get stale almost before it is out. Reports such as this will "expire" on a quarterly basis, or perhaps sooner if it is deemed necessary.

Complete configuration and software release/version information is given in Sections 2 through 5. In addition please note the following key points:

1. The 11/780 COBOL measurements are using COBOL-11 v3 in compatibility mode, which is presently the only COBOL supported on VAX/VMS.
2. 11/780 measurements were run under VAX/VMS base level 4; At first customer ship time base level 6 is scheduled to be available.
3. The 2040 and 2050 runs were done on 'Model B' processors.

Figure 1 is intended to provide a brief summary of the results obtained in this study. These numbers are for tests using system default parameter settings; performance improvements due to parameter "tuning" are covered in Sections 2 through 5 along with workload descriptions and positioning analysis details. Please be aware that this summary must be used in conjunction with the material in those sections for a complete understanding of the topic.

Finally, this document has been reviewed and approved for distribution by the product managers for their respective organizations.

CAUTION

This document should not be made available to the field in its unabridged form due to the sensitive nature of its contents. Please extract and distribute only data that is pertinent to the need at hand, emphasizing the expiration date information.

Category:	11/70 w: FP-11c			VAX 11/780		DECsystem-20's			For Further Info. See:
	'11M	RSTS	IAS	no FPA	FPA	2020	2040	2050	
FORTRAN Computation:									
- Integer Arithmetic	1.	0.3	1.0	1.0	1.2	0.5	0.8	2.6	Pages 7-11
- Single Prec Flt Point	1.	0.3	1.0	1.0	1.7	0.4	0.8	2.2	
- Double Prec Flt Point	1.	0.2	1.0	0.5	1.5	0.2	0.7	1.6	
FORTRAN Disk I/O:									
- Direct Access	1.	--	0.6	0.8		0.6	1.4	2.1	Pages 12-13
- Sequential Access	1.	--	0.7	0.9		1.0	1.7	4.7	
COBOL Computation:									
- U.S. Steel	1.1	1.	1.0	1.3		4.1	11.	27.	Pages 14-18
- Timetest	1.2	1.	1.0	1.3		1.4	5.5	12.	
COBOL Disk I/O:									
- Sequential Writes	--	1.	0.4	0.5		2.1	5.0	7.6	Pages 19-22
- Sequential Reads	--	1.	0.6	0.9		1.9	3.1	4.6	

Figure 1. Positioning Overview: Relative FORTRAN and COBOL Performance
 <1 → Worse than 11/70 (11/70, '11M = 1 for FORTRAN)
 >1 → Better than 11/70 (11/70, RSTS = 1 for COBOL)

P. Kampas
6 Feb 78

INTRODUCTION

This study was requested by Gordon Bell to provide impartial performance information for the purpose of positioning products across "family" lines. The need for information of this nature has become more critical as the overlap of performance (and functionality) increases between high-end -11's/VAX's and low-end -20's.

Part I of the study was published in November, and compared the hardware features and characteristics as well as design philosophies of the selected machines.

This report, Part II, investigates single-user performance, and includes the following topics:

1. The Meaning of Single-User Performance: A brief discussion of what single-user performance DOES and does NOT tell you about real-life performance.
2. FORTRAN Computational Performance: Integer, Single-Precision and Double-Precision Floating Point performance is evaluated.
3. FORTRAN I/O Performance: Covers Formatted and Unformatted disk I/O for both Sequential and Direct access files.
4. COBOL Computational Performance: Analyzes U.S. Steel and Timetest performance data.
5. COBOL I/O Performance: Sequential access file operations are timed for various record sizes.

Beyond the scope of this report is the topic of Multi-User performance (see Section 1). Based on feedback from the readers of this study and availability of funding, support and machine resources, a multi-user study of a similar nature will be implemented.

Finally, the information presented in this report is the work of many individuals from the development groups and product lines of Digital. Being that this study was essentially unfunded, the time with which these people were so generous is gratefully acknowledged.

1.0 THE MEANING OF SINGLE-USER PERFORMANCE

There are basically three stages of performance measurement applicable to today's complex virtual memory, multi-user interactive/batch computer systems; they are the following:

1. SINGLE-USER: A single job ("benchmark") is run on a system with no other users running. The elapsed time to complete that job is measured.
2. MULTI-STREAM BATCH: A number of different jobs or multiple copies of the same job are submitted in a "batch" (all at once) and the computer executes them one or more at a time (depending on the setting of the "multi-job level") until all are finished. The elapsed time to complete all jobs is measured.
3. MULTI-USER INTERACTIVE/MULTI-STREAM BATCH: In addition to the execution of multi-stream batch jobs, a Remote Terminal Emulator enters pre-defined command "scripts" with controllable think times, typing speeds, etc. as if it were one or more timesharing users. Response times to "user" commands and batch elapsed times are measured.

There are a number of reasons for one to select one stage of analysis over another. As you can see, the more advanced the stage, the more time and resource consuming the task becomes. For a "distributed" project such as this one, this constraint is foremost. It should be also noted that the more complex the measurement environment, the more difficult it is to associate effects with causes, as well as to guarantee the exact same environment on numerous feature-variant machines.

Single-User Performance is a GOOD measure of the following performance factors:

1. HARDWARE Performance (e.g. CPU, disks).
2. Architecture (INSTRUCTION SET) efficiency.
3. Ability of COMPILERS to generate efficient executable code.
4. Efficiency of the Operating System in handling disk FILE I/O.

Single-User Performance is NOT a good measure of the

following factors:

1. The Operating System's primary MEMORY ALLOCATION algorithms.
2. The Operating System's SCHEDULING algorithms.
3. The Operating System's DISK OPTIMIZATION (seek and latency) algorithms.
4. The Operating System's TERMINAL I/O handling overhead.
5. The Operating System's PAGE FAULT handling efficiency.
6. The Operating System's SHARED CODE capabilities.

As you can readily see, the performance factors that single-user measurements don't pick up are those features built into the operating system specifically to support multi-user environments!

It is in the multi-user support area that there are many subtle differences (as well as many similarities) among the various operating systems, especially VAX/VMS and TOPS-20. Some of these differences are being investigated (e.g. paging against oneself as in VMS or against the system as in TOPS-20), but most are yet to be evaluated. In many cases understanding the pluses and minuses of such features takes not only careful study, but critical customer feedback to develop.

2.0 FORTRAN COMPUTATIONAL PERFORMANCE

2.1 Workloads

Approximately 50 FORTRAN compute oriented benchmarks have been run on most of the systems, 27 of which are included in this analysis. For your information, times, descriptions, as well as the source programs themselves are available upon request from Systems Performance analysis. Of those 27 benchmarks chosen, 4 are Integer (Fixed Point), 18 are Single Precision Floating Point, and the remaining 5 are Double Precision Floating Point. Included are most of the "Famous Fifteen" jobs (Hanoi, Airco, Rolls Royce, Whetstones, etc.) for which extensive information is available (also from Systems Performance Analysis) on -11 products not evaluated here as well as competitive machines.

Let me make one comment here about how the benchmarks were timed. Into each job, special "calls" to the high precision timer were inserted. Precise time was recorded before the job began and after it finished, with the elapsed time being the difference. The practice of using "CPU Time" when running on a system with other users and assuming it will closely approximate single-user elapsed time is risky, as every system includes different amounts of system overhead in computing CPU time. All times in this study are elapsed times and were taken on systems with no other active users.

2.2 Software/Hardware Notes

Software release/version information pertaining to the tests made in this section are the following:

Operating Systems:

- IAS v2
- RSX-11m v3.1
- RSTS/E v6b
- VAX/VMS base level 4X9
- TOPS-20 rel 2 and 3

Compilers:

- FORTRAN IV v1 (RSTS only)
- FORTRAN IV PLUS v2
- FORTRAN-20 v5a
- FORTRAN IV PLUS/VAX v77

All systems were configured with adequate memory so that the entire benchmark being run could reside in main memory without paging or overlaying.

Also the 11/780 runs were made with a working set size of 1000 pages so that no paging overhead was incurred.

2.3 Performance Data

The FORTRAN computation timings collected can be presented in many ways; we shall look at two. Figure 2 is designed to show the relative performance ranges for benchmarks doing Integer, Single Precision and Double Precision Floating Point arithmetic across all machines. In order to get ranges for all systems, we cannot normalize the times against any one because its performance is always unity. To circumvent this, performance for each benchmark is normalized against a calculated "average" system. This benchmark average is the simple mean of the individual elapsed times of the 11/70 (RSX-11m), 11/780 (with FPA), 2020, 2040 and 2050. Each benchmark result is indicated by a circle, triangle or asterisk, representing Integer, Single Precision and Double Precision arithmetic jobs respectively. The minimum and maximum relative performance for each range is given at the top of the graph. The 11/70 IAS and RSTS as well as the 11/780 with no FPA (Floating Point Accelerator) columns have fewer data points due the lack of a complete set of timings for these machines.

The median performance (Integer, Single Precision and Double Precision) for each system was determined from Figure 2 and plotted relative to the 11/70 with FP-11c (RSX-11m) in Figure 3. The 11/70 was chosen as the basis of comparison because it has been around the longest, its performance is well understood both here and in the field, and it has about mid-range performance for the group.

2.4 Positioning Analysis

1. The 2050 is the best performer in all three categories, with a large advantage over the 11/780 with FPA in Integer (2.1 to 1.), a lesser advantage in Single Precision (1.3 to 1.), and a small advantage in Double Precision (1.1 to 1.). This relationship is due primarily to hardware implementation factors, with the 2050 using faster ECL logic and general purpose data path, while the 11/780 uses slower TTL logic but dedicates a large amount of special purpose hardware to provide high performance floating point arithmetic.
2. The 11/780 with FPA is only slightly faster than the 11/70 with FP-11c (RSX-11m or IAS) in Integer arithmetic (1.2 to 1.), but significantly faster in Floating Point (about 1.6 to 1. in both Single Precision and Double

max:	.45 .63 .36	1.41 1.67 1.89	1.45 1.74 1.90	1.32 1.64 1.03	1.65 2.95 2.49	67 .55 .43	1.14 1.23 1.19	3.09 3.31 2.88
min:	.15 .18 .35	.83 .90 1.16	.84 .92 1.19	1.04 1.03 .63	1.21 1.66 2.31	.45 .36 .32	.77 .73 .92	2.17 2.17 2.12

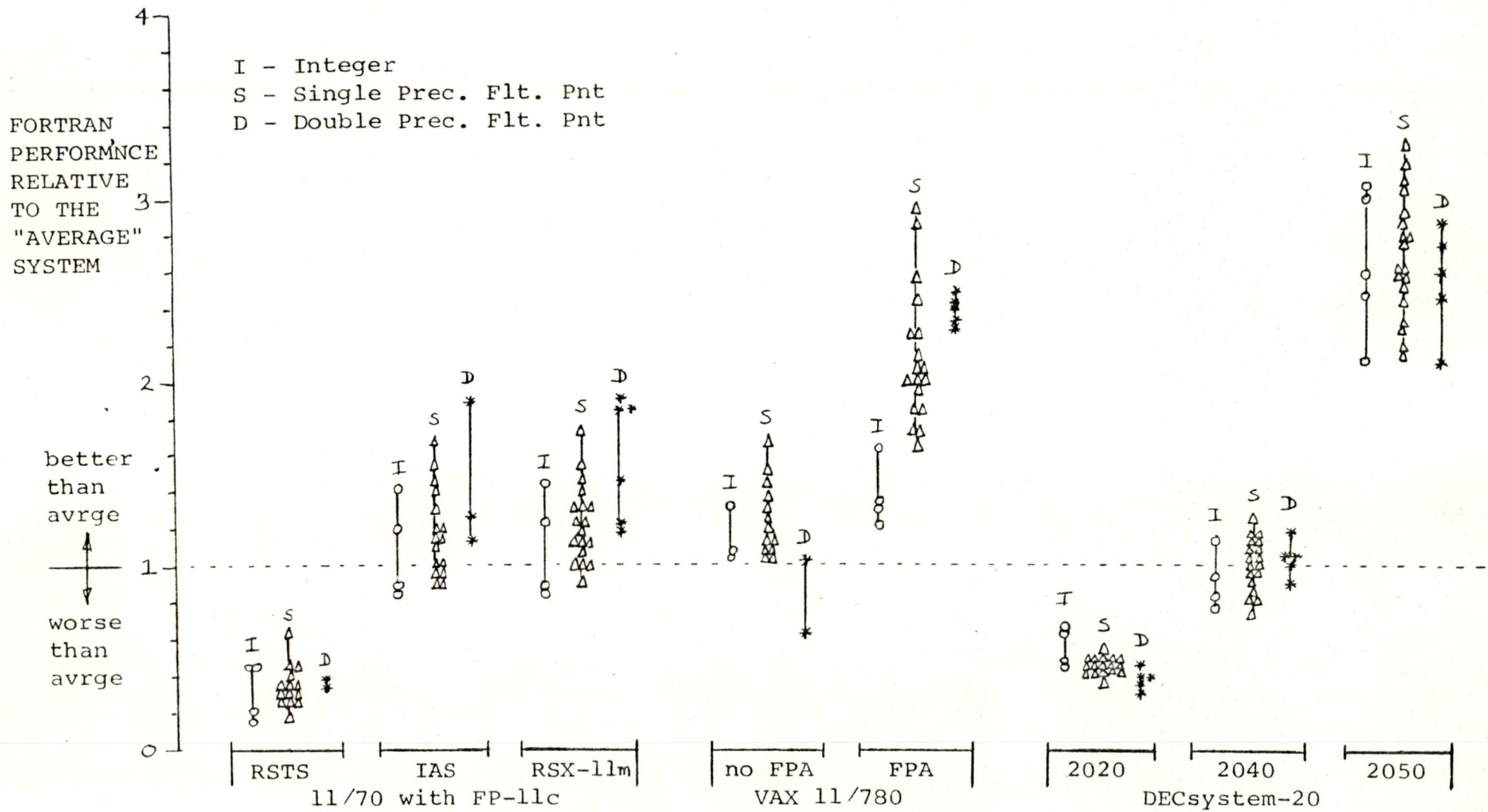
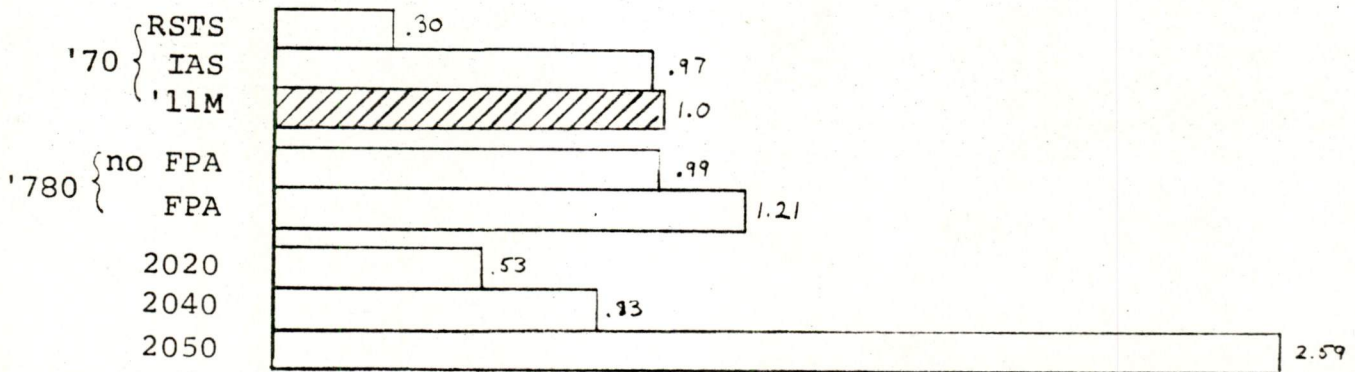


Figure 2: FORTRAN Compute Performance Ranges for 18 Benchmarks

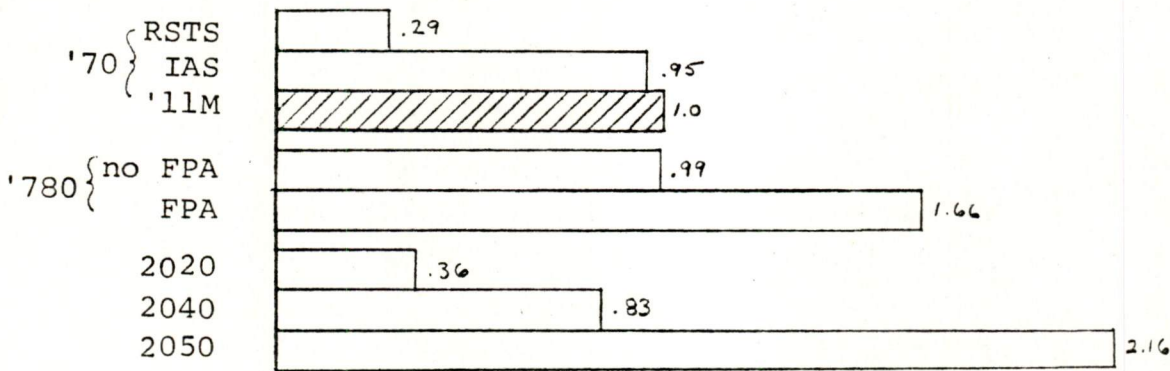
PKampas
24 JAN 78

Figure 3: Integer and Floating Point FORTRAN Computational Performance
(11/70, FP-11c, RSX-11m = 1)

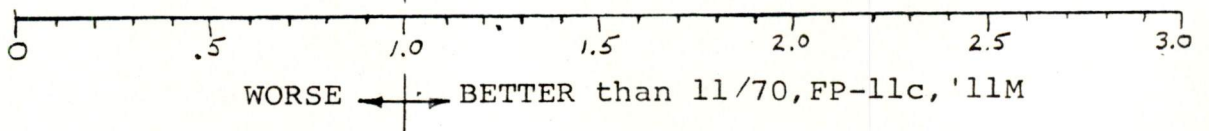
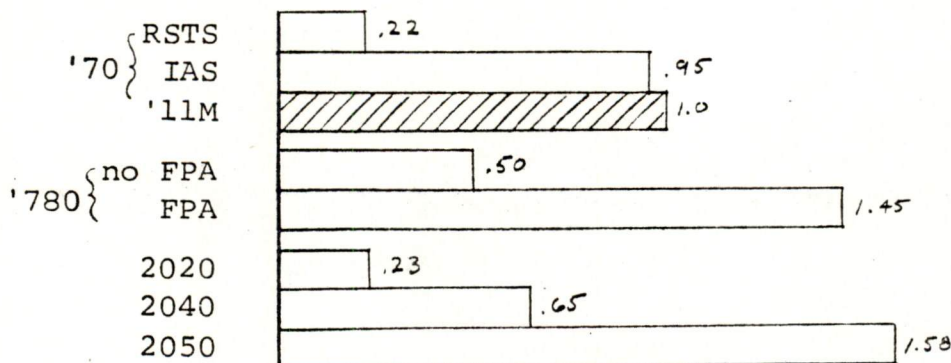
Integer Arithmetic:



Single Prec. Flt. Point Arithmetic:



Double Prec. Flt Point Arithmetic:



P. Komper
25 Jan 78

Precision). Again the 11/780's large investment in floating point hardware pays off.

3. The 11/70 with FP-11c gives vastly reduced performance when using FORTRAN IV (under RSTS) instead of FORTRAN IV PLUS which is supported by RSX-11m and IAS. This is a good example of how good software can make orders of magnitude differences.
4. The 2040 has pretty consistent performance relative to the 11/70 with FP-11c and FORTRAN IV PLUS, being about 0.8 across the board.
5. The 2020 is at its best in Integer arithmetic, being about .65 of a 2040. In Single Precision arithmetic it falls off to .45 of a 2040, and down to .35 for Double Precision. This profile is due mostly to hardware factors, with the 2020 datapath purposely kept simple to keep board counts and costs down. It was noted in Part I of this study that the entire 2020 CPU has the same number of boards as the 11/780 Floating Point Accelerator.
6. The 11/780 without FPA performs just about on par with the 11/70 with FP-11c for Integer and Single Precision Floating Point arithmetic, but only at around half of an 11/70 for Double Precision (note that this figure is based on only two data points). This again reinforces the advantage of a double-word datapath and "heavy-duty" data shifter for good Double Precision performance.
7. Some popular floating point performance metrics are the British Whetstone tests. The "Whetstones per second" values for the systems being discussed as well as some smaller -11's are the following:

	Single Precision	Double Precision
11/70 ('11m, IAS)	699	518
11/780 (no FPA)	588	---
11/780 (FPA)	1041	710
2020	242	103
2040	512	304
2050	1136	667
11/34 FP-11A	202	155
11/34 Cache FP-11A	262	196
11/60 FP-11E	591	435

For those having a copy of the VAX Marketing Guide, the 11/780's Double Precision Whetstone performance was incorrectly reported as 794 Whetstones per second, instead of the actual 710.

You may notice that the Whetstone performance ratios differ somewhat from the overall ratios (which include the Whetstone programs). This is because the Whetstone tests generally have a higher percentage of floating point instructions as compared to the others.

8. All integer benchmarks utilized 16-bit integers. If more precision is needed, the 11/70 can handle 32-bit integers, but incurs a significant performance degradation. This is because the 11/70 integer datapath is only 16-bits wide, and must add the low-order 16-bits, add the carry (if any) to the high-order 16-bits, and then add the high order 16-bits. As a result, the VAX 11/780, 2020, 2040 and 2050 processors will perform noticeably better than the 11/70 (due to their wider datapaths) than shown in Figures 1,2,3 due to their ability to handle the larger integers in a single operation. Actual figures for the amount of this difference will hopefully be available in the next edition of this report.

3.0 FORTRAN DISK I/O PERFORMANCE

3.1 Workloads

Ten routines were run here, each probing the FORTRAN I/O performance of the systems under study. Both Direct (also called Random or Relative) and Sequential Access files were tested, using Formatted and Unformatted data in each. The difference between Formatted and Unformatted data is that Formatted data is generally text (alphanumeric) for human "consumption", while Unformatted data is an internal form (binary) for efficient computation and storage.

3.2 Software Notes

Software release/version information for this section is the same as that previously stated.

3.3 Performance Data

The results of the tests are depicted in Figure 4, with performance normalized to the 11/70 running under RSX-11m. Files of 1000 records were used for this data.

RSTS measurements are not available for this section.

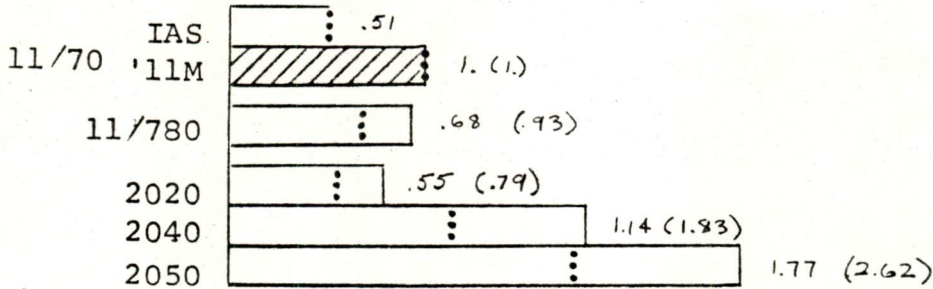
3.4 Positioning Analysis

1. The 11/780 figures are using the current version of RMS-32 which has the system default blocking factor (number of disk blocks read or written for a single file request) set to four (each disk block contains 512 bytes). Tests were also run with eight block grouping, showing an improvement in Sequential Write performance of about 25%.
2. The 2050 has a noticeable advantage in Sequential Access I/O due the large amount of computing needed in file allocation and bookkeeping (more for writes than for reads). For Direct Access I/O the performance of the disk (which is equal for all systems here) appears to be the overriding factor, resulting in a fairly tight overall performance grouping.

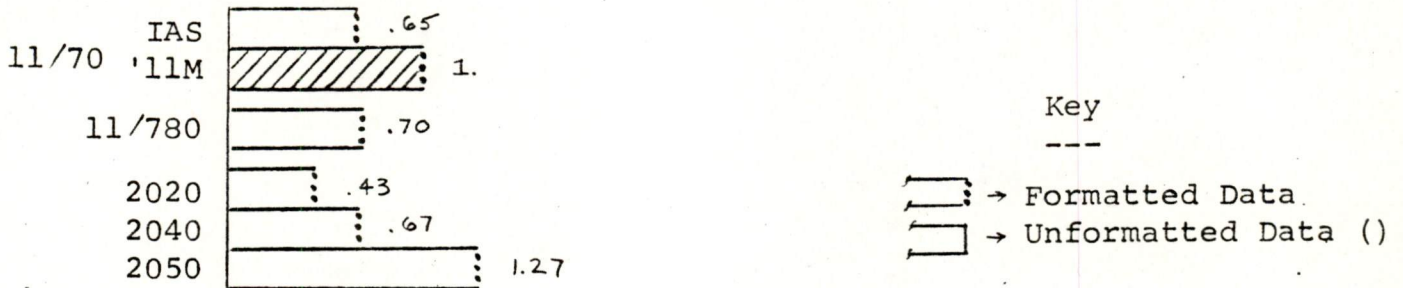
Figure 4: Direct and Sequential Access FORTRAN Disk I/O Performance

(11/70, RSX-11M = 1)

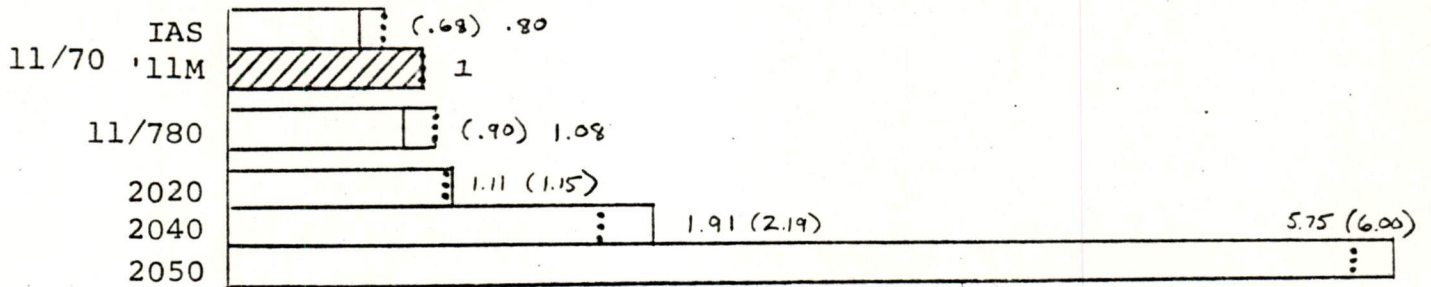
Direct Access Writes:



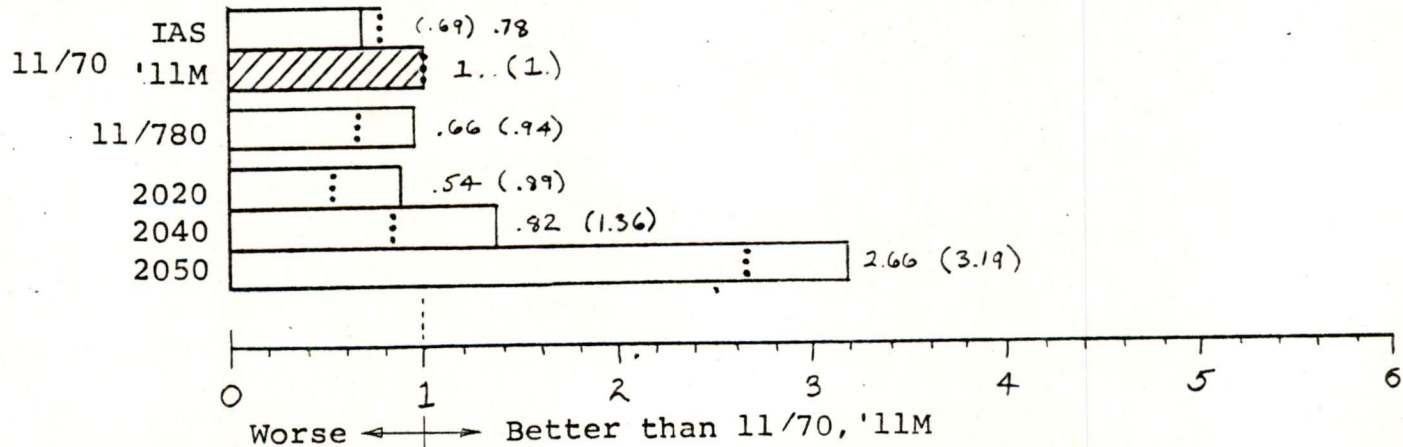
Direct Access Reads:



Sequential Access Writes:



Sequential Access Reads:



4.0 COBOL COMPUTATIONAL PERFORMANCE

4.1 Workloads

Two benchmarks have gained a considerable amount of visibility with COBOL and potential COBOL users. These programs, U.S. Steel and Timetest, are each a collection of eleven routines, both supposedly representing "typical" mixes of COBOL computational operations. Neither benchmark measures (nor does) disk I/O.

These programs were chosen for this study for a number of reasons, including their ready availability, the large number of competitive systems for which timings are known, the fact that new releases of COBOL for 11's and VAX's have their performance goals based on these programs, and finally because many computer "shoppers" put a lot of value in how fast these programs run.

For those not familiar with the U.S. Steel or Timetest "outputs", please read the following:

U.S Steel: calculates a "productivity index" which is simply how many times the system measured is faster than an IBM 1460 system (e.g. a productivity index of 10 means that system is 10 times faster running the U.S. Steel routines than an IBM 1460 is).

Timetest: calculates an "average instruction execution time" which is an average of the times for each of the 11 routines weighted by their expected frequency of occurrence (more highly used statements count more in the average than less used ones).

4.2 Software Notes

Software release/version information pertaining to the tests in this section are the same as those previously stated with the following additions:

COBOL Compilers:

COBOL-11 v3
COBOL-20 v11 and v12

Note that the 11/780 presently uses COBOL-11 in compatibility mode which takes no advantage of the improved VAX instruction set and enlarged address space.

4.3 Performance Data

Since both U.S. Steel and Timetest have a similar intent (to measure performance of a weighted mix of COBOL statements), one would expect the results that they produce to be fairly similar. Surprisingly, the results are very different as Figure 5 testifies. The reasons for this discrepancy are explained below.

4.4 Positioning Analysis

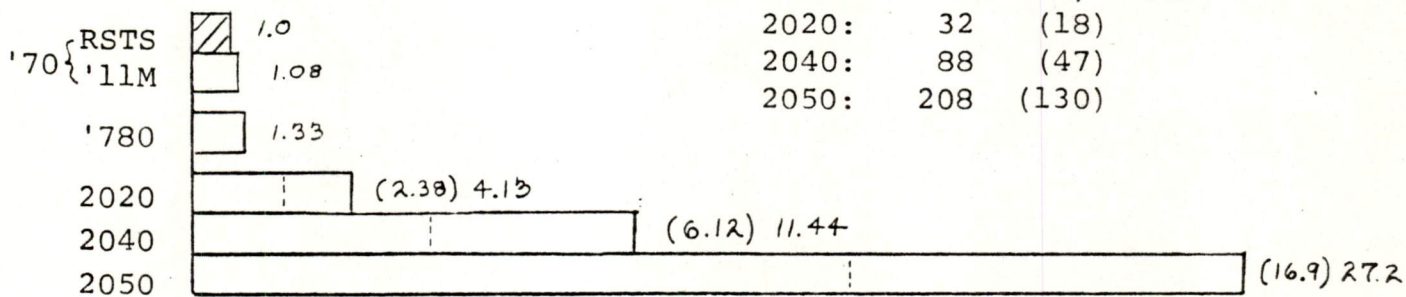
1. U.S. Steel performance on the 2020, 2040 and 2050 is roughly 4, 11 and 27 times that of the 11/70 (RSTS) respectively, with the -20's using v12 COBOL. The 11/780 improves on the 11/70's times by about 30%.
2. Timetest performance ratios for -20's to -11's are a factor of two to three less than for U.S. Steel, with the 2020, 2040 and 2050 being 1.4, 5.5 and 12 times faster than the 11/70 (RSTS) respectively. VAX performance here is only slightly less than that of the 2020.
3. The two primary factors that we have identified as causing this large discrepancy are the following:
 1. Timetest does not align data fields on word boundaries as does U.S. Steel. Since -20's are word addressable (as opposed to the -11's which are byte addressable), moving unaligned data from source to destination requires many costly character manipulation operations instead of an efficient block transfer. When Timetest was modified to use aligned data on the 2020 instead of unaligned, performance of the benchmark was doubled. In fact, one routine which does a 120 byte alphanumeric move, went from 2940 microseconds to 80 microseconds, a 37-fold improvement. Just for comparison, the 11/780 (COBOL-11) time for this operation is 450 microseconds. For the 12 byte and 6 byte alphanumeric moves, the 2020 times improved by 8 and 6 times respectively. Since aligning data is a relatively simple task (except when the data is in a disk file records), it is apparent that it should always be done when benchmarking -20's.
 2. A second factor in the U.S. Steel/Timetest discrepancy is the abundance of subscripted data in U.S. Steel and its total absence in Timetest. COBOL-20 has a decided advantage over COBOL-11 in this area (see below).

Figure 5: COBOL Computational Performance

(11/70, RSTS = 1)

"U. S. Steel" Performance:

Productivity Index:



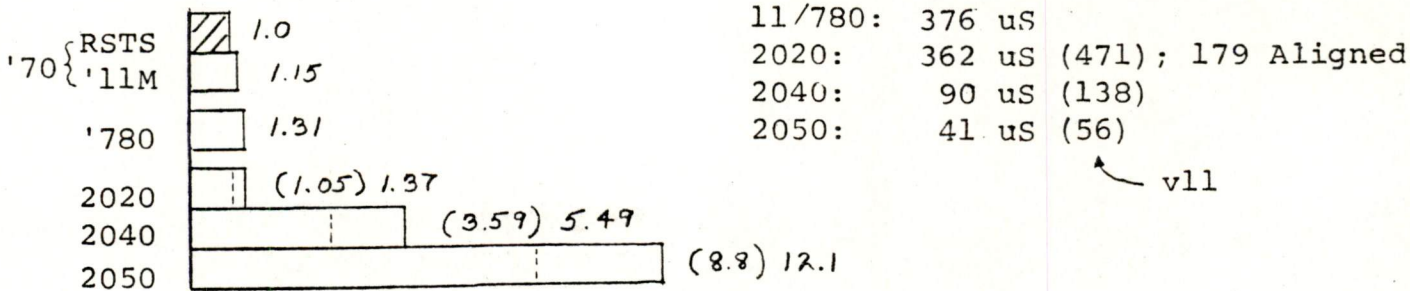
11/70:	8	
11/780:	10	v11
2020:	32	(18)
2040:	88	(47)
2050:	208	(130)

-20 COBOL v11 ()

 -20 COBOL v12

"Timetest" Performance:

Average Stmt Execution Time:



11/70:	497 uS RSTS; 430 uS '11M
11/780:	376 uS
2020:	362 uS (471); 179 Aligned
2040:	90 uS (138)
2050:	41 uS (56)

v11

0 1 5 10 15 20 25 30

Worse ← Better than 11/70, RSTS

4. The 2020 appears to suffer somewhat more than the 2040 and 2050 from the unaligned data of Timetest, incurring a 3:1 performance loss (relative to the 11/70) while the 2040 and 2050 lose at 2:1.
5. The version-12 COBOL compiler for the -20's offers a significant performance improvement over version-11, as the numbers in Figure 5 attest. Briefly, the areas that were improved in version-12 are the following:
 1. The entry and exit code for PERFORM are generated in-line for v12 in most cases. "In-line" means that machine code is generated for that statement which is executed directly without jumping to the Object Time System. In-line code has the advantage that it executes faster (no parameter passing, register saving, etc. is required), but generally makes the program larger because that code is repeated as many times as it appears in the source program and not just once in the Object Time System. COBOL-11 uses jumps to the Object Time System extensively. Improvements to COBOL-11 will reduce this, but the 16-bit address limitation will limit the use of in-line code because of its program size penalties.
 2. All subscript code is generated in-line in v12.
 3. MOVES and COMPARES are done whenever possible with multiple bytes or block transfers.
 4. Double precision ADD, SUBTRACT and MULTIPLY are done in-line in v12 for 2040's and 2050's, and double precision COMPARES are done in-line for all machines.
 5. The SORT/MERGE interface has been made more efficient.
6. SORT performance was not evaluated in this study, but it appears that both -11's and -20's have reputations for good performance in this area. As a result, this factor is most likely not a deciding one for positioning, but nonetheless should be examined at a future date.
7. Also not evaluated here was COBOL compile performance. However, from having compiled the same programs on all systems, it is readily apparent that COBOL-11 compiles quite slowly. The reason for this is quickly determined by looking at the RS0x (fixed-head swapper) lights and seeing that heavy overlaying is occurring. If compile performance is

of interest (such as in a software development environment), this area should be pursued.

8. New versions of COBOL for -11's and VAX are under development and have aggressive performance goals. Beware, however, of performance estimates that are not based on actual program performance; instruction mix predictions can be misleading.

5.0 COBOL DISK I/O PERFORMANCE

5.1 Workloads

Due to time and resource constraints, performance for Sequential Access files only was evaluated. The program used was written by Dave Braithwaite of LCG Commercial Systems Group, and measures read and write times for 1000 record files, varying record sizes from 1 to 1024 characters.

5.2 Software/Hardware Notes

Software release/version information is the same as before.

These tests were run on systems using RP04, RP05 and RP06 disks. Since these devices have nearly identical seek and latency times, the factor of disk hardware performance should be essentially equal for all systems. The number of spindles on the systems being tested is a moot point here, for only one file was read or written at a time and no paging or overlaying was occurring.

RSX-11M data is not available for this section.

5.3 Performance Data

The tests were run on all systems using their most common data format, which is 8-bit ASCII for -11's/VAX and 6-bit ASCII for -20's. Record sizes of 32, 64, 128, 256, 512 and 1024 characters are compared in Figures 6 and 7, which depict Sequential Access write and read performance respectively.

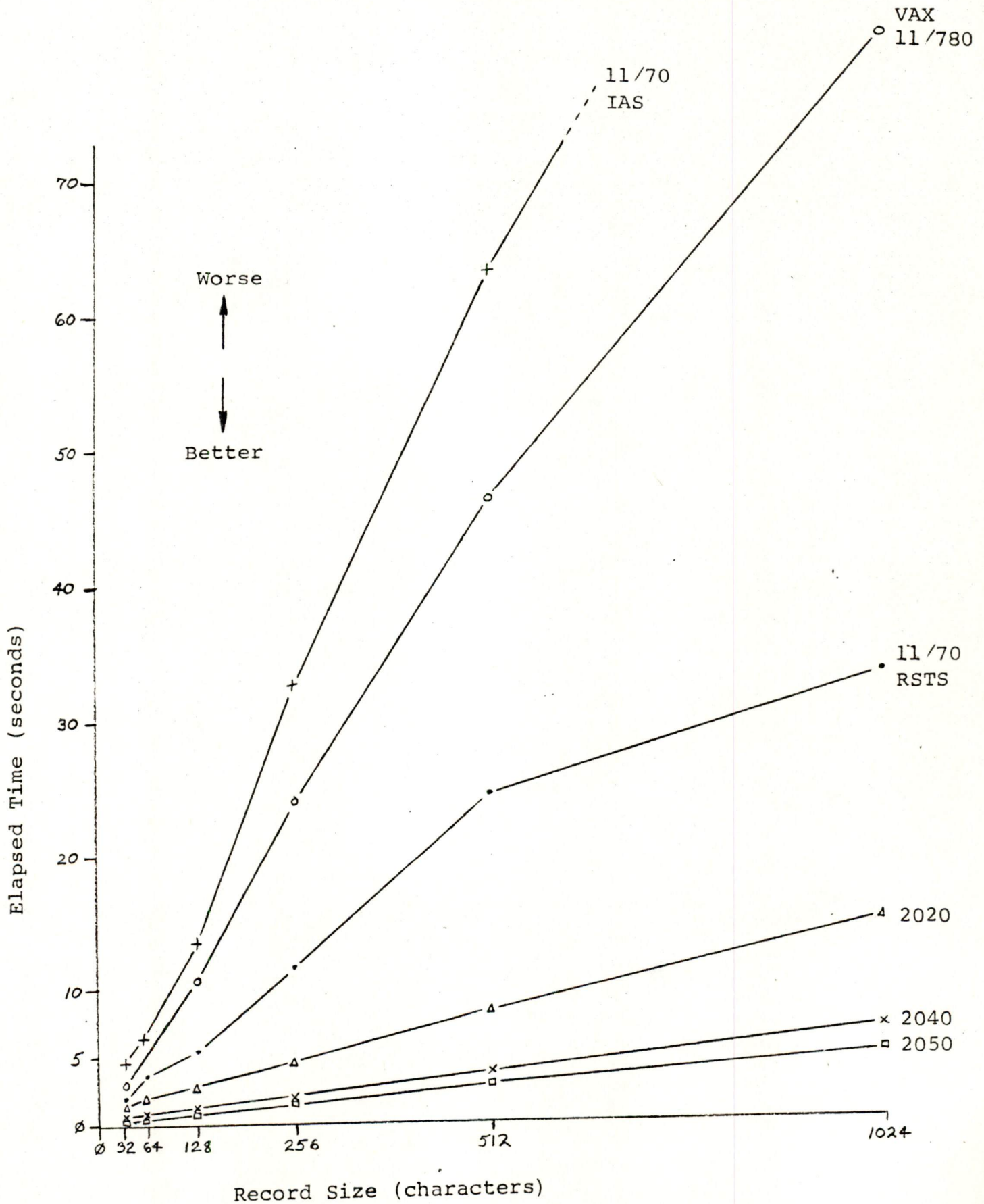
5.4 Positioning Analysis

1. If the relative performance ratios for the systems are calculated based on the 11/70 (RSTS) and averaged over all record sizes, we have the following:

Sequential Access: WRITES

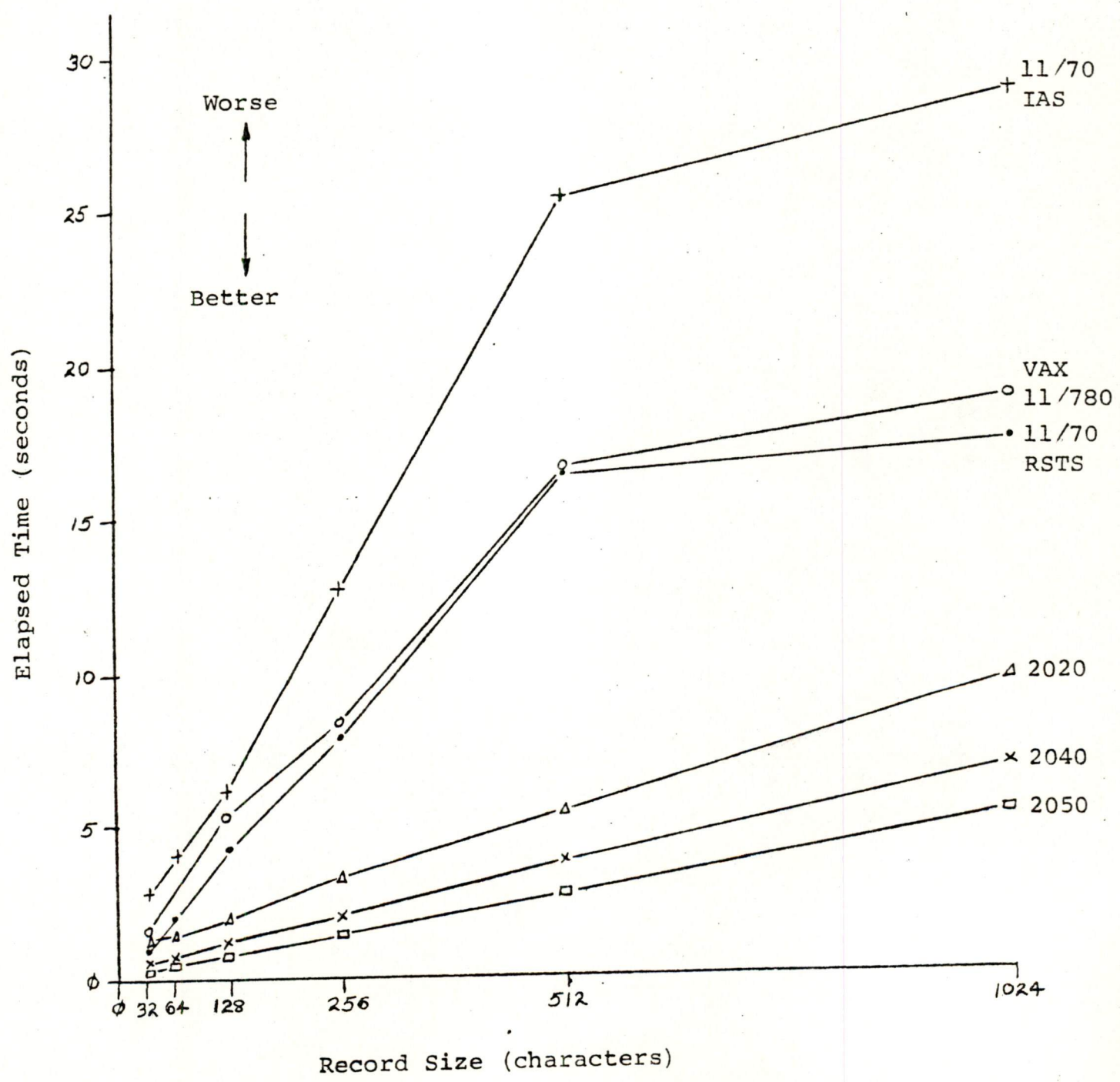
11/70 (IAS)	0.4
11/780	0.5
11/70 (RSTS)	1.0
2020	2.1

Fig. 6. COBOL Disk I/O Performance: Time to WRITE a 1000 Record Sequential Access Disk File



P. Kempster
29 Jan 78

Fig. 7. COBOL Disk I/O Performance: Time to READ a 1000 Record Sequential Access Disk File



P. Kempner
29 Jan 78

2040	5.0
2050	7.6

Sequential Access: READS

11/70 (IAS)	0.6
11/780	0.9
11/70 (RSTS)	1.0
2020	1.9
2040	3.1
2050	4.6

2. One reason why the 11/70 (IAS) and 11/780 perform the slowest is that they do file I/O one block (512 bytes) at a time, whereas the 11/70 (RSTS) and -20's do file I/O 4 blocks at a time. As you would expect, for files larger than 512 bytes a significant performance improvement is possible due to the reduction in disk delays.
3. It is possible on the 11/780 (and others) to request contiguous allocation of disk blocks (the file is placed on physically adjacent disk sectors). Thus, when a program is writing sequentially, the system does not have to go through the allocation procedure each time it hits a sector boundary. Also the data is always on the same cylinder, so a seek is never required. Performance wise, this results in a 2-3 times improvement in writes, but no improvement in reads.

This means of improving performance is reasonable for small files, but becomes difficult for larger ones due to the problem of finding large, unassigned contiguous disk areas.

4. As stated before, the -20 tests were done using 6-bit ASCII character representation. If the user requires 7-bit representation (e.g. he wants files with upper and lower case letters), a significant performance degradation is encountered, with larger record sizes (256 characters and up) suffering the most. This is an example of the many performance uncertainties caused by COBOL's proliferation of data types, and applies to all systems, not just -20's (though they appear to be more sensitive). With as many two dozen formats of data representation (e.g. zoned decimal, packed decimal, binary, signed and unsigned, left and right overpunch, ASCII, EBCDIC, etc.), performance estimates can be very misleading if made on a different data type.

Gordon

+-----+
! d i g i t a l ! i n t e r o f f i c e m e m o r a n d u m
+-----+

Subject: VAX-11 and 11's (and 10's) vs. 360/370

To: Marketing Committee, OOD, Date: 22 MAY 78
Product Line Managers, From: Gordon Bell
Dick Berube, Peter Connell, Dept: OOD
Al Mullin Loc.: ML12-1 Ext.: 2236

CC: Sam Fuller, Bill Strecker follow up 6/5/78

I was under fire by a government customer as to why they should buy DEC when the future gets a 360/370 processor-on-a-chip (e.g., National's latest announcement).

My answer was:

1. The 370 software was optimized for large memory configurations (although there are small configuration systems) -- this gives us a cost advantage over IBM because of memory size and performance due to movement from disk.
2. IBM has commitments to many other architectures (e.g., Series 1, System 3-15, System 34, 3290, etc.).
3. Our encoding in VAX is better oriented to higher level than the 370. (Sam) Bill, can we start to compile data to prove this? (This gets us cheaper systems by the encoding amount.)
4. We have compatibility with the 11 for bounded applications, e.g., terminals. (They don't have this in their machines.)

They ask why we made a VAX vs. 10/20 and what to buy. I said:

1. The 10 software was implemented at high end.
2. The 10 had to be extended, it was out of address-space too.
3. We wanted more compatibility with the 11 than an emulate mode would have given. There's incompatibility (11-10) on: files and data-types that would have made incompatibility.
4. Buy 11's if bounded; VAX if software fits and compatibility with 11's needed; 10's if current base and software exists.

GB:ljp

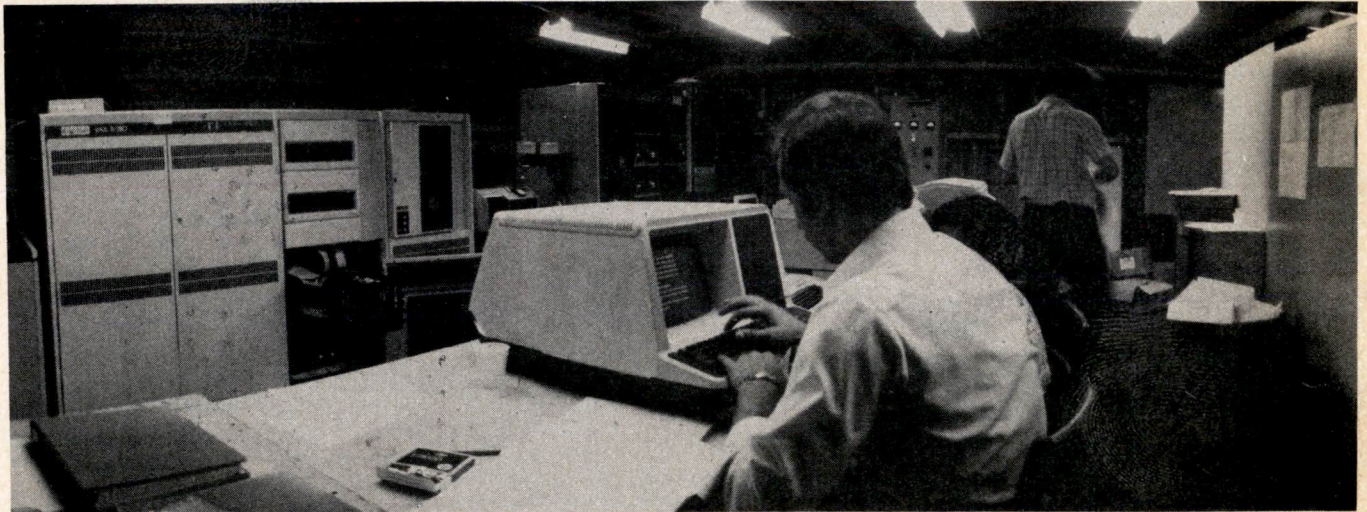
D I G I T A L INTEROFFICE MEMORANDUM

DIST:	Dick Berube	PK3-2/M18	Peter Connell	PK3-2/M18
	Sam Fuller	TW/A08	Al Mullin	PK3-2/F40
	Bill Strecker	ML3-2/E41		
	Jim Bell	ML3-2/E41	Roger Cady	MK1/E25
	Bill Chalmers	MR2-2/M67	Dick Clayton	ML12-2/E71
	Jim Cudmore	ML1-5/E30	Bill Demmer	TW/D19
	Bruno Durr	PK3-2/S56	Ulf Fagerquist	MR1-2/E78
	John Fisher	PK3-2/A93	Jack Gilmore	MK
	Win Hindle	ML5-2/A53	John Holman	PK3-1/P84
	Irwin Jacobs	MK	Ted Johnson	PK3-2/A55
	John Kevill	ML1-3/E58	Dave Knoll	ML1-4/P69
	Andy Knowles	MR2-2/A52	Ed Kramer	MR2-4/M16
	Bob Lander	PK3-2/F33	Bob Lane	HD
	John Leng	MR1-1/F35	Bill Long	PK3-1/A60
	Julius Marcus	MK2/C37	John Meyer	ML12-1/A11
	Gerry Moore	PK3-2/A55	Stan Olsen	MK
	JC Peterschmitt	GE	Larry Portner	ML12-3/A62
	Bob Puffer	ML12-2/E38	Jack Shields	PK3-2/A58
	Jack Smith	ML1-4/F31	Charlie Spector	ML5-2/M40
	Bill Thompson	ML12-1/F41	Gerry Witmore	ML5-2/M40

Giving the PDP-11 a bigger virtual memory required a whole new design,
not just bigger words.

1202 11 0340216 26849V078
DIGITAL EQUIPMENT CORP
C G BELL VP ENG
ML12/A51
146 MAIN ST
MAYNARD MA 01754

THE VAX-11, DEC'S 32-BIT VERSION OF THE PDP-11



Digital Equipment has long had two product lines, the minis and larger mainframes like the DECSYSTEM-10. The new VAX line will be an upgrowth of the 16-bit machines, but may also go a long way toward displacing the bigger

computers. Overall acceptance of the VAX is unpredictable, but several hundred are already in the field (and several, including the one pictured, have been kept back at the barn for in-house use).

by Dileep P. Bhandarkar and
Steve Rothman

Minicomputers emerged in the 1960s as cost effective machines that offered limited functionality at significantly lower cost than for larger computers. Since then, advances in technology have contributed to the increasing functionality of minis. Instruction sets have been enhanced to include operations on floating-point numbers for scientific processing and on character strings for commercial applications. The availability of low cost semiconductor memory has allowed large memories to be attached to small machines at reasonable cost. In fact, semiconductor technology has been the driving force for improvements in

minicomputer systems; while technology does not provide new ideas, it does present the basis for their cost effective implementation.

Further, the trend toward distributed and decentralized processing has created a market for large scale functionality at minicomputer costs. To meet that demand, minicomputer operating systems have evolved from simple monitors to powerful executives capable of supporting multiple environments simultaneously. Often a single minicomputer must perform a mixture of batch, real-time, interactive, and program development tasks using multiple languages.

This increased function, in terms of computing speeds and larger memories and operating system capabilities, has further expanded the scope of applications for minicomputers. Today, distrib-

uted minis are even viewed as viable alternatives to large centralized mainframes. For many of these new applications, a minicomputer's 64KB or 128KB virtual address space is not and will not be a severe limitation. For others, especially where the minis are doing jobs originally assigned to maxis, the small machines require the ability to store and execute large programs without resorting to overlaying techniques.

Handling these larger virtual memories requires changing existing minicomputer designs, coming up with new architectures. In Digital Equipment Corp.'s products, this has led to the VAX-11 architecture, a Virtual Address eXtension of the PDP-11 family.

Virtual addresses are the addresses a program generates to fetch instructions and data. The maximum range of these

DEC claims that typical PDP-11 files and programs may be moved to the VAX in binary form and executed directly.

addresses is called the virtual address space. On 16-bit minis, this space is usually 64KB (64KB being the largest address representable in 16 bits) or 128KB (if separate address spaces are used for instructions and data as on the PDP-11/70). The addresses that the cpu uses to actually address the memory are called the physical addresses and the total amount of memory that can be put on the system is called

the physical address space, which is often several million bytes.

The original PDP-11/20, introduced in 1970, was limited to a physical address space of 56KB. The PDP-11/70, introduced in 1975, extended that to almost 4MB. However, the virtual address spaces of these machines are 64KB and 128KB respectively. Even with large physical memory sizes, these computers are used to run

many small jobs concurrently instead of fewer large jobs. The primary goal of the VAX-11 architecture, on the other hand, was to significantly increase the size of the virtual address space—again, a Virtual Address eXtension.

Why not just give the PDP-11 a larger word size? Well, the size of the virtual address is deeply ingrained in the machine language of a computer. For example, when computing the location of an element within an array, the program must know how large the addresses are so that it can use the appropriate instructions to fetch and operate on them. Therefore, when changing the size of virtual addresses, all assembly language programs must be modified. All higher level language programs, such as those written in FORTRAN or COBOL, must be recompiled with a compiler that generates addresses and code for the new size. Since it is not possible to extend the virtual address size and allow old programs to take advantage of the extension without changing them, the VAX-11 architecture became a new architecture.

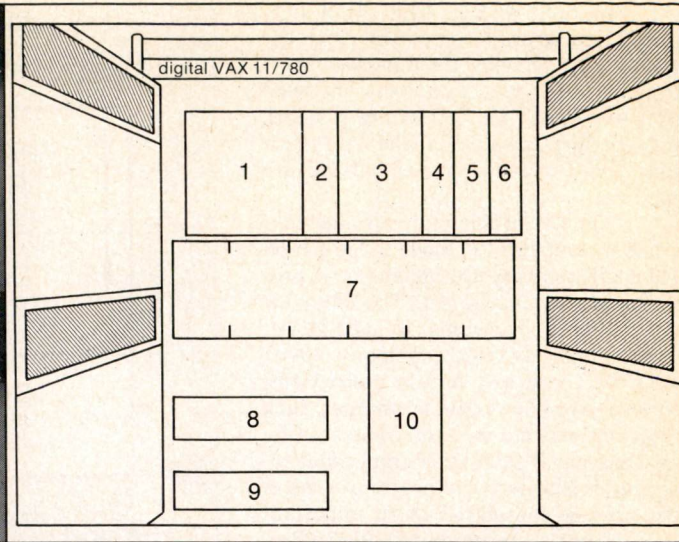
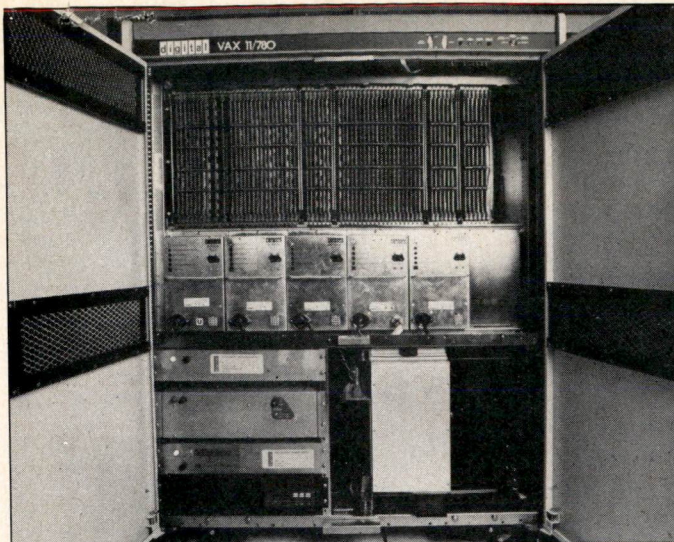
DESIGN GOALS The designers of the new architecture spent several months defining the goals. The first, and by far the most important, was that the architecture should have a long lifetime. This is necessary because of the huge investments that the vendor makes in designing hardware and software for the architecture, and the equally large commitment made by customers with the software they write. The desire for a long lifetime caused us to try to understand the general directions of computing for the next 10 years, specifically the developments that might be expected in cpu hardware technology and software technology.

Our assumptions about hardware were that the cost of both logic and memory would continue declining, so that we could design a much more complex architecture and still build machines with minicomputer prices. Our primary assumption about software was that higher level languages would be used to try to control the constantly increasing percentage of computer expenditures that is spent on software. Thus, we wanted to design an architecture for which good optimizing compilers could be written, and which would include some of the primitives common to higher level languages in the hardware itself. We also left room for the architecture to grow, because we realized we could not predict everything needed.

Our second major goal for the architecture was to base it on the PDP-11. By making it similar, large numbers of users

Hexadecimal Format	Explanation	Assembler Notation
0 5	opcode for RSB	RSB Return from subroutine
D 4	opcode for CLRL	CLRL R9
5 9	register R9	Clear register R9
B 0	opcode for MOVW	MOVW 356(R4), 25(R11)
C 4	word displacement mode, register R4	Move a word from address which is 356 plus contents of R4 to address which is 25 plus contents of R11
6 4	356 in hexadecimal	
0 1		
A B	byte displacement mode, register R11	
1 9	25 in hexadecimal	
C 1	opcode for ADDL3	ADDL3 #5, RO, @ #A[R2]
0 5	short literal 5	Add 5 to a 32-bit integer in RO and store the result in address which is sum of A and 4 times the contents of R2
5 0	register mode R0	
4 2	index prefix R2	
9 F	autoincrement mode, program counter relative	
address of A		

Fig. 1. VAX-11 instructions may be from 1 byte to 37 bytes long. Each consists of an opcode and up to six operand specifiers (which may be up to 9 bytes long each). The opcode dictates the operation and data format; the specifiers tell where the data is.



Components of this VAX-11/780 mainframe include: (1) the cpu, (2) Unibus adaptor, (3) 4MB of MOS memory, (4) and (5) Massbus adaptors, (6) slots for options, (7)

modular power supplies, (8) battery backup power supply, (9) LSI-11 microcomputer, and (10) the floppy disk.

would not have to learn everything from scratch, but could more easily move to the new architecture. Thus the initial cost of starting up would be significantly reduced.

We defined our last major goal as the desire to make the architecture appropriate over wide ranges of system cost, performance, and applications. This would allow a wide range of user needs to be served by a single architecture and eliminate unnecessary spending on our part associated with the support of many different architectures. Our goal was to create a family of machines with the same architecture; within the family, performance would be traded off for price, with function a constant.

In extending the virtual addresses, our major decision was whether to stop at 24 bits, allowing for addressing 16 million bytes, or go to 32 bits and 4 billion bytes. Since the *main* reason for going to new architectures in the past, as with VAX-11, has been the need for larger virtual addresses, it was felt that 24 bits would be extremely short-sighted. Therefore, we chose to go with 32 bits, which should last through the 1980s. (Note that even mainframes such as the IBM 370 presently provide only 16MB of virtual address space.)

The goal of easy use by higher level languages was broken down into several specific requirements:

1. Orthogonality of operation, data type, and address mode. This means that the operation being performed (ADD, SUB, etc.), the type of data (integer, floating-point, etc.) and method of addressing (indirect, indexed, etc.) can be considered independently by the compiler, which makes compilers faster and more efficient. We believe that the level of orthogonality achieved is unique to the VAX-11.

2. No forced alignment even on longword (32-bits) boundaries. This means that data items larger than a byte

can still be on any byte boundary, as is required in some languages (such as the FORTRAN COMMON facility). The architecture supports full byte addressability.

3. Variable number of operands.

By picking an instruction format which allows each instruction to have its "natural" number of operands, it is possible to have instructions in the format desirable for compilers ($A + B - C$ is ADDL3 A, B, C while $A + B - B$ is ADDL2 A, B). The PDP-11, on the other hand, had only two-address instructions. Thus $A + B - C$ takes two instructions on that machine.

4. Good primitives. Many operations commonly found in higher level languages are built directly into hardware, including three-address arithmetic, loop control (the FORTRAN DO and BASIC FOR loops are one instruction), and output formatting (the EDIT instruction can be used for COBOL PICTURE statements).

In addition, instructions were added to make operating systems more efficient. Examples of these are hardware support

of queues, access to variable length bit fields, and simple instructions to save and restore program context. (Such instructions are not available on PDP-11s.)

COMPATIBILITY The goal of compatibility with the PDP-11 was approached in several

ways:

1. The formats for representing data are the same.

2. The media formats for I/O are the same—even the same size, 16 bits. Combined with the same data formats, this means all data files can be read by both PDP-11 and VAX-11 systems.

3. The assembly language syntax and mnemonics are basically the same. Not only will assembly language users find the new architecture very similar to the old at this level, but this cultural compatibility also allows language compilers that generate efficient PDP-11 code to be readily adapted to generate efficient VAX-11 code.

System Price Comparisons

PDP-11/70 vs. VAX-11/780

	Small System		Large System	
	PDP-11/70	VAX-11/780	PDP-11/70	VAX-11/780
Memory	256KB Core	256KB MOS	1MB Core	1MB MOS
Console Terminal	1	1 +LSI-11 & Floppy Disk	1	1 +LSI-11 & Floppy Disk
Mass Storage				
Disk	28MB Dual RK06	56MB Dual RK07	176MB RP06	176MB RP06
Tape			1600bpi 45ips	1600bpi 45ips
Communication lines			8	8
Line printer			600 lpm	600 lpm
Software	RSX-11M	VAX/VMS	RSX-11M FORTRAN COBOL	VAX/VMS FORTRAN COBOL
Price	\$92,000	\$128,000	\$228,000	\$240,000

In fact, VAX-11 high level languages are upward compatible with existing PDP-11 versions. Even the VAX operating system, VAX/VMS, was modeled after the PDP-11 RSX operating systems, and key functions, such as the file system and record management facilities, are identical.

The VAX architecture also includes a PDP-11 compatibility mode in hardware, which allows many user mode PDP-11 programs to run unchanged. The emulator can support programs in FORTRAN, COBOL, BASIC-PLUS-2, and many PDP-11 utilities. There are certain restrictions, however; for one reason or another, such programs may not use sense switches (the VAX has none), PDP-11 floating-point instructions (the performance improvement available is considered more important than the cost of recompiling), PLAS (Program Logical Address Space) directives, or DECnet data communications facilities, and may not rely on PDP-11 mapping or PDP-11 memory arrangement.

Migration from PDP to VAX is aided by the use of an Applications Migration Executive, which is a subroutine package for emulating RSX-11M services. RSX-11M service requests are translated to VAX/VMS format and the status results translated back.

Together, these features allow typical RSX-11M user programs and files to be moved to VAX in binary form and executed directly there.

INSTRUCTIONS AND MEMORY

To understand how and why all this happens, it is necessary to look into VAX-11's instructions and memory management.

The instructions consist of a variable number of bytes, from 1 to 37. Each instruction consists of an opcode, which may or may not be followed by 1 to 6 operand specifiers. (See Fig. 1.) The opcode dictates the operation and the data format, the specifiers tell where the data is. Opcodes are 1 byte, with provisions for going to 2 bytes; operand specifiers are from 1 to 9 bytes. (For comparison, PDP-11 instructions have up to two operand specifiers, and instruction lengths of 1, 2, or 3 words 1-1 of 16 bits.)

Operand specifiers exist in several formats. There are formats that provide for literals (immediate data), registers, direct (absolute) addressing, indirect addressing, relative addressing, stacks, plus others allowing for indexing in conjunction with most of the other formats.

The VAX-11 architecture supports arithmetic in six different formats: 8-bit, 16-bit, and 32-bit integers, plus 32-bit and 64-bit floating-point numbers, and 31-digit packed decimal. The architecture also provides instructions for working with bit fields between 0 and 32 bits, and character strings up to 64KB long. (Neither 32-bit integers, decimal, bit fields, nor character string operations was provided on the PDP-11.)

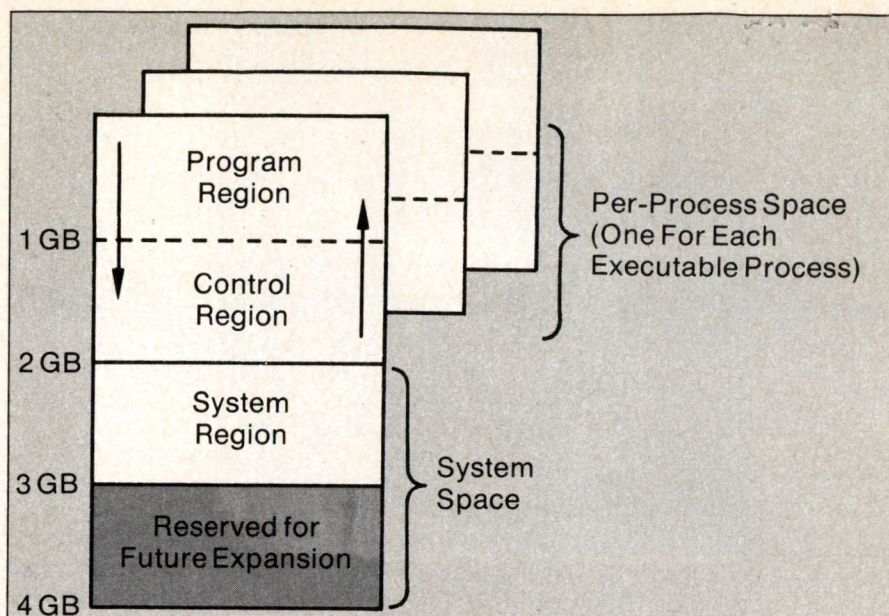


Fig. 2. The VAX-11's virtual address space allows each user to command 1 billion bytes of program space and another billion bytes of "control region," which is mostly data space.

Operations are divided into several groups; integer and floating-point, arithmetic, logical, special, bit field, queue, address, control, character string, decimal string, and edit. And there are many of the common optimizations to reduce program size, for example:

CLear A acts like MOVE #0, A
TeST A acts like CoMPare A, #0
INCrement A acts like ADD #1, A

Several instructions provide functionality hitherto not available on many contemporary architectures. A few examples are:

- EMOD: Does extra precise range reduction of floating-point numbers, for example to normalize the argument of a trigonometric function to 2π .
- POLY: Computes the value of a polynomial of the form $a_n x^n + a_{n-1} x^{n-1} \dots a_1 x + a_0$, which is useful for efficient computing of math library functions.
- INDEX: Checks the value of an index and computes the location of an element within an array.
- CASE: The equivalent of a FORTRAN computed GO TO including range checks.
- CRC: Is used to calculate a cyclic redundancy check code. These codes are commonly used as an error detection mechanism for long streams of data, such as in data communications.

Fewer instructions are required to perform a given function on the VAX-11 than on the PDP-11. As an example consider the high level language construct $A[I] = B[I] + C[I]$. The following illustrates how this construct is handled on the VAX-11 and on

the PDP-11.

VAX-11

```
MOVL I, RO
ADDL3 B[RO], C[RO],
      A[RO],
```

PDP-11

```
MOV I, RO
ASL RO
ASL RO
MOV C(RO),
  A(RO)
MOV C+2(RO),
  A+2(RO)
ADD B(RO), A(RO)
ADC A+2(RO)
ADD B+2(RO),
  A+2(RO)
```

Some of the major features of the memory management and protection mechanism are:

1. Virtual memory management built into the hardware and VMS operating system. This allows multiple user programs to be supported simultaneously, each with a virtual address space exceeding the total physical memory available.
2. Automatic sharing of half the virtual address space between all programs.
3. Four hierarchical protection levels (kernel, executive, supervisor, user) instead of the PDP-11/70's three.

There are many more features of the architecture, especially relating to the operating system environment, that have not been described. Protection and sharing are provided at the page level. Memory is divided into pages, each consisting of 512 bytes. This page size, although relatively small, yields better memory utilization than that available with the page size of 4KB in the PDP-11. (Availability of low cost semiconductor components allows the cost-effective implementation of the more complex memory management hardware required.) Page tables are used to map virtual addresses to physical mem-

ory. They also specify the type of accessibility—(no access, read-only, read/write) for each of the four protection modes. (See Figs. 2 and 3.)

THE VAX-11/780

The VAX-11/780 computer system, the first to use the new architecture, consists of the central processing unit, main memory subsystem, I/O subsystem, and console subsystem. (See Fig. 4.) The cpu, memory and I/O subsystems are connected through a high speed bus called the Synchronous Backplane Interconnect (SBI), which is the primary control and data transfer path. Currently, up to two memory controllers with a total of 8MB of memory can be connected to it, but the SBI is capable of addressing up to 512MB. The UNIBUS adaptor (UBA) provides a standard 1.5MB/sec UNIBUS for the attachment of a wide variety of peripherals. Up to four MASSBUS adaptors (MBA) are used for disk and tape block transfers to 2MB/sec. Thus, the VAX-11/780 uses existing PDP-11 peripherals.

The SBI has a 32-bit data path and operates at a cycle time of 200nsec. Its protocol permits a 32-bit transfer using one address cycle and one data transfer cycle, or a 64-bit transfer using one address cycle and two data transfer cycles. The maximum data rate achievable is 13.3MB/sec. To enhance its reliability, substantial protocol checking occurs on every cycle. In addition, maintainability is aided by the recording of the history of

the last 16 SBI cycles by the cpu.

As might be expected, the cpu itself uses microprogramming to implement the instruction set, including floating-point, character string, decimal string, and compatibility mode instructions. A 12KB writable diagnostic control store (WDCS) is standard and can be loaded from the console subsystem. Further, floating-point performance can be enhanced with the optional Floating-Point Accelerator (FPA). A 12KB user writable control store is also available.

Internal data paths are 32-bits wide and standard Schottky-TTL logic circuits are used. Emitter coupled logic circuits and custom LSI have been used in selected places to optimize system performance and reliability.

The cpu employs cache buffers extensively to achieve high performance. For example, an 8KB cache provides fast access to frequently used data and reduces traffic on the SBI. A 128-entry address translation buffer acts as a cache for recent virtual to physical address translations. An 8-byte instruction buffer enables the cpu to fetch and decode the next instruction while the current one completes execution. The cpu-SBI interface includes a write buffer. Thus, when the cpu performs a write to memory, it need not wait for memory to accept the data. It writes into the buffer, initiates a memory transfer, and continues execution.

The main memory subsystem consists of error-correcting MOS memory using Hamming codes. Single-bit errors are

COBOL Benchmark Performance (with decimal subscripts)

VAX-11/780	97
IBM 370/135	59
IBM 370/145	133
IBM 370/158	597
IBM 370/168	1079

On benchmark programs supplied by U.S. Steel, VAX-11/780 performance on COBOL jobs using decimal subscripts falls somewhere between that of the IBM 370/135 and 370/145. (Such benchmarks must have been running for years, as the numbers express relative productivity compared to an IBM 1460.)

corrected automatically to improve the mean time between failure by a factor of 10 over memory systems without error correction. All 2-bit errors, and most involving more bits are detected. All corrected and uncorrected errors are reported to the operating system for appropriate logging and corrective action.

The VAX-11/780 is capable of operating despite faulty memory locations. The memory management scheme allows the operating system to map around failed memory locations, thus significantly improving system availability. Two memory controllers can be connected to a VAX-11/780 system, each handling up to 4MB. Interleaving is possible if the two controllers manage equal amounts of memory.

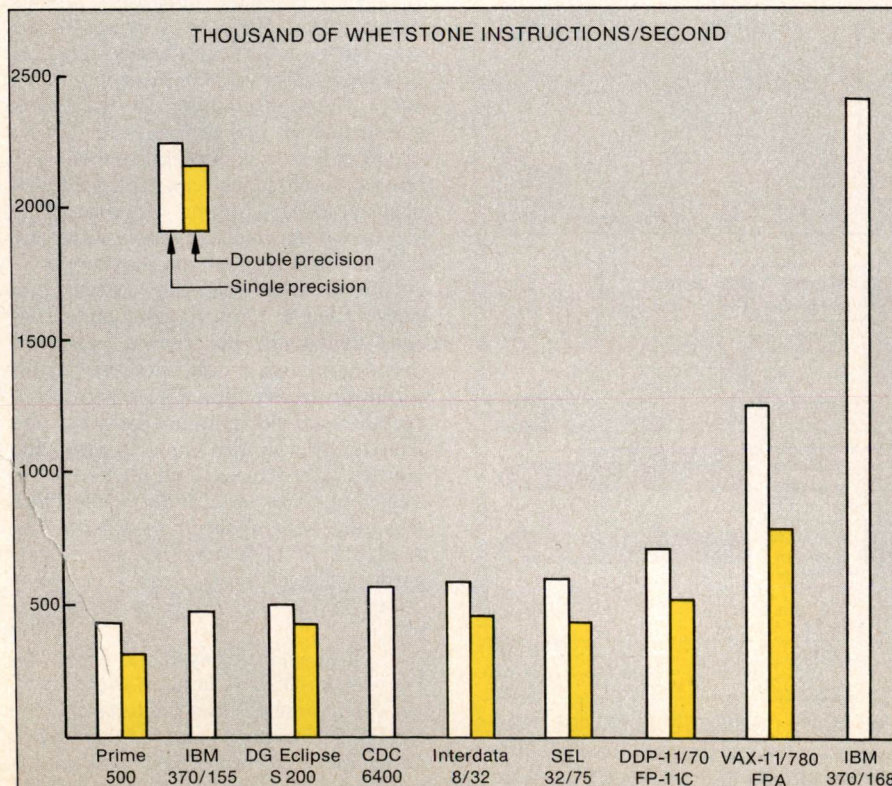
Memory is organized in 64-bit quadwords using 4Kbit or 16Kbit MOS RAM's. Memory controllers have buffers that hold up to four memory requests. And optional battery backup is available to protect memory contents over short-term power failures.

The console subsystem consists of an LSI-11 microcomputer with 16KB of RAM, 8KB of ROM, a floppy disk, a console terminal interface, and a port for remote diagnostics. The floppy is used both as a load device for system installation as well as a medium for the distribution of software updates. We believe this intelligent console is unique among minicomputer systems.

THE OPERATING SYSTEM

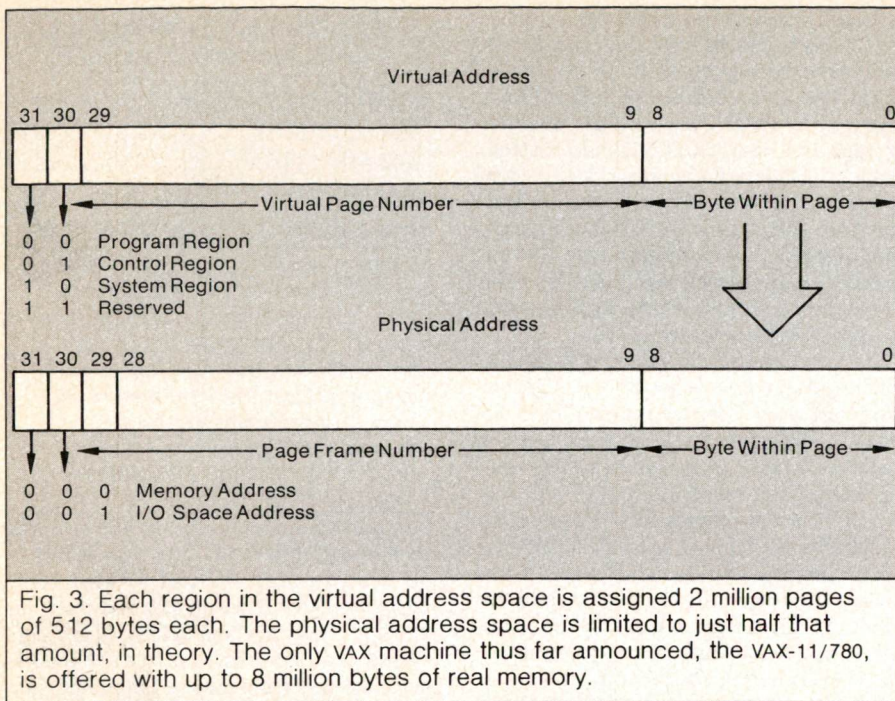
VAX/VMS is a multiuser, multifunction virtual memory operating system that supports multiple languages, an interactive interface, and program development tools. The operating system is designed for many applications, including scientific/time-critical, computational, time-sharing, data processing, transaction processing, and batch. The system performs demand paging, and under VAX/VMS a process pages only against itself—thus individual processes cannot significantly degrade the performance of other processes.

The memory management facilities provided by the operating system can be controlled by the user. Any program



DEC claims powerful performance for the VAX-11/780 on FORTRAN jobs. The results above are for a "Whetstone" job mix including single- and double-precision arithmetic, and the DEC processors tested both have optional hardware for floating-point instructions.

Memory is divided into small but manageable pages of 512 bytes.



can prevent pages from being paged out of its working set. With sufficient privilege, it also can prevent the entire working set from being swapped out, to optimize program performance for time-critical or interactive environments. Sharing and protection are provided for individual 512-byte pages. As mentioned, four hierarchical access modes (kernel, executive, su-

ervisor, and user) provide page protection.

VAX/VMS schedules CPU time and memory residency on a preemptive priority basis. Thus, time-critical processes do not have to compete with lower priority processes for scheduling services. Scheduling rotates among processes of the same priority. The scheduler adjusts the priori-

ties of processes assigned one of the low 16 priorities to overlap I/O and computation. Time-critical processes can be placed in one of the top 16 scheduling priorities, in which case the scheduler does not alter their priority. Their priorities can be altered only by an appropriately privileged user.

Interprocess communication is provided through shared files and shared address space, event flags, and mailboxes, which are record-oriented virtual devices.

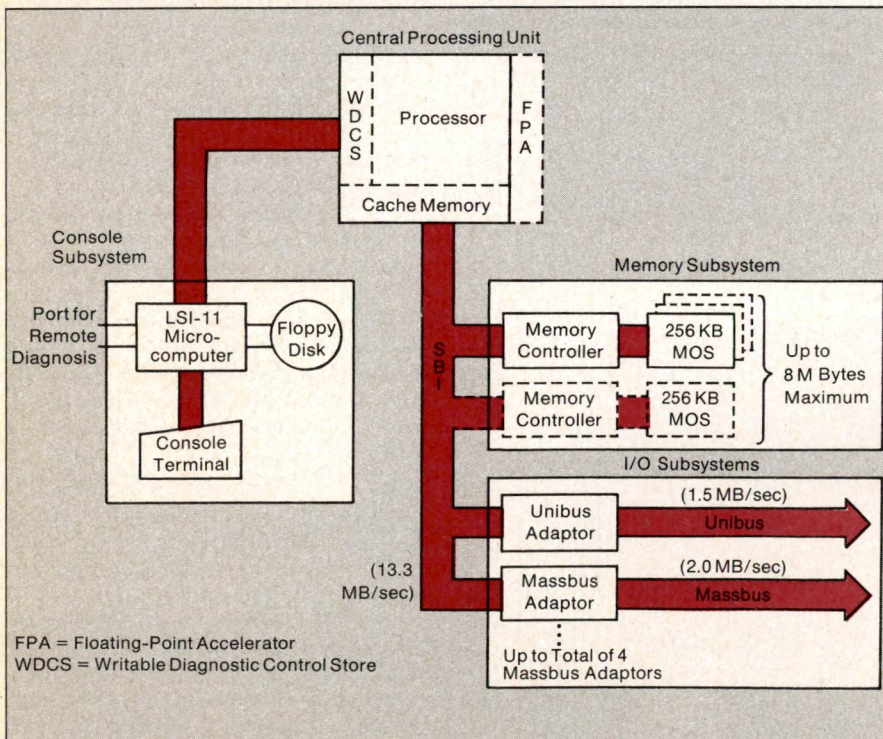
The VAX/VMS command language is suitable for both the interactive and batch environments. Batch facilities under VAX/VMS include job control, multistream spooled input and output, operator control, conditional command branching and accounting.

Command procedures are supported by the command languages as a method of executing frequently used sequences of commands, or creating new commands from the existing command set. Command procedures accept parameters and can include extensive control flow.

VAX/VMS provides a program development capability that includes editors, language processors, and a symbolic debugger. Its FORTRAN IV-PLUS and VAX-11 MACRO language processors produce native code. The PDP-11 COBOL-74/VAX and PDP-11 BASIC-PLUS-2/VAX language processors produce compatibility mode code. Although compatibility mode and native mode cannot be freely mingled in a single program, compatibility mode jobs and native mode jobs can be run simultaneously, sharing system resources. Also, the two programs can share data and communicate through mailboxes.

VAX/VMS operating system provides a file and a record management facility that allows the user to create, access, and maintain data files and records within protected files. The record management services handle sequential and relative file organizations, sequential and random record access, and fixed- and variable-length records. Indexed files with sequential and random record access are available to compatibility mode programs, such as those written in PDP-11 COBOL-74/VAX or PDP-11 BASIC-PLUS-2/VAX.

The operating system supports the Files-11 On-Disk Structure Level 2 (ODS-2), which provides the facilities for file creation, extension, and deletion with owner-specified protections and multi-level directories. On-Disk Structure Level 2 is an upward extension of Level 1, the file system currently available under the TRAX, IAS, and RSX-11 PDP-11 operating systems. Both native and compatibility



mode programs can access both Level 1 and Level 2 volume structures.

DECnet/VAX networking capabilities are optionally available for point-to-point interprocess communication, file access, and file transfer, and include down-line command file and RSX-11S system image loading; but these all require more explanation than can be provided here.

FUTURES

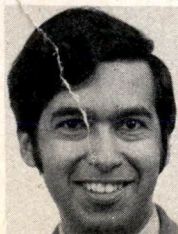
Whether the goals for broad product ranges have been met in the VAX design will be determined by time. Certainly the PDP-11 has demonstrated the type of range that is desirable. Eight years after their introduction, PDP-11's span almost three orders of magnitude in price and almost two orders of magnitude in performance, and are used in nearly every conceivable application. Equally certainly, future advances in LSI technology will allow the capabilities of large scale machines, which are already translatable to minicomputers as evidenced by the VAX-11, to be implemented at microcomputer scale—perhaps by the mid-'80s. *

STEVE ROTHMAN



Mr. Rothman is an engineering manager in the VAX-11 engineering group at Digital Equipment Corp., and was a member of the original VAX architecture development team. Since joining Digital in 1969 he has worked on several of the PDP-11 central processors.

D.P. BHANDARKAR



Dr. Bhandarkar is a principal engineer in the VAX-11/PDP-11 Systems architecture department at Digital Equipment Corp. Earlier, he was a member of technical staff in corporate development at Texas Instruments Inc. He is a holder of U.S. patents on magnetic bubble memory organization, and fault-tolerant monolithic memories.

A useful freebie.

Everybody offers free information.

But not everyone offers *useful* free information.

We do.

We'd like you to know about our Remote Computing Services (RCS). How flexible they are. How efficient, economical and versatile they are. But most of all, how you can benefit from them.

Just fill out the coupon and return it to us. We'll send you our Remote Computing Services brochure. It may be the best 15¢ investment you make this year.

**Martin Marietta
Data Systems**
*We
Build & Run
Systems*

**Remote
Computing
Services**
Brochure

Marketing Services
Martin Marietta Data Systems
300 East Joppa Road
Baltimore, Maryland 21204
Phone: 301-321-5744

Name _____

Co. _____

Title _____

Address _____

Phone _____

City _____

State _____ Zip _____

Now using RCS? Yes No

CIRCLE 38 ON READER CARD

**Symbolic Debugging for COBOL
& ASSEMBLER Applications Now Available!**

Down in the dumps?

INTEREST eliminates the need to wade through those complex CICS dumps. INTEREST automatically detects errors, tells you where the bugs are, keeps your CICS system from crashing, and lets you solve your problems on-line. It is used in CICS installations worldwide.

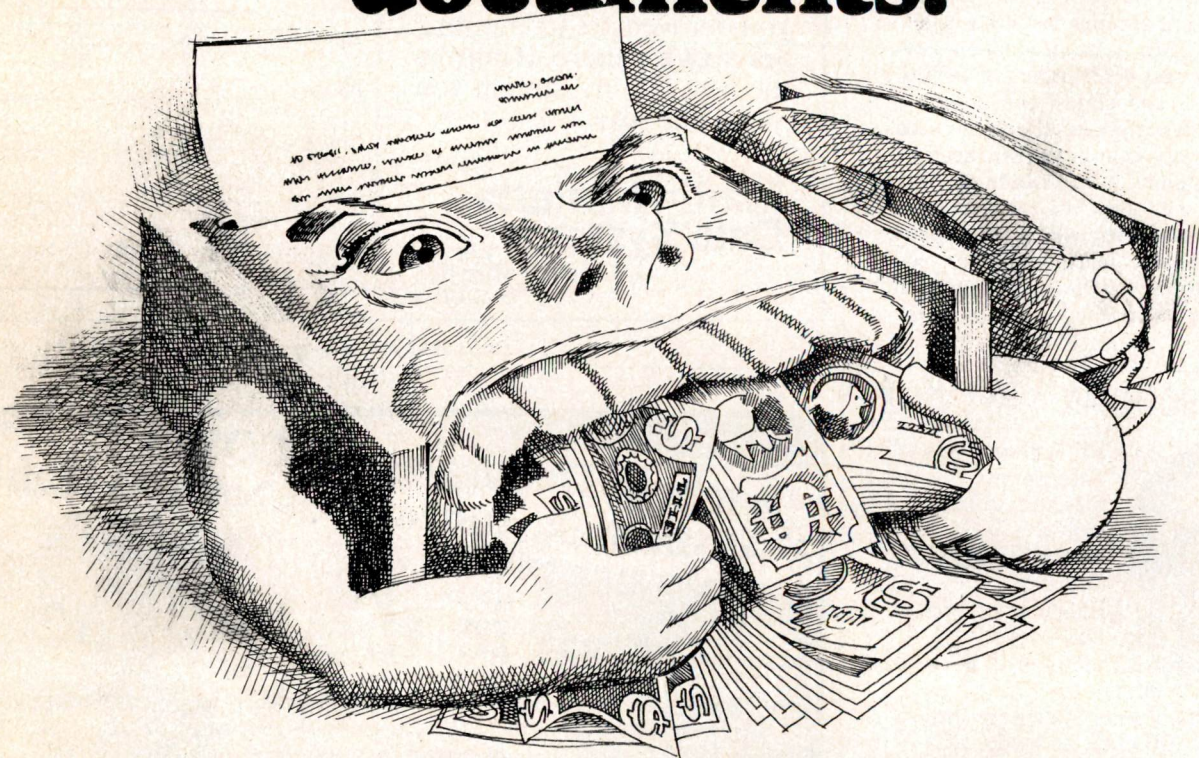
If you have CICS or CICS/VS, On-Line Software, the industry leader in teleprocessing software and services, provides a variety of solutions to your problems. Write to us, or, for immediate action on INTEREST, give us a call.

**ON
LINE SOFTWARE
INTERNATIONAL**

65 ROUTE 4 EAST — RIVER EDGE, N.J. 07661
(201) 488-7770 — (800) 526-0272 — Telex 642171

CIRCLE 139 ON READER CARD

There's only one problem with slow-speed facsimile machines. They go through money faster than they go through documents.



If you have a slow-speed facsimile machine, you've probably noticed that it consumes a lot of time.

And eats up a lot of money.

That's because the more time your machine spends on the phone, the more money you have to spend on phone bills.

In fact, if you transmit as little as 10 documents a day on some slow-speed machines, you can actually save money by getting a high-speed machine.

On a high-speed Rapifax 100, you can transmit the same document that eats up four to six minutes of expensive telephone line time in just 35 seconds.

And you'll get a better looking copy at the other end.

The Rapifax 100 is as easy to operate as an office copier.

It can faithfully reproduce six-point type

or detailed drawings. And it's completely automated.

But its most important feature is one that no other high-speed machine has.

An established track record.

Rapifax 100 was the first commercial high-speed machine for use over ordinary telephone lines.

It has more than four years' experience behind it.

And it's been used successfully by thousands of businesses.

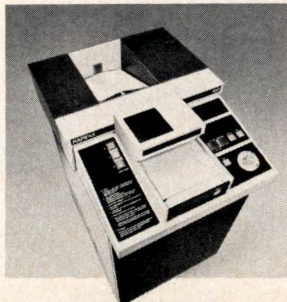
If you're fed up with the amount of time and money your slow-speed facsimile machine consumes, get the high-speed machine with a proven record of performance.

The Rapifax 100.

For more information about it, call our toll-free number: 800/631-1155.*

Or write to Rapicom, Inc., Seven Kingsbridge Road, Fairfield, N.J. 07006.

*Except from Hawaii or Alaska.



RAPICOM

JOINING RAPIFAX CORPORATION AND DACOM, INC.

to whiter

Copy Al. Mullin + return. Fyi gB ✓

1184
C. Miro
32 bit

SUBJ: WHITE PAPER NEEDED ON 32 BITS, USER-MICROPROGRAMMING,
AND US

+ T. Johnson

To: Distribution

From: Gordon Bell
Date: August 16, 1974

We keep getting asked about these issues; questions won't diminish. Let's understand what we're going to say. The problems seem to be: 16 vs 32; data-types; concern over implementation; larger programs; user microcode; and education/pr.

The issues seem to be:

1. 16 vs 32 (or possibly 24). We were wrong before in not coming around faster to the issue of word length when the 12 bit machine was competing with the 16 bitters. I don't think 32-bits is the same fundamental issue. In the previous case, IBM had turned on to 8-bit characters, and people didn't think they could pack characters efficiently in 2 12-bit words. Also, whereas the 12-bit machine is really optimal for 4K memories, memory sizes are now larger, and users thought they were losing some efficiency on greater than 4K systems.

2. What do you mean a 32-bit machine? PDP-11 is not word length sensitive in the same way that older machines were, because it is really variable length in both data types and instructions.

Instructions are 1, 2, or 3 16-bit words long, with the use case being 1.6 to 2.0 words/instruction.

As for data types, we provide 8-bit bytes (used as characters, bit-vectors, and integers), 16-bit words (used as bit vectors and integers), 32 bit floating point (with 25-bit integers mantissa and an 8-bit exponent), and 64-bit floating point for high precision operations. In some of the applications, I've seen the 11 misapplied by using programmed double precision arithmetic, where the customer needed 24 bits of precision, instead of using the floating point. The reason presumably is the cost of the floating point, but our miseducation of him loses in these ways: it's slower; it takes more instructions to handle the scaling; and it takes longer to program because of the care needed in the scaling of numbers; also with improper care, accuracy can be lost easily.

Another significant data-type is addresses. A few of the 32-bit machines go beyond the standard 16-bit address, as such, data-operation has to be permitted on greater than 16-bit integers.

3. Implementation versus the user Instruction-set. A user should not care how we implement a machine, unless it affects his use in some way. As long as a machine has the proper facilities, it could be implemented in any word length (including 1-bit serial), and the buyer is free to pay his money and take a choice of performance. In essence, the sophisticated user only worries about whether a machine has the right data-types for his job; the speed a given implementation interprets (operates on) these data-types (or higher level language); and how a program fits into the facilities we provide.

↑
perhaps of
I send you
the
stuff when
you
send this
stuff
at night
later.

In essence, users should have stopped worrying about implementation long ago.

4. Real issues. The issues of a 32-bit or larger implementation seem to be:

Cost	Generally larger, but if Error correction must be in memory, it's a cheaper memory.
Band width for I/O, etc.	2X
Band width for 32-bit integers, floating point	2X
Performance for instruction-set	Perhaps 1.5

or conversely, for a given performance level a wider word machine might be cheaper.

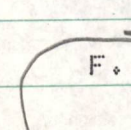
Data-types for an 11 Must solve addressing and data-types for this implementation.

A user may thusly emphasize:

- A. Implementation--32 bit memories (who cares)
- B. Data-types--we are currently only remiss in 32-bit integers, but he should probably be using floating point (maybe important for address arithmetic too).
- C. Address-space (logical address)--some competitors don't do any better than PDF-11, e.g. Modcomp (except they do provide full 65K words). In the future, we may have to adopt a different solution to increase task sizes beyond 65K bytes or 65K + 65K if we utilize I and D space. People want to run bigger programs, but mostly access large arrays!
- D. Memory Management--they don't understand that large memories doesn't solve this problem, but creates it. Having a large, linear address space while solving a small problem (physical memory size) just creates a bigger problem (how are multiple programs or multiple tasks placed in the memory with protection and sharing?)
- E. Physical memory size--easy to increase. Our processors have been probably overpowered for the small memory size. IBM believes 1 instruction/sec. requires 1 byte of memory. By this token an 11/40 would need about 256K words, and a 45 could take maybe 2-3 times this. A slow KA10 performs at about 11/45 speed and is relatively balanced at a user space of about 400K bytes.
- F. 32-bits as a memory bandwidth solution--it helps a bit. This can be solved by higher speeds and more parallelism in the structure (e.g. multiport memories).

Note for very fast floating point this is unimportant generally.

transaction oriented and recent process-oriented and independent, small processing systems emphasize, independent, small tasks.



5. The 11/55 is a 32-bit machine by any reasonable criteria. The 11/55 has 32-bit data-types, instructions, and most data paths. The fact that it has a 16-bit UNIBUS for some I/O is irrelevant unless it affects performance of some I/O. It shouldn't, because the high performance

11/70

about x2 for an 11/45 a 360!

I/O is handled direct to memory through other MASSBUS controllers.

By increased physical memory, systems can be balanced with proper I/O, and physical memory to absorb the processor capacity. While we may not support I and D space to give the user somewhat larger programs, more programs and/or tasks can be run in parallel because memory can hold them. Our monitors are all multiprogrammed and/or multitasking and have overlays. These address the large program issue in many respects better than competitive monitors.

6. The 11/55⁷⁰ cache as an answer to problem of setting performance automatically without user microprogramming. Also it automatically finds important time consuming parts of a program without user analysis and program movement. This is carried out by automatically migrating user code into the cache (fast) part.

7. User microprogramming and the 11/45 bipolar. We've done an abysmal job of selling the 11/45 bipolar here as the answer to user microprogramming. The microprogrammers fall into 2 camps: experimenters with concept; and people who see it as a fast machine they need. A reasonable sales/promotion strategy can win them both.

Performance increase through microprogramming is predicated on the fact that a small physical part of the program is executed most of the time, hence can be placed in a small memory.

The issues I see:

A. Speed. 11/45 bipolar generally faster than user microcoded machine.

B. Speed for floating point. Both 40 and 45 are faster than general purpose microcode machines.

C. Programming use. User microcode requires different assemblers, compilers, etc. For high performance unencoded microprogramming (i.e. horizontal microprogramming, as it has been erroneously named), it is possible to affect the machine part and really build an impossible system.

D. Program size--small for user microprogram scheme.

E. Analysis of what to microprogram is difficult and maybe counter intuitive. User must always be re-coding program. With the 11/45 the same, PDP-11 instruction-set is used for bipolar and regular memory, hence the decision is not a major one requiring a different program, programming techniques, system software, etc.

F. Applicability to higher level programs. The 11/45 scheme works for FORTRAN, COBOL, etc. programs. Namely, one finds the part of program that needs to be fast, and proceeds to place it in the bipolar part. McBride has pointed out an excellent ad campaign based on the use: COBOL MACRO, and FORTRAN--our Microprogramming Languages.

G. Memory management problem is created if user microprogramming really works.

SUBJ: PC.ISP COMPATIBILITY

PAGE 1
03-31-75
DATE: GORDON BELL
FROM:

* * * * *
PLEASESEND TO: JIM BELL ML3-4/E44
* * * * *

SUBJ: Pc,ISP COMPATIBILITY INVESTMENT ANALYSIS BASED ON
DEC SOFTWARE

To: Peter Christy

CC: Bob Bean, Denny Pavlock, VAXC

In getting a handle on cost to convert (upgrade) to an extended PDP-11, we need to get a better definition of this dimension. This seems best done by assuming one, and using it.

(Would you please attempt to analyze the investment using this definition. Modify, if necessary, interacting with Richy, me, VAXA, etc.)

MODEL

Figure 1 shows the general operational notion of machines consisting of hardware, operating system(s), and languages. At each level, a machine exists which may be well defined by the previous level, or several lower levels (e.g. in 11/D privileged VM).

MACHINES WE SUPPORT AND THE PROBLEM OF POORLY DEFINED INTERFACES

We support a fairly large number of machines ranging from many hardware models to languages. In trying to extend the 11 architecture, we need to provide a higher level interface that supports the notion of a process. However, it is not clear that many of our multiple operating systems have a very well defined interface. Instead operating systems have "holes" to allow direct access to particular I/O and other hardware features. Hence, each combination of machine X operating system provides another machine which constrains compatibility. I believe we must develop a well-defined interface (e.g. 11/D VM) that all languages, utilities, special applications use. In this way, we can modify, evolve, change, etc. any lower level without affecting our investment in a higher level software.

SUBJ: PC.ISP COMPATIBILITY

DATE:
FROM:

PAGE 2
03-31-75
GORDON BELL

It is the task of Richy Lary to design this interface in such a way that software can be transported as we develop new machines and operating systems throughout the next 5 years.

Table 1 attempts to define the various hardware, software, languages machines we support. Note, the relatively large number--say, in contrast to PDP-10 which is 1 software, going on 2. There are 2 hardware machines going on 3. Richy is getting this in a better operational state, as a basis for definition of the next 11.

WHAT WILL THE 11 EXTENDED USER MACHINE BE?

If Table 1 is correct, we probably should provide the ability to create user interfaces like the current 11/VM. We can also, by software, provide multiple 11/20 virtual machines; in contrast, it is probably less useful to provide multiple 11/70 VM's, except for development of monitors. As we extend the address space, we automatically get an 11/VAX user machine interface. To get better performance and be competitive, we must place more operating system capability in hardware. This implies that we cannot support the plethora of interfaces we're used to.

The task of VAXA is to define this hardware interface together with the software interface.

SOFTWARE INVESTMENT AS A KEY DESIGN CRITERIA OF THE INTERFACE

The interface we provide in subsequent machines should be a tradeoff of: permitting as much user-level software to run, versus taking advantage of new capabilities that might be provided (e.g. Recursive Virtual Machines).

For the machines in Table 1, would you:

1. Quickly ascertain that these are the machines we should consider.
2. Enumerate the programs by type, name, quantity, and general quality for each machine. This would include: diagnostics, operating systems, operating system specific handlers (e.g. COMTEX), VM-level (e.g. FORTRAN IV+), and language specific applications (e.g. DIBOL Machine: Accounts Receivable Package).
3. Estimate the code (in %) investment that our users have.

SUBJ: PC.ISP COMPATIBILITY

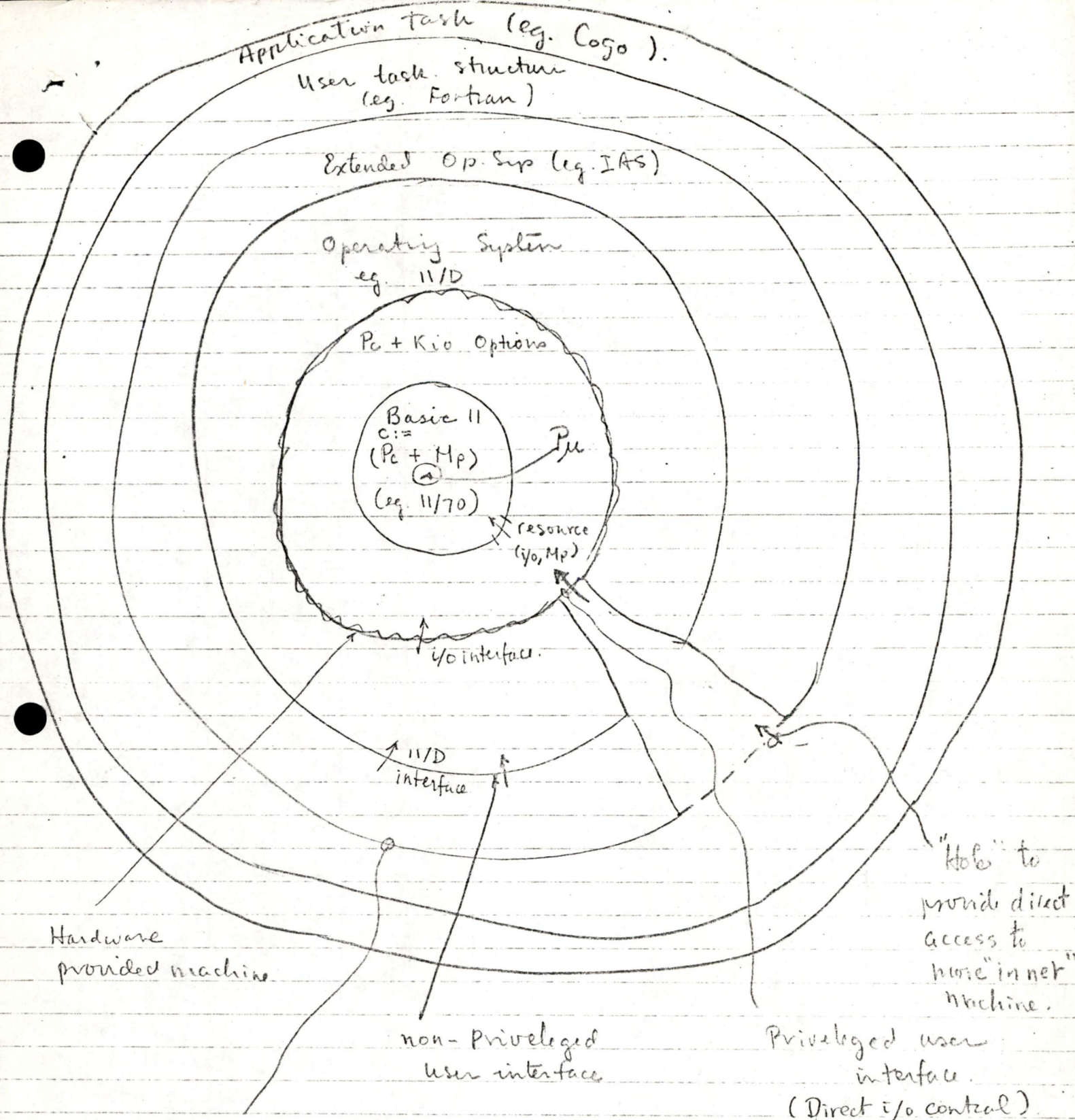
DATE:
FROM:

PAGE 3
03-31-75
GORDON BELL

This will enable us to: first, find out what the machines really are; second, evaluate the investment we have; and finally, estimate the investment of our users,

GB:mjk

Attachments (2)



Note each level, provides a particular machine with generally increasing capability

Fig. 1 Ring representation of I.I. Machines. (specific instance).

Machine	Provides	Build-on.	Provision in - 11extended
11/User mode	32 K Protected in 8-64~8K byte Segs.	11's & (04,05,20). (KT)	yes, Segs $\geq 2^{10}$
11/VAX User.	22^{24} K protected.	all but, except LSI-11, PDQ	Main interface
11/20	11/20 hardware. 28K + 4K i/o page, Interrupts, All 11's		Buildable w/ software (useful)
11/40KT	11/20; 124K phys + 4K i/o mappable onto 11/UM.	40/A40(BOS)	↑
11/45KT	11/40KT + Kernel + Supervisor Machines (\approx 11/UM); I+O-space; Default traps for Segs.	45	not provided in hdw. (Probably)
11/70KT	11/45KT + Unibus map into 22-bit physical memory		not useful in software ↓
11/D Prim. VM	11/UM w i/o page mapped into mem. space	40KT? ;	? - Can do except for 32 word i/o.
11/D VM	11/UM + D-ops + S/M-ops	11/UM	↑
11/S/M VM	11/UM + S/M-ops	11/UM	main interface ←
IAS VM	11/UM + IAS + D-ops	11/UM.	
KT VM	11/UM + RT-ops + 11/20?	?	redefine as
DOS	?	?	Buildable
RSTS (Int.)	?	?	Redefine as 11/D VM
Mumps	?	?	?
RSTS (Ext)	BASIC, Files	RSTS (Int.)	?
Mumps (Ext)	MUMPS Lang, FILES.	MUMPS (Int.)	?

Table 1. 11-FAMILY MACHINES AND WHAT THEY PROVIDE - 98 2/10/75

competitive update

NUMBER 27

MARCH 27, 1978

VAX11 780

**COMPETITIVE
PERFORMANCE
MEASUREMENTS
AND PRICING**

competitive update

<u>COMPETITOR</u>	<u>CONTACT</u>	<u>PRODUCT LINE AFFILIATION</u>	<u>MAIL STOP</u>	<u>DTN</u>
BURROUGHS	FRED BALFOUR	ISG	MK	264-5319
DATA GENERAL	JIM KINLAN	OEM	PK3/M28	223-3575
HEWLETT-PACKARD 1000	RICH BADER	LDP	MR2-4	231-5441
HEWLETT-PACKARD 3000				
Commercial Scientific/Lab	FRED BALFOUR DONNA GRAVES	ISG MDP	MK MR2/M79	264-5319 231-6413
IBM	LARRY TASHBOOK	BUS	MK	264-5410
INTERDATA/ PERKIN-ELMER	LOU PHILIPPON	OEM	PK3/M28	223-2038
MODCOMP	STEVE MIKULSKI	IPG	ML5-2/M17	223-3820
PRIME	TIM NEILL	ESG	MR1-1/M42	231-5162
TANDEM	FRED BALFOUR	ISG	MK	264-5319
TERMINALS	PAUL STEVENS	Terminals	MR2-2/M67	231-5357
TEXAS INSTRUMENTS	JIM KELLY	DCG	MR2-2/M65	231-6817
WIDE-WORD LENGTH	LOU PHILIPPON	OEM	PK3/M28	223-2038

The chairman of the committee is Gus Ashton, IPG Product Line

The following are trademarks of DIGITAL EQUIPMENT CORPORATION

COMPUTER LABS; DDT; DEC; DECUS; DIGITAL; FLIP CHIP; FOCAL; PDP;
TYPESET-8; UNIBUS; DECtape; IDAC; DIBOL; EDUSYSTEM; OMNIBUS; OS/8;
COMTEX; RSX; RSTS; DECCOMM; PHA; LAB-8; DECsystem-10; MASSBUS;
TYPESET-11.

COMPETITIVE UPDATE IS PUBLISHED BY SALES TRAINING DEPARTMENT,
BG/S51 MAYNARD.

EDITOR: DEANNA GRICCI 223-6353

COVER: BRENDA HAYWOOD 223-5255

EDITORIAL DIRECTOR: PAT MANNING 223-4373

PREFACE

COMPETITIVE DATA PUBLISHED IN COMPETITIVE UPDATE IS BASED ON RECOGNIZED INDUSTRY REPORTS, TRADE JOURNALS, AND CURRENT MANUFACTURERS' PUBLICATIONS AVAILABLE TO US. AS THE FIELD OF COMPETITIVE ANALYSIS IS NEVER STATIC, IT IS PREDICTABLE THAT THE INFORMATION IN THESE UPDATES WILL CONTINUALLY CHANGE. ACCORDINGLY, COMPETITIVE UPDATE ARTICLES WILL BE UPDATED AS WARRANTED.

LASTLY, WE REMIND YOU THAT COMPETITIVE UPDATE REPORTS ARE CLASSIFIED 'COMPANY CONFIDENTIAL'. ALTHOUGH THE CONTENTS CAN BE USED FOR SALES PURPOSES, ACTUAL PAGES OR COPIES OF THIS DOCUMENT ARE NOT TO BE DISTRIBUTED OUTSIDE THE COMPANY. 'DO NOT PASS YOUR AMMUNITION!' RECOGNIZING THAT RELIABLE INFORMATION IS EVERYONE'S RESPONSIBILITY, WE URGE AND ENCOURAGE YOUR INSIGHT, EXPERIENCE, AND INPUT.

CONTENTS

SECTION 1	VAX-11/780 FORTRAN PERFORMANCE AGAINST COMPETITION	
	INTRODUCTION.....	1
	PURPOSE	1
	VAX-11/780 VS. HARRIS	3
	Figure 1-1 Performance: VAX-11/780 vs. Harris	3
	Table 1-1 VAX-11/780 vs. Harris	3
	VAX-11/780 VS. INTERDATA	4
	Figure 1-2 Performance: VAX-11/780 vs. Interdata	4
	Table 1-2 VAX-11/780 vs. Interdata	4
	VAX-11/780 VS. PRIME	5
	Figure 1-3 Performance: VAX-11/780 vs. PRIME	5
	Table 1-3 VAX-11/780 vs. PRIME	5
	VAX-11/780 VS. SEL	6
	Figure 1-4 Performance: VAX-11/780 vs. SEL	6
	Table 1-4 VAX-11/780 vs. SEL	7
	CACHE PERFORMANCE	7
	BENCHMARK PROGRAM DESCRIPTIONS	8
SECTION 2	VAX-11/780 PRICE COMPARISONS WITH THE COMPETITION	
	Table 2-1 Pricing Summary: VAX-11/780 Packaged Systems vs. Eight Competitive Models	13
	Table 2-2 CPU Options and Prices	14
	Table 2-3 Detailed Price Comparison With VAX-11/780 System 1	14
	Table 2-4 Detailed Price Comparison With VAX-11/780 System 2	15
	Table 2-5 Detailed Price Comparison With VAX-11/780 System 3	15

**SECTION 1 VAX-11/780
FORTRAN PERFORMANCE AGAINST COMPETITION**

Tom Rarich
X4744 ML3-4/E88

INTRODUCTION

The VAX-11/780 has been marketed for over two months now giving us a chance to run numerous benchmarks against competitive machines. We have begun to accumulate a library of programs and test results that should give you a better understanding of how the VAX-11/780 performance compares to the competition.

At the present time we are including information on VAX-11/780 FORTRAN performance against Harris, Interdata, PRIME, and SEL. Also included is a section showing how the VAX-11/780 cache improves program performance. This is of particular interest to highly technical customers who may have concerns about cached machines.

The competitive performance comparisons below are all FORTRAN programs; these are the only benchmark situations we have encountered so far. The programs that have been benchmarked are primarily CPU bound. Some programs process large data files, but the program emphasis is always on computation.

All programs have been measured on a VAX-11/780 system that included the optional Floating Point Accelerator. The FPA gives the customer the best FORTRAN performance, and is a small part of total system price. FORTRAN users should be encouraged to buy the FPA to get the maximum benefit from their VAX-11/780 system.

All programs were run in stand-alone mode on the VAX-11/780 to get an accurate measure of machine performance. These are *not* throughput or multi-programming tests. Elapsed time is the value measured for each program.

The material published here will be included in the forthcoming update of the VAX-11/780 Sales Guide which will be published by early Q4. By that time, we plan to include performance benchmarks against CDC and IBM. We will also report machine configuration and throughput measurements.

PURPOSE

The following charts show how competitive machines perform relative to the VAX-11/780. To convey this information, relative performance of a program on a competitive machine is computed as a percentage of the VAX-11/780's performance for the same program. Relative performance is defined as:

The speed of a program run on a competitor's machine
divided by
the speed of the same program run on a VAX-11/780 (with FPA).

Speed is merely 1/time. For example, if a program runs in 200 seconds on an SEL machine and 100 seconds on the VAX-11/780, the relative performance of that program on the SEL machine is 50% the performance on the VAX-11/780.

In all the competitive charts, the VAX-11/780 performance is given as 100% and competitive performance numbers for each program are expressed as a percentage of that performance. The better the competitor's performance, the larger the value.

No single benchmark program is truly representative of a machine's performance. However, the range of values for many programs represents the range of capabilities of a machine compared to the VAX-11/780. We'll add more data points to these charts as the information becomes available.

Descriptions of all the benchmark programs are also included so you can identify particular programs which may relate to your customer's application.

For each competitor, comparison information is presented in two formats. At the top of each page is a scatter chart that shows the performance of one or more of the competitor's machines against the VAX-11/780. Each horizontal line represents the performance of one benchmark program on the competitor's machine.

At the bottom of each page is the detailed data from which the scatter plot was drawn. Each program is identified by name, and the execution time is shown for all machines that have been measured for that program. In each column, the performance percentage is shown next to the execution time. Each percentage is a vertical axis coordinate for the chart at the top of the page.

Following each program name, there is a suffix which indicates whether the program is:

- C — Complex single precision floating point
- D — Double precision floating point
- I — Integer
- S — Single precision floating point

Note that we do not have all the data for all programs on each machine.

Comparisons between competitors must be done carefully. For example, the chart of the VAX-11/780 vs. Harris shows two Harris machines; from the scatter chart, it is difficult to draw any conclusions about models /6 vs. /7. From the data table, however, one can see the relative performance of the two for the JACOBI and SP1111 programs. In both cases, the model /7 performs better than the model /6, but both are lower than the VAX-11/780.

VAX-11/780 VS. HARRIS

The Harris /6 (sold to end users as the model 115) is an older and slower machine than the model /7. We have data on only two programs on the /6. For these double precision programs, /6 performance is 55% to 70% of the VAX-11/780.

The new /7 model is encountered more frequently in competitive situations. The relative performance of the Harris /7 ranges from 23% to 82% of the VAX-11/780.

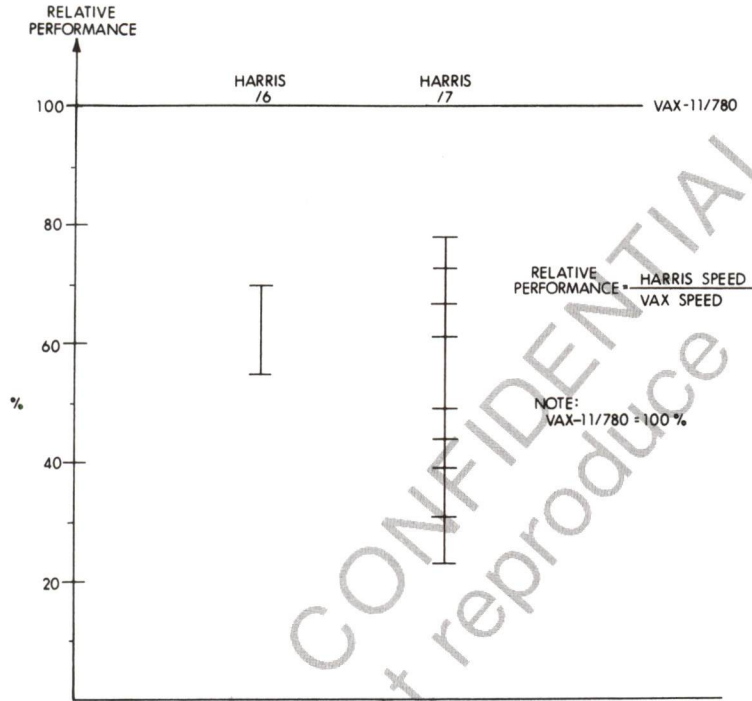


Figure 1-1
Performance: VAX-11/780 vs. Harris

Table 1-1
VAX-11/780 vs. Harris

Program Name	VAX +FPA (100%)	Harris/6		Harris/7	
	Seconds	Seconds	%	Seconds	%
JACOBI-D	367.0	667.0	55	597.0	61
SAAB1-S	8.0			35.2	23
SAAB2-S	15.8			40.3	39
SAAB3-S	9.9			20.2	49
SAAB4-I	30.6			41.8	73
SCRIP-D	21.3			32.0	67
SCRIP2-S	14.1			32.0	44
SCRIP3-C	11.7			38.0	31
SP1111-D	198.0	282.0	70	253.0	78

VAX-11/780 VS. INTERDATA

The relative performance of the Interdata 8/32 ranges from 27% to 62% of the VAX-11/780. The relative performance on integer calculations appears to be higher than floating point performance.

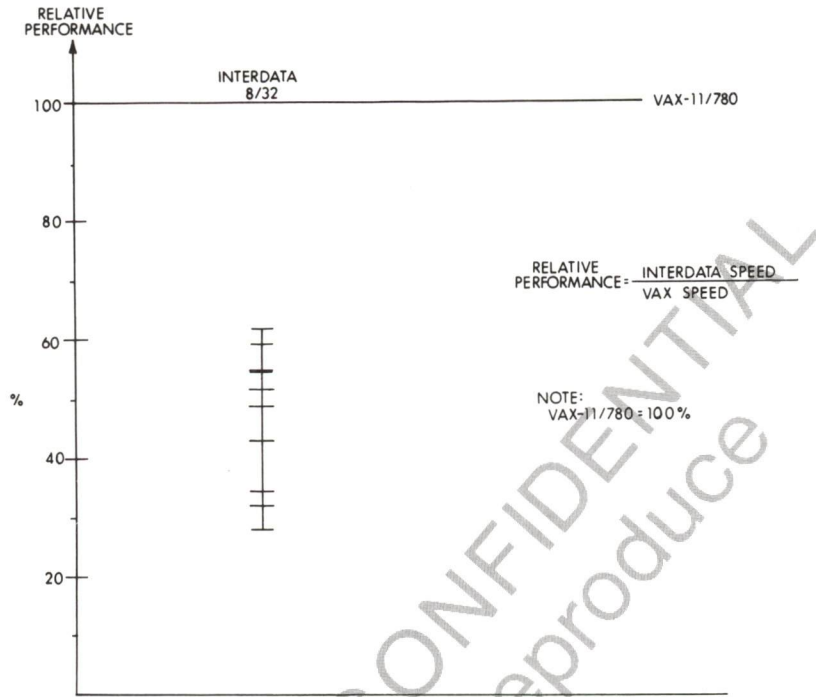


Figure 1-2
Performance: VAX-11/780 vs. Interdata

Table 1-2
VAX-11/780 vs. Interdata

Program Name	VAX +FPA (100%)	INTERDATA 8/32	
	Seconds	Seconds	%
JACOBI-D	367.0	665.0	55
LITTL-S	1.4	4.0	35
LUSTY-S	293.0	687.0	43
SAAB1-S	8.0	30.0	27
SAAB2-S	15.8	32.0	49
SAAB3-S	9.9	19.0	52
SAAB4-I	30.6	49.0	62
SCRIP3-C	11.7	37.0	32
SP1111-D	198.0	338.0	59
VALLEAU-S	308.0	558.0	55

VAX-11/780 VS. PRIME

The floating point relative performance of the PRIME 400 ranges from 11% to 42% of the VAX-11/780. The PRIME 500 floating point relative performance ranges from 25% to 50% of the VAX-11/780.

SAAB4 is an integer benchmark routine with a relative performance on the PRIME 400 of 102% of the VAX-11/780. It appears that the two machines are quite similar in performance when doing integer arithmetic.

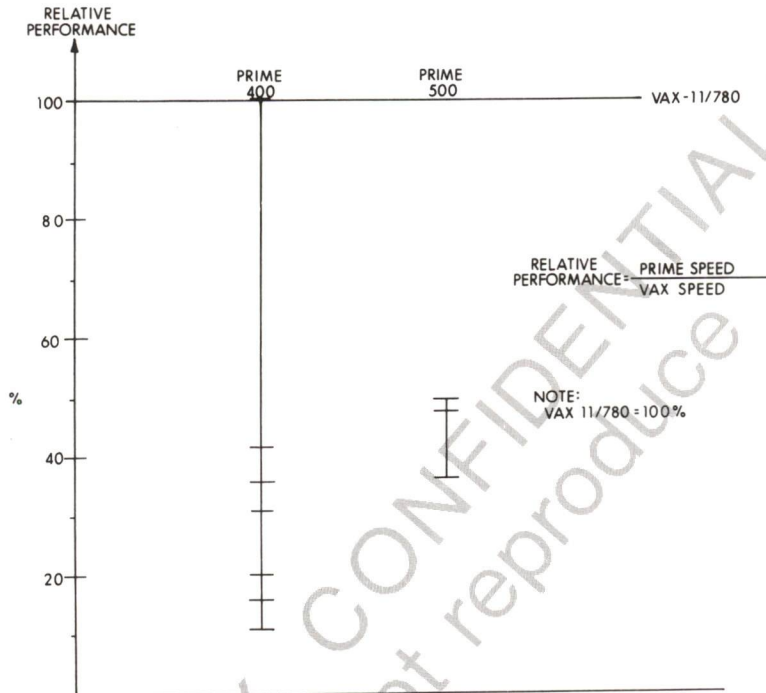


Figure 1-3
Performance: VAX-11/780 vs. PRIME

Table 1-3
VAX-11/780 vs. PRIME

Program Name	VAX +FPA (100%)	PRIME 400		PRIME 500	
	Seconds	Seconds	%	Seconds	%
JOLLA-S	12.9			27.0	48
JOLLA-D	300.0			600.0	50
NSC1A-S	3.2	8.9	36		
NSC1B-S	1.3	7.9	16		
NSC1C-S	5.6	13.3	42		
SAAB1-S	8.0	70.0	11		
SAAB2-S	15.8	80.7	20		
SAAB3-S	9.9	31.9	31		
SAAB4-I	30.6	29.9	102		
STELOS-S	219.0			615.0	36

VAX-11/780 VS. SEL

The relative performance of the SEL 32/55 ranges from 9% to 75% of the VAX-11/780. The performance of the 32/75, on a less complete set of programs, ranges from 45% to 73% of the VAX-11/780.

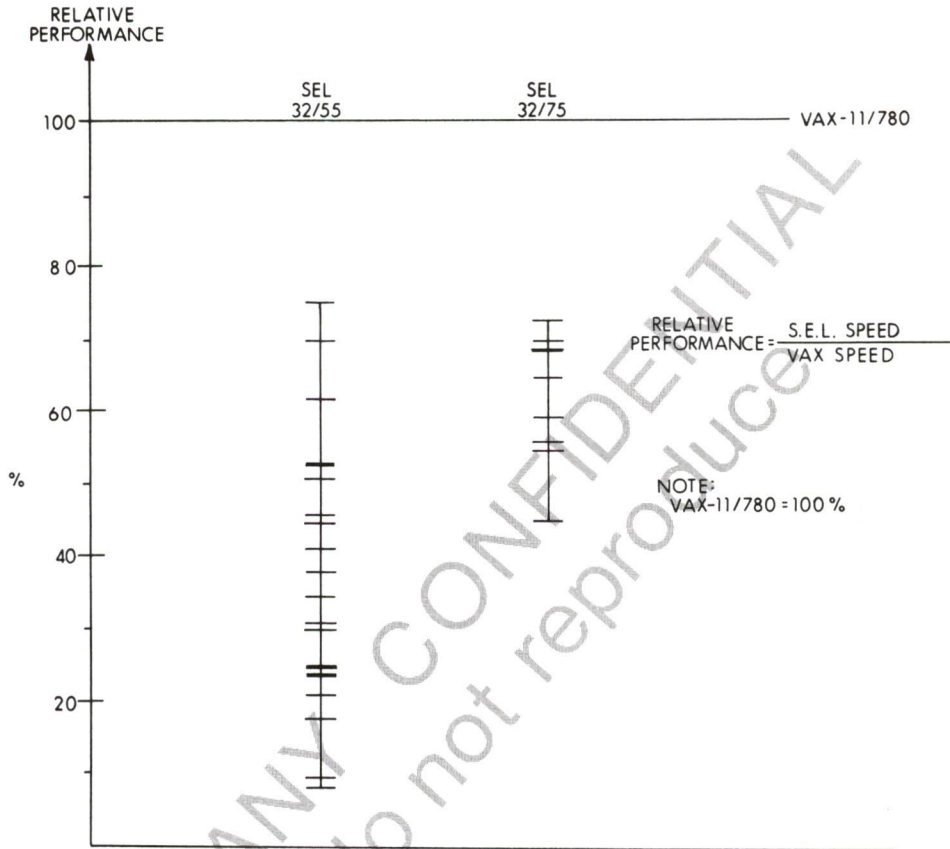


Figure 1-4
Performance: VAX-11/780 vs. SEL

**Table 1-4
VAX-11/780 vs. SEL**

Program Name	VAX +FPA (100%)	SEL 32/55		SEL 32/75	
	Seconds	Seconds	%	Seconds	%
LITTL-S	1.4	6.7	21		
LUSTY-S	293.0	574.0	51	426.0	69
NSC1A-S	3.2	6.0	53		
NSC1B-S	1.3	4.2	31		
NSC1C-S	5.6	7.5	75		
NSC2-S	0.8	9.5	8		
NSC2-I	1.3	7.1	18		
RAV1-S	.078	.316	25	.133	59
RAV2-I	3.5	5.0	70	4.8	73
RAV3-I	3.5	40.0	9		
RAV4-S	21.7	57.0	38	48.4	45
RAV5-D	33.9	113.0	30	61.8	55
RAV6-S	3.2	6.0	53	4.6	70
RAV7-D	.37	1.5	25	.57	65
RAV8-I	1.7	3.7	46		
SAAB1-S	8.0	34.0	24		
SAAB2-S	15.8	38.3	41		
SAAB3-S	9.9	28.1	35		
SAAB4-I	30.6	49.3	62		
SP1111-D	198.0	810.0	24	288.0	69
VALLEAU-S	308.0	684.0	45	553.0	56

CACHE PERFORMANCE

Technical customers in several markets have expressed concern about the VAX-11/780 because it is a cached machine. The usual concern is that their application is "different from most programs when referencing memory", and therefore cache will not do much for performance on their program.

Two characteristics are usually attributed to programs that "won't benefit" from cache:

- The program references a large data array in a random fashion.
- The program rarely loops.

We have run several programs on the VAX-11/780 with its cache disabled to measure the *actual* impact of cache on program speed. For a random set of nine programs, performance improvement due to cache ranged from a factor of 2.0 to 4.35.

One program (SP1111) is of particular interest because it has both the characteristics mentioned above. It manipulates over 50 Kb of data in a highly random fashion, and the 5,300 bytes of code in the inner loop subroutine contains only one loop. All other iterative sections of the program have been processed by a program generator which replaces DO loops with a set of

sequential FORTRAN statements. Even so, cache improves performance by a factor of 2.34 on this program! This happens because:

- The VAX-11/780 cache fetches 8 bytes of data for each reference to main memory. This causes even "straight line" coded programs to benefit from instruction pre-fetching. (Note that the VAX-11/780 fetches twice as much data into cache on each reference as the PDP-11/70, making the benefit of instruction stream referencing twice as large.)
- The VAX-11/780 cache is 8192 bytes (twice the size of the PDP-11/70), allowing much data to remain in cache. Even programs that reference data randomly often touch a datum more than once, so they also frequently benefit from cache.
- Stacks, constants, high-level language run-time support routines, and variables used as counters are always sources for "cache hit" memory references, no matter what the nature of the program.

If customers have concerns about cache, you should discuss these points with them. Stress also that cache is a concept that has been proven on many other machines, at Digital and in the industry.

If they remain skeptical, urge them to test the VAX-11/780 cache on our marketing demo machine. The VAX-11/780 is sufficiently complex and subtle that simple performance projections about cache are rarely accurate. The best projection is one done from actually running a program.

Cache can and does improve performance.

BENCHMARK PROGRAM DESCRIPTIONS

Jacobi

- FORTRAN
- Double Precision
- Size: 186 Kb
- Performs a Jacobi Diagonalization of a 100 x 100 double precision matrix of Eigenvectors. The matrix is first created, then diagonalized and later transferred to a second matrix and placed in ascending order. The matrix is checked by forming the product $U*U$ and tested to see if a unit matrix is obtained.

Littl

- FORTRAN
- Single Precision
- Size: 18 Kb
- A little program which computes 100,000 times:
 $A = ((x-1.)/(x+1.))*x*.39+1$
The result is compared to the known correct answer as a precision check.

Lusty

- FORTRAN
- Single Precision
- Size: 24 Kb
- Main routine and seven subroutines which perform calculations related to atomic particle motion and interactions.

NSC1

- FORTRAN
- Single Precision
- Size: 24 Kb
- Three part test of arithmetic capabilities:
 - NSCIA — Calculates 10,000 times:
 $ABS(SIN + x^{**}(ALOG) + EXP (-ATAN))$
 - NSCIB — Calculates 100,000 times:
 $(x-1)/(x+1) + x*.39 + A$
 - NSCIC — Sorts a 1,000 element R*4 array

NSC2

- FORTRAN
- Mixed
- Size: 22 Kb
- Two-part test of the local and global optimization capabilities of the FORTRAN compiler.
 - NSC2A — Calculates 327,660 times:
 $C = 2.2*BUF(1000)/113.3 + (999.-BUF(500))$
 - NSC2B — Calculates 327,660 times:
INTEG = $16*4$
ID = $256/4$
ITEMP = $16*IT/2 + (8-3)$

RAV1

- FORTRAN
- Double precision and single precision
- Size: 34 Kb
- Program does some geometric calculations including heavy use of trigonometric functions.

RAV2 and RAV3

- FORTRAN
- Integer
- Size: 18 Kb
- Performs 800,000 integer arithmetic operations (add, multiply, and divide). RAV2 uses the machine default integer size (I*4 on the VAX-11/780), and RAV3 is identical to RAV2 except that Integer*4 is specified in the program.

RAV4

- FORTRAN
- Single precision
- Size: 31 Kb
- Random program that does real, integer, and logical calculations on scalars and array elements.

RAV5

- FORTRAN
- Double precision
- Size: 45 Kb
- Double precision version of RAV4

RAV6

- FORTRAN
- Single precision
- Size: 18 Kb
- Calculates 10,000 times:
 $A = A + \text{SQRT}(\text{ABS}(\text{SIN}(X)) + X^{**}(\text{IFIX}(\text{ALOG}(X) * 0.43429448)) + \text{EXP}(-\text{ATAN}(X/666)))$

RAV7

- FORTRAN
- Double Precision
- Size: 18 Kb
- Calculate 1000 times:
X=1
Y=2.D0*PI/X
Y=1.D0/X
X=DSQRT(DABS(DSIN(X)+DCOS(X)+DATAN(Y)))

RAV8

- FORTRAN
- Integer
- Size: 44 Kb
- Manipulates eight 1605 element arrays to perform a histogram calculation, accumulating results in a 200 element array.

SAAB

- FORTRAN
- Mixed
- Size: 23 Kb
- A four part test program:
 - SAAB1 — Euler angles transformation between aircraft coordinate system and earth-bound system. Transformation of three angles is done 49130 times. Most of program is SIN, COS, or Multiplication.
 - SAAB2 — Simple 5 degrees of freedom aircraft model which is looped through 40,000 times
 - SAAB3 — Interpolation routine which uses a slowly changing variable. 401,000 values are fetched from the routine.
 - SAAB4 — Packing and unpacking of bit flags from a logical variable. A series of AND, OR, NOT operations is repeated 50,000 times. All integer arithmetic.

Scrip

- FORTRAN
- Double Precision — FFT
- Size: 282 Kb
- Performs a double Fast Fourier Transform of a 16,384 element double precision matrix. The root mean square of the result is then computed.

Scrip2

- FORTRAN
- Single Precision — FFT
- Size: 151 Kb
- Single precision version of SCRIP program

Scrip3

- FORTRAN
- Complex — FFT
- Size: 282 Kb
- Complex variable, single precision version of SCRIP program

SP1111

- FORTRAN
- Double precision arithmetic
- Size: 81 Kb
- An inner loop subroutine from a Quantum mechanics application package which solves the Schrodinger Equation for atoms and molecules. SP1111 is the inner loop of a 6-dimension Gaussian Integral. It has been produced by a code generator which produces mostly in-line code. DO loops have been removed and replaced by a sequence of separate FORTRAN statements. The subroutine is executed 5000 times.

Stelos

- FORTRAN
- Single precision, heavy trig usage
- Size: 21 Kb
- Astrophysics routine which performs a calculation, using many trig functions which produces "Heliocentric Distance" as the resulting answers. The answers are stored in a print file.

Valleau

- FORTRAN
- Single precision
- Size: 70 Kb
- A type of atomic particle physics calculation

COMPANY CONFIDENTIAL
do not reproduce

**SECTION 2 VAX-11/780
PRICE COMPARISONS WITH THE COMPETITION**

Tom Rarich
X4744 ML3-4/E88

Though the VAX-11/780's performance is important in selling situations, comparisons against competition are incomplete without price. The VAX-11/780 may offer only 10% the performance of a CDC 7600, for example, but if the VAX-11/780 system's price is only 3% of a 7600, a customer may be very interested.

Price comparisons, however, must be carefully made to insure that similar hardware configurations are being compared. The following charts show how each of the VAX-11/780 packaged systems compares to similarly configured competitive systems. (Note that our standard systems do not contain the optional FPA.)

The Pricing Summary chart (Table 1) is a one page reference; each column in the summary chart is supported by detailed charts which show all of the hardware and software components that are included in the total system price. Prices of commonly quoted options are shown in Table 2 to develop system prices in competitive situations.

In order to develop complete cost comparisons, a chart of monthly maintenance charges is included in Table 1. Note that the VAX-11/780 maintenance is *very attractively priced*. This is a strong selling point that should not be overlooked.

**Table 2-1
Pricing Summary: VAX-11/780 Packaged Systems vs.
Eight Competitive Models**

	System 1 ¹		System 2 ²		System 3 ³	
	CPU	8 Hr. O.S.	CPU	8 Hr. O.S.	CPU	8 Hr. O.S.
	128 Kb		256 Kb		512 Kb	
	2-RK06		RM03		RP06	
	LA36		TE16		TE16	
	8 Lines	8 Hr. O.S.	8 Lines	8 Hr. O.S.	8 Lines	8 Hr. O.S.
VAX-11/780 ⁴	\$128,600	\$ 588	\$153,000	\$ 722	\$185,000	\$ 832
Interdata 8/32	91,740	815	115,790	1040	157,350	1570
SEL 32/55	95,180	925	126,900	1060	184,600	1635
SEL 32/75	107,780	970	149,340	1260	182,200	1500
Harris 115 ⁵	106,450	800	135,000	1075	190,000	1525
PRIME 400	110,700	709	150,200	871	196,200	1159
PRIME 500	174,100 ⁶	911	191,600	1063	238,000	1351
HP 3000-II ⁷	111,245 ⁸	610 ⁸	118,645 ⁸	652 ⁸	181,445 ⁹	922 ⁹

1 SV-AXKKA-LA

2 SV-AXTVA-LA

3 SV-AXCVA-LA

4 Remote diagnosis standard.

5 Model 115 is end-user equivalent to the OEM model /6.

6 Minimum memory size is 256 Kb.

7 HP claims remote diagnosis capabilities on the 3000-II series.

8 Model 6

9 Model 8

**Table 2-2
CPU Options and Prices**

	VAX-11/ 780	Interdata 8/32	SEL 32/75	Harris	PRIME 500	HP 3000-II
Memory						
Type	ECC MOS	Core	Core (900 nsec)	MOS	ECC MOS	MOS
Increment	128 Kb	128 Kb	128 Kb	48 Kb ¹	256 Kb ²	65 Kb
Price	\$8K	\$13K-16K (estimate)	\$13K	\$5.5K	\$31K	\$4K
FPP Price	\$9.9K	\$6.5K	\$6K	\$10.5K	Std.	No
WCS Price	\$10K	\$4K	\$10K	No	?	No

- 1 24-bit word length
2 Minimum increment

**Table 2-3
Detailed Price Comparison With VAX-11/780 System 1¹**

	VAX-11/780 System 1 ¹	Interdata 8/32	SEL 32/55	SEL 32/75	Harris Model 6	PRIME 400	PRIME 500	HP 3000-II ²
With FPP	No	No	No	No	No	Yes	Yes	No
Memory	128 Kb ECC	131 Kb	131 Kb	131 Kb	144 Kb	128 Kb ECC	256 Kb ECC	128 Kb
Console	LSI-11 + floppy	Lights + switches	Lights + switches	Lights + switches	Lights + switches	Lights + switches	Lights + switches	Lights + switches
Terminal	LA36	Carousel	KSR	KSR	CRT	Terminet	Terminet	Printing Terminal
Disk	28 Mb	30 Mb	30 Mb	30 Mb	28 Mb	24 Mb	24 Mb	50 Mb
Tape	None	None	None	None	9 tr	None	None	9 tr
Lines	8	8	8	8	8	8	8	16
Software	VAX/VMS	OS/32 MT	RTM	RTM	Vulcan + all Lang.	PRIMOS + FORTRAN	PRIMOS + FORTRAN	MPE-II
TOTAL (\$)								
System	128,600	91,740	95,180	107,780	106,450	110,700	174,100	111,245
Maint.	588	815	925	970	800	709	911	610

1 SV-AXKKA-LA (dual RK06)

2 Model 6

Table 2-4
Detailed Price Comparison With VAX-11/780 System 2¹

	VAX-11/780 System 2 ¹	Interdata 8/32	SEL 32/55	SEL 32/75	Harris Model 6	PRIME 400	PRIME 500	HP 3000-II ²
With FPP	No	No	No	No	No	Yes	Yes	No
Memory	256 Kb ECC	256 Kb	262 Kb	262 Kb	242 Kb	256 Kb ECC	256 Kb ECC	256 Kb
Console	LSI-11 + floppy	Lights + switches	Lights + switches	Lights + switches	Lights + switches	Lights + switches	Lights + switches	Lights + switches
Terminal	LA36	Carousel	KSR	KSR	CRT	Terminet	Terminet	Printing Terminal
Disk	67 Mb	67 Mb	80 Mb (2)	80 Mb (2)	80 Mb	80 Mb	80 Mb	50 Mb
Tape	9 tr/45ips	9 tr/45ips	9 tr	9 tr	9 tr	9 tr	9 tr	9 tr
Lines	8	8	8	8	8	8	8	16
Software	VAX/VMS	OS/32 MT	RTM	RTM	Vulcan + all Lang.	PRIMOS + FORTRAN	PRIMOS + FORTRAN	MPE-II
TOTAL (\$)								
System	153,000	115,790	126,900	149,340	135,000	150,200	191,600	118,645
Maint.	722	1040	1060	1260	1075	871	1063	652

1 SV-AXTVA-LA (RM03/TE16)

2 Model 6

Table 2-5
Detailed Price Comparison With VAX-11/780 System 3¹

	VAX-11/780 System 3 ¹	Interdata 8/32	SEL 32/55	SEL 32/75	Harris Model 6	PRIME 400	PRIME 500	HP 3000-II ²
With FPP	No	No	No	No	No	Yes	Yes	No
Memory	512 Kb ECC	512 Kb	512 Kb	512 Kb	482 Kb	512 Kb ECC	512 Kb ECC	512 Kb
Console	LSI-11 + floppy	Lights + switches	Lights + switches	Lights + switches	Lights + switches	Lights + switches	Lights + switches	Lights + switches
Terminal	LA36	Carousel	KSR	KSR	CRT	Terminet	Terminet	Printing Terminal
Disk	176 Mb	256 Mb	160 Mb	160 Mb	160 Mb	300 Mb	300 Mb	150 Mb
Tape	9 tr/45ips	9 tr/45ips	9 tr	9 tr	9 tr	9 tr	9 tr	9 tr
Lines	8	8	8	8	8	8	8	16
Software	VAX/VMS	OS/32 MT	RTM	RTM	Vulcan + all Lang.	PRIMOS + FORTRAN	PRIMOS + FORTRAN	MPE-II
TOTAL (\$)								
System	185,000	167,540	184,600	182,200	190,000	196,600	238,000	181,445
Maint.	832	1570	1635	1500	1525	1159	1351	922

1 SV-AXCVA-LA (RP06/TE16)

2 Model 6

TM/1-2

+++++++
digital
+++++++

I N T E R O F F I C E

SEP 28 1978

TO: Gordon Bell
cc: Bernie Lacroute

DATE: 25 Sep 78
FROM: Tom Rarich *Tom Rarich*
DEPT: Mid-range Systems
EXT: 247-2120
LOC/MAIL STOP: TW/A08

SUBJ: Performance Information

As per your request for the latest VAX Performance Information, I am enclosing the data which is to be published next month in the VAX Sales Guide.

Because I wasn't sure what type of information you wanted, and because it is not yet in final form, I have divided the benchmark results into four categories as follows:

- 1) Finished form information from the May Sales Guide:
VAX versus HARRIS
Interdata
Prime
SEL
- 2) Scatter charts only (I can supply program names and times if you wish) for:
VAX versus Amdahl
CDC
IBM
SIGMA
UNIVAC
- 3) Scatter charts and backup data for DEC machine comparisons...
e.g.
VAX versus 11/70
2020
2040
2060
- 4) Brief descriptions of the benchmarks shown on the scatter charts.

I regret that the form of the attached material is not yet in finished form. I will see that you get a copy of the Performance Measurement section of the Sales Guide as soon as it is available.

Please call me if you have any questions on the attached material.

:c

Must remain as is, on separate page

RP

Results

Competitive Positioning of the VAX-11/780 The following charts show how various machines perform relative to the VAX-11/780. To convey this information, relative performance of a program on a competitive machine is computed as a percentage of the VAX-11/780's performance for the same program. Relative performance is defined as:

$$\frac{\text{The speed of a program run on a competitor's machine}}{\text{divided by}} \\ \text{the speed of the same program run on a VAX-11/780 (with FPA).}$$

Speed is merely 1/time. For example, if a program runs in 200 seconds on an SEL machine and 100 seconds on the VAX-11/780, the relative performance of that program on the SEL machine is 50% the performance on the VAX-11/780.

In all the competitive charts, the VAX-11/780 performance is given as 100% and competitive performance numbers for each program are expressed as a percentage of that performance. The better the competitor's performance, the larger the value.

No single benchmark program is truly representative of a machine's performance. However, the range of values for many programs represents the range of capabilities of a machine compared to the VAX-11/780. We'll add more data points to these charts as the information becomes available.

Descriptions of most of the benchmark programs are also included so you can identify particular programs which may relate to your customer's application.

For each competitor, comparison information is presented in two formats. At the top of each page is a scatter chart that shows the performance of one or more of the competitor's machines against the VAX-11/780. Each horizontal line represents the performance of one benchmark program on the competitor's machine. "tick mark"

At the bottom of each page is the detailed data from which the scatter plot was drawn. Each program is identified by name, and the execution time is shown for all machines that have been measured for that program. In each column, the performance percentage is shown next to the execution time. Each percentage is a vertical axis coordinate for the corresponding chart.

Following each program name, there is a suffix which indicates whether the program is

- C — complex single precision floating point
- D — double precision floating point
- F — File I/O is performed
- I — integer
- S — single precision floating point

what operation does:

Note that we do not have all the data for all programs on each machine.

Comparisons between competitors must be done carefully. For example, the chart of the VAX-11/780 vs. Harris shows two Harris machines; from the scatter chart, it is difficult to draw any conclusions about models /6 vs. /7. From the data table, however, one can see the relative performance of the two for the JACOBI and SP1111 programs. In both cases, the model /7 performs better than the model /6, but both are lower than the VAX-11/780.

tp

REVISED MAY 1978

VAX-11/780 vs. Harris

The Harris /6 (sold to end users as the model 115) is an older and slower machine than the model /7. We have data on only two programs on the /6. For these double precision programs, /6 performance is 55% to 70% of the VAX-11/780.

The new /7 model is encountered more frequently in competitive situations. The relative performance of the Harris /7 ranges from 23% to 82% of the VAX-11/780.

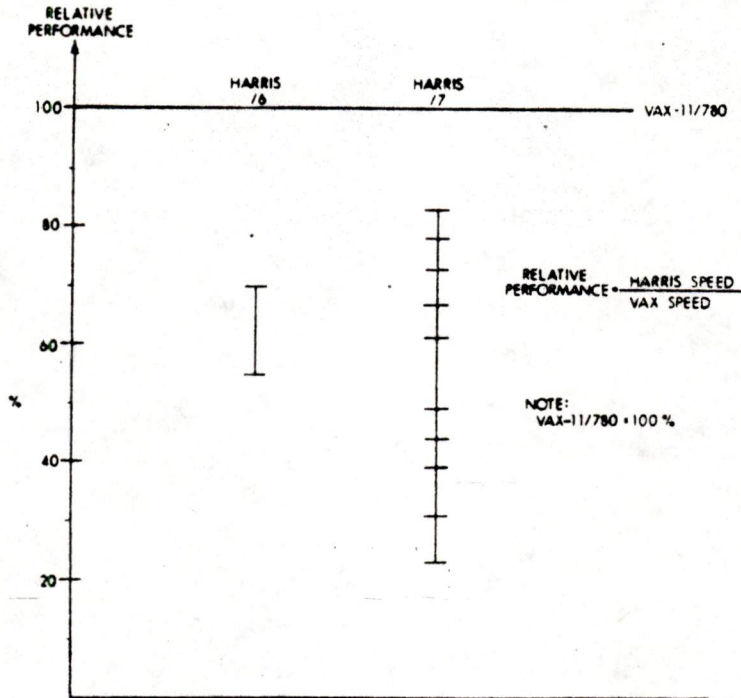


Figure 6-1
Performance: VAX-11/780 vs. Harris

Table 6-3
VAX-11/780 vs. Harris

Program Name	VAX +FPA (100%) Seconds	Harris/6		Harris/7	
		Seconds	%	Seconds	%
JACOBI-D	367.0	667.0	55	597.0	61
SAAB1-S	8.0			35.2	23
SAAB2-S	15.8			40.3	39
SAAB3-S	9.9			20.2	49
SAAB4-I	30.6			41.8	73
SCRIP-D	21.3			32.0	67
SCRIP2-S	14.1			32.0	44
SCRIP3-C	11.7			38.0	31
SP1111-D	198.0	282.0	70	253.0	78
STELoS-S	155.0			189.0	82

tp

REVISED MAY 1978

VAX-11/780 vs. Interdata

The relative performance of the Interdata 8/32 ranges from 27% to 62% of the VAX-11/780. The relative performance on integer calculations appears to be higher than floating point performance.

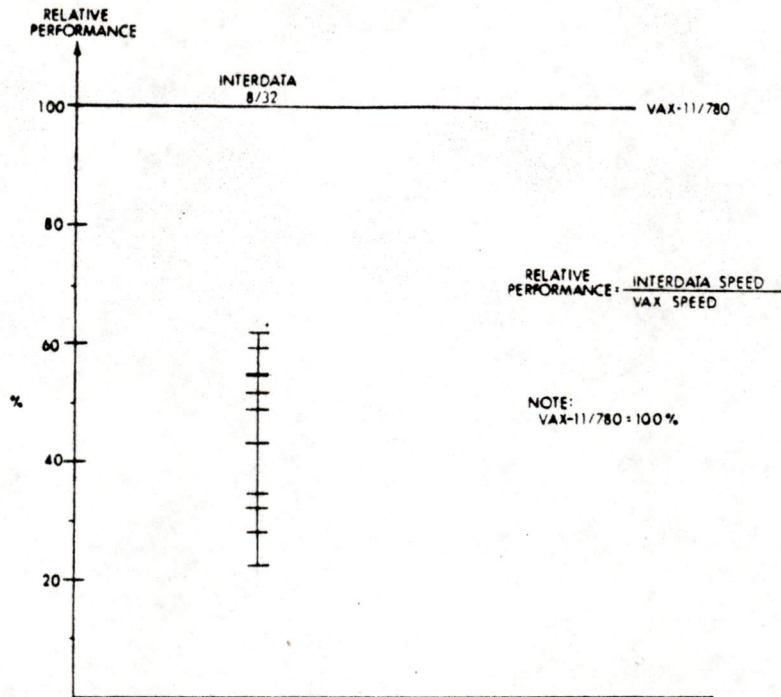


Figure 6-2
Performance: VAX-11/780 vs. Interdata

Table 6-4
VAX-11/780 vs. Interdata

Program Name	VAX +FPA (100%)	INTERDATA 8/32	%
	Seconds	Seconds	
JACOBI-D	367.0	665.0	55
LITTL-S	1.4	4.0	35
LUSTY-S	293.0	687.0	43
SAAB1-S	8.0	30.0	27
SAAB2-S	15.8	32.0	49
SAAB3-S	9.9	19.0	52
SAAB4-I	30.6	49.0	62
SCRIP3-C	11.7	32.0	37
SP1111-D	198.0	338.0	59
VALLEAU-S	308.0	558.0	55

tp

REVISED MAY 1978

VAX-11/780 vs. PRIME

The floating point relative performance of the PRIME 400 ranges from 11% to 42% of the VAX-11/780. The PRIME 500 floating point relative performance ranges from 25% to 50% of the VAX-11/780.

On integer benchmarks (SAAB4 and HANOI), it appears that the PRIME machines and the VAX-11/780 are quite similar in performance.

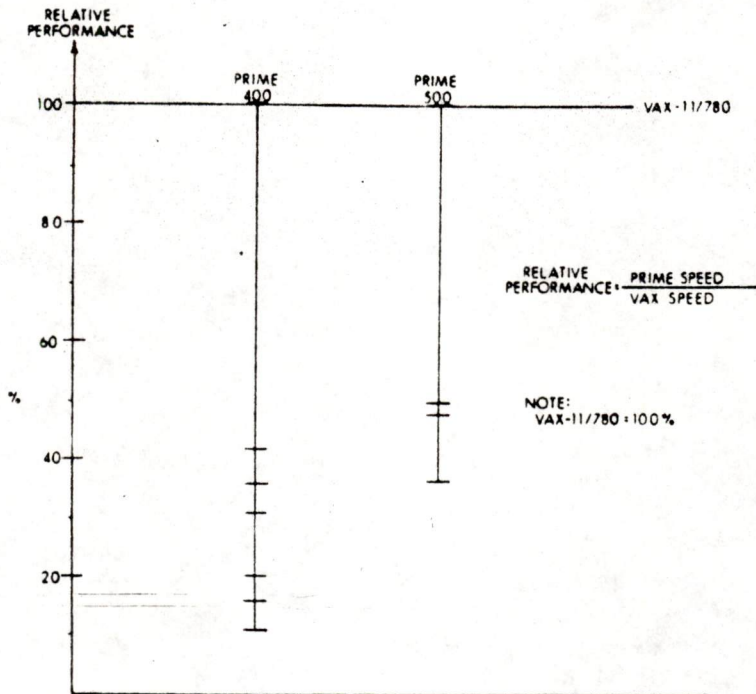


Figure 6-3
Performance: VAX-11/780 vs. PRIME

Table 6-5
VAX-11/780 vs. PRIME

Program Name	VAX +FPA (100%) Seconds	Prime 400		Prime 500	
		Seconds	%	Seconds	%
HANOI-I	18.5			18	103
JOLLA-S	12.9			27.0	48
JOLLA-D	300.0			600.0	50
NSC1A-S	3.2	8.9	36		
NSC1B-S	1.3	7.9	16		
NSC1C-S	5.6	13.3	42		
SAAB1-S	8.0	70.0	11		
SAAB2-S	15.8	80.7	20		
SAAB3-S	9.9	31.9	31		
SAAB4-I	30.6	29.9	102		
STELOS-D	219.0			615.0	36

REVISED MAY 1978

VAX-11/780 vs. SEL

The relative performance of the SEL 32/55 ranges from 9% to 75% of the VAX-11/780. The performance of the 32/75, on a less complete set of programs, ranges from 45% to 73% of the VAX-11/780.

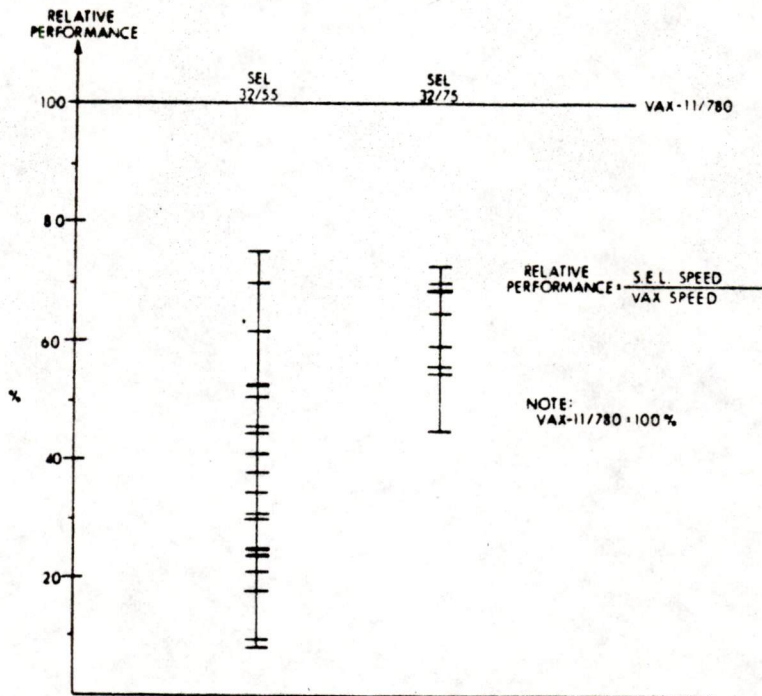


Figure 6-4
Performance: VAX-11/780 vs. SEL

Facing Page 6

tp

REVISED MAY 1978

Table 6-8
VAX-11/780 vs. SEL

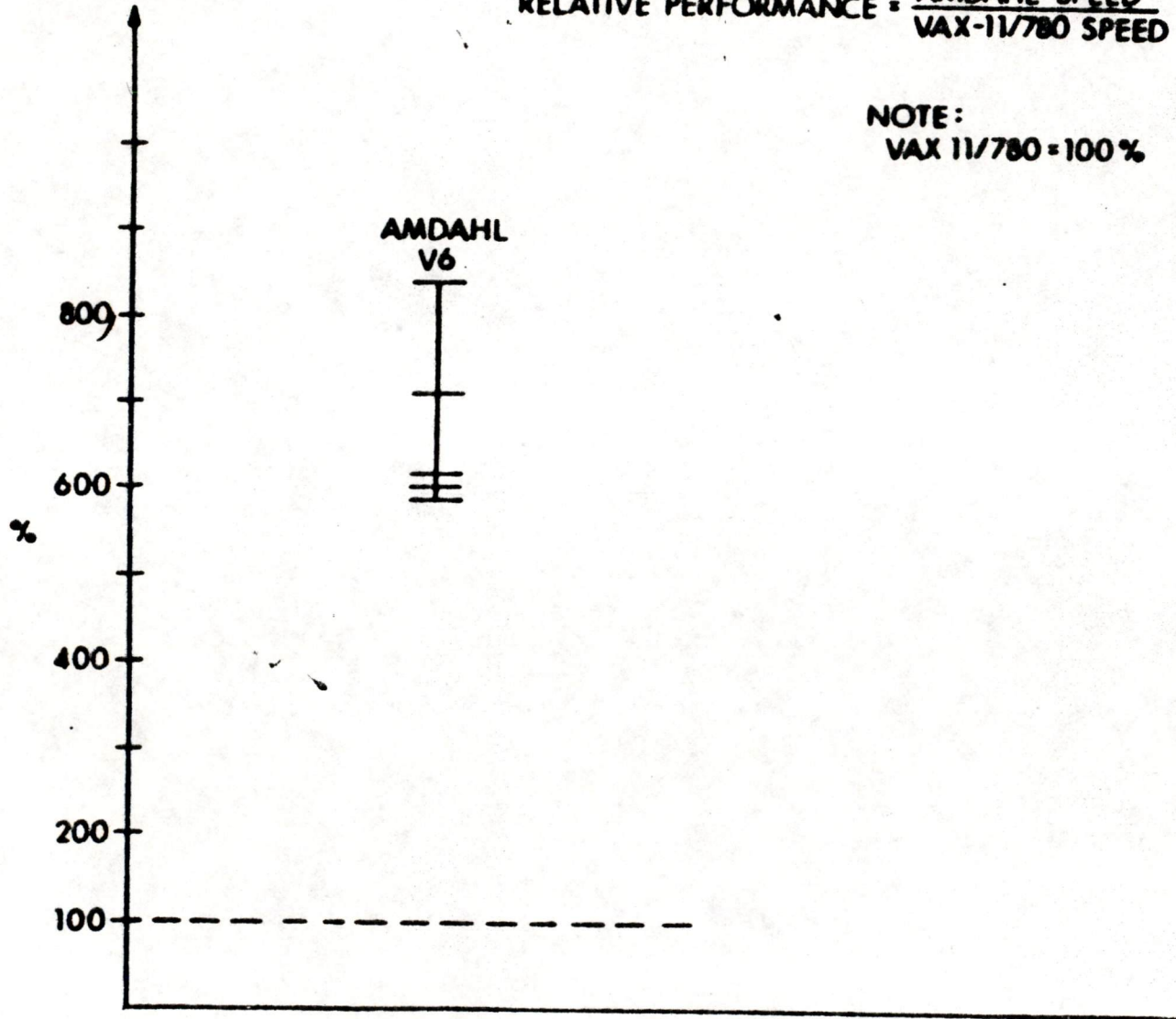
Program Name	VAX +FPA (100%)	SEL 32/55		SEL 32/75	
	Seconds	Seconds	%	Seconds	%
LITTL-S	1.4	6.7	21		
LUSTY-S	293.0	574.0	51	426.0	69
NSC1A-S	3.2	6.0	53		
NSC1B-S	1.3	4.2	31		
NSC1C-S	5.6	7.5	75		
NSC2-S	0.8	9.5	8		
NSC2-I	1.3	7.1	18		
RAV1-S	078	316	25	.133	59
RAV2-I	3.5	5.0	70	4.8	73
RAV3-I	3.5	40.0	9		
RAV4-S	21.7	57.0	38	48.4	45
RAV5-D	33.9	113.0	30	61.8	55
RAV6-S	3.2	6.0	53	4.6	70
RAV7-D	.37	1.5	25	.57	65
RAV8-I	1.7	3.7	46		
SAAB1-S	8.0	34.0	24		
SAAB2-S	15.8	38.3	41		
SAAB3-S	9.9	28.1	35		
SAAB4-I	30.6	49.3	62		
SP1111-D	198.0	810.0	24	288.0	69
VALLEAU-S	308.0	684.0	45	553.0	56

→ Facing Page 6-12 !!

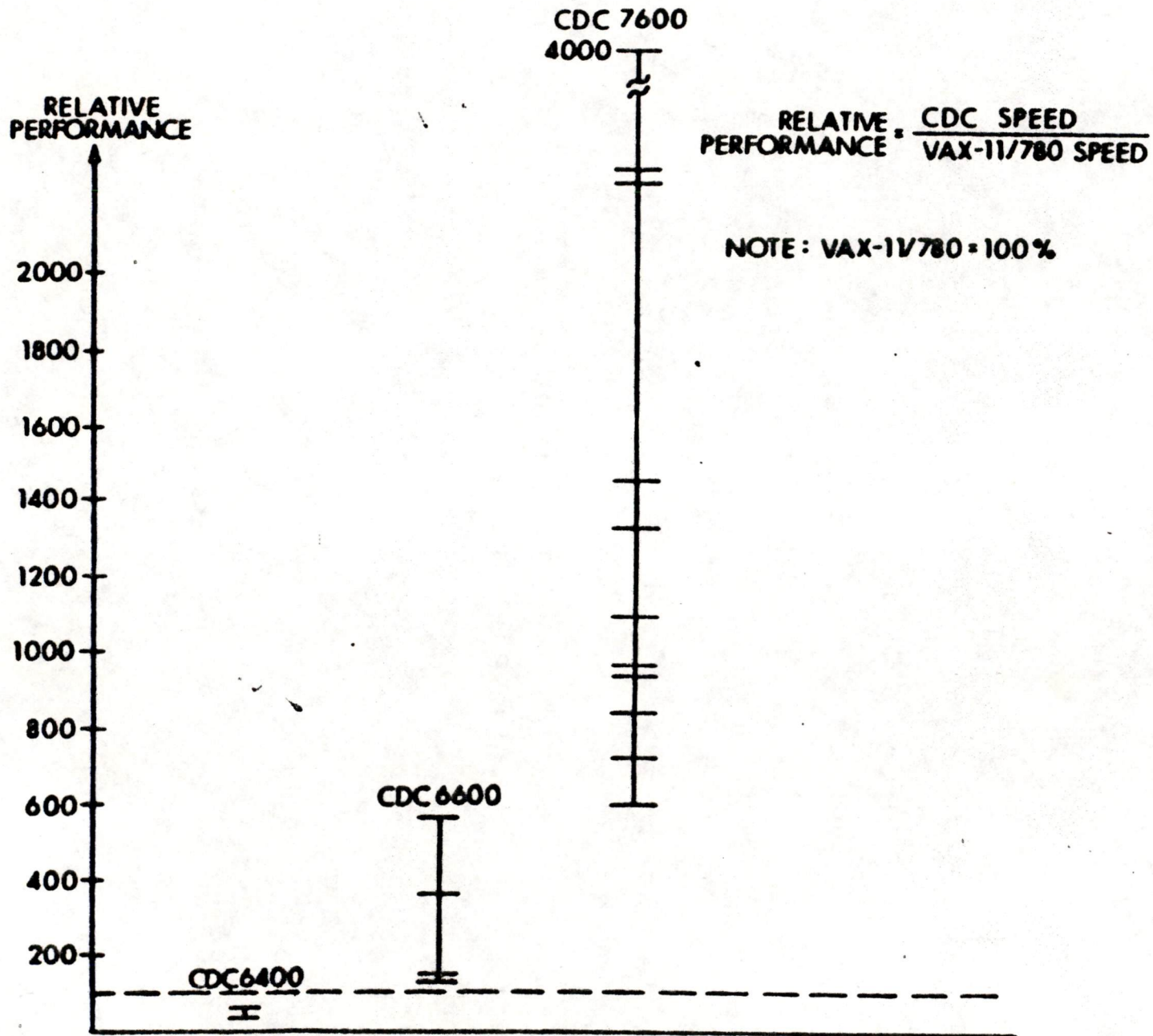
RELATIVE PERFORMANCE

$$\text{RELATIVE PERFORMANCE} = \frac{\text{AMDAHL SPEED}}{\text{VAX-11/780 SPEED}}$$

NOTE:
VAX 11/780 = 100%



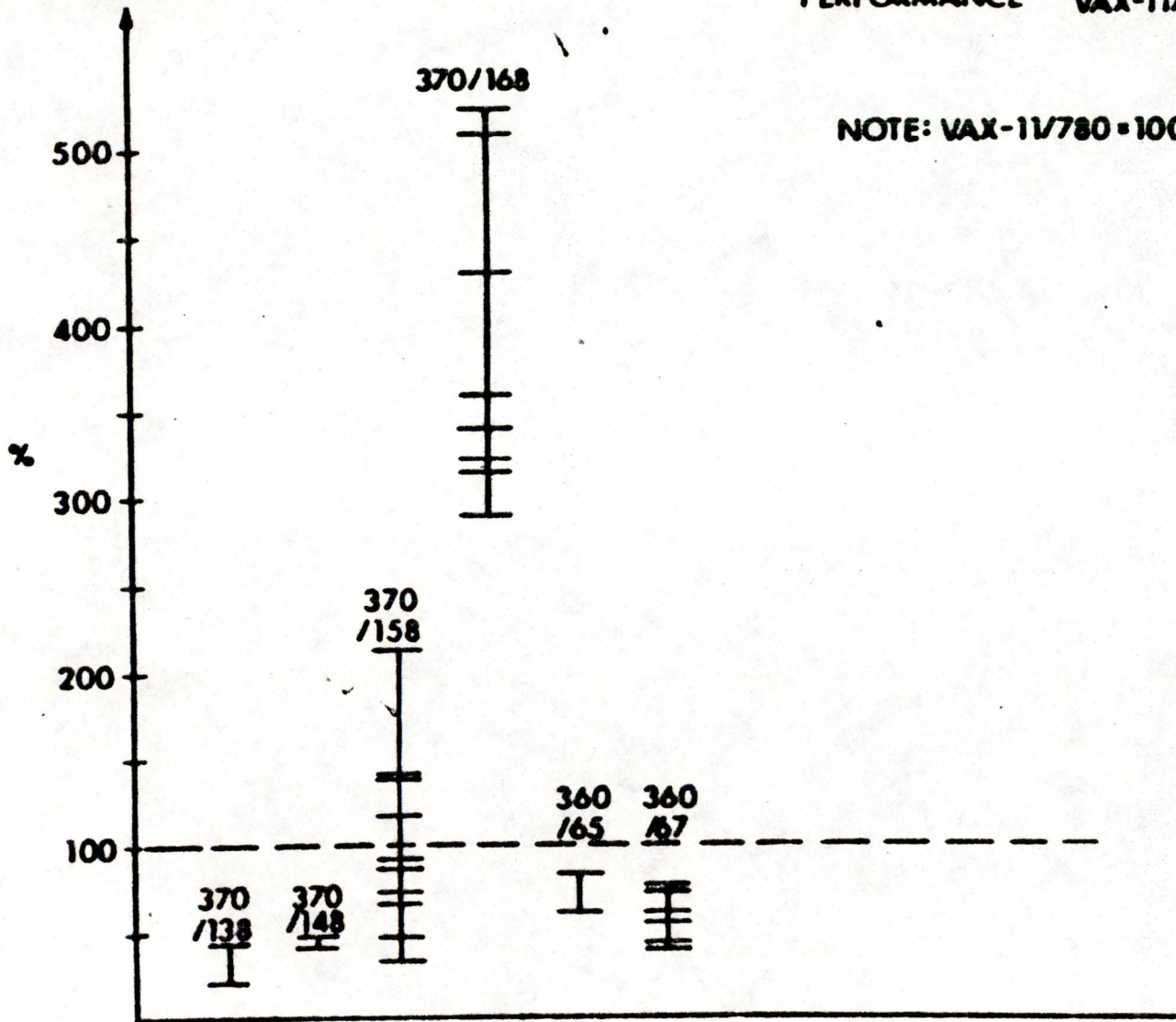
PERFORMANCE : VAX-11/780 vs AMDAHL



RELATIVE PERFORMANCE

$$\text{RELATIVE PERFORMANCE} = \frac{\text{IBM SPEED}}{\text{VAX-11/780 SPEED}}$$

NOTE: VAX-11/780 = 100%

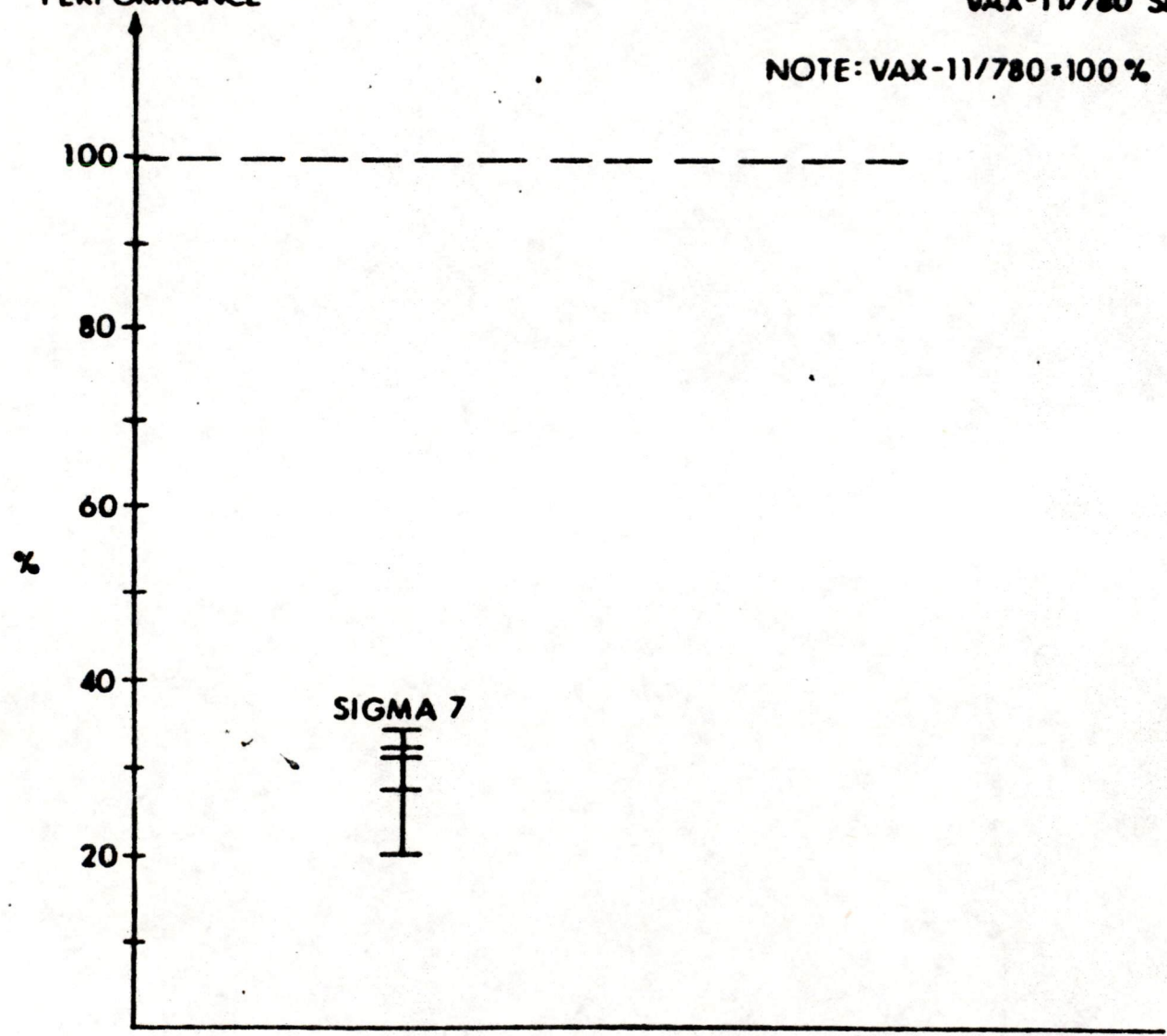


PERFORMANCE: VAX-11/780 vs IBM

RELATIVE PERFORMANCE

$$\text{RELATIVE PERFORMANCE} = \frac{\text{SIGMA SPEED}}{\text{VAX-11/780 SPEED}}$$

NOTE: VAX-11/780 = 100 %

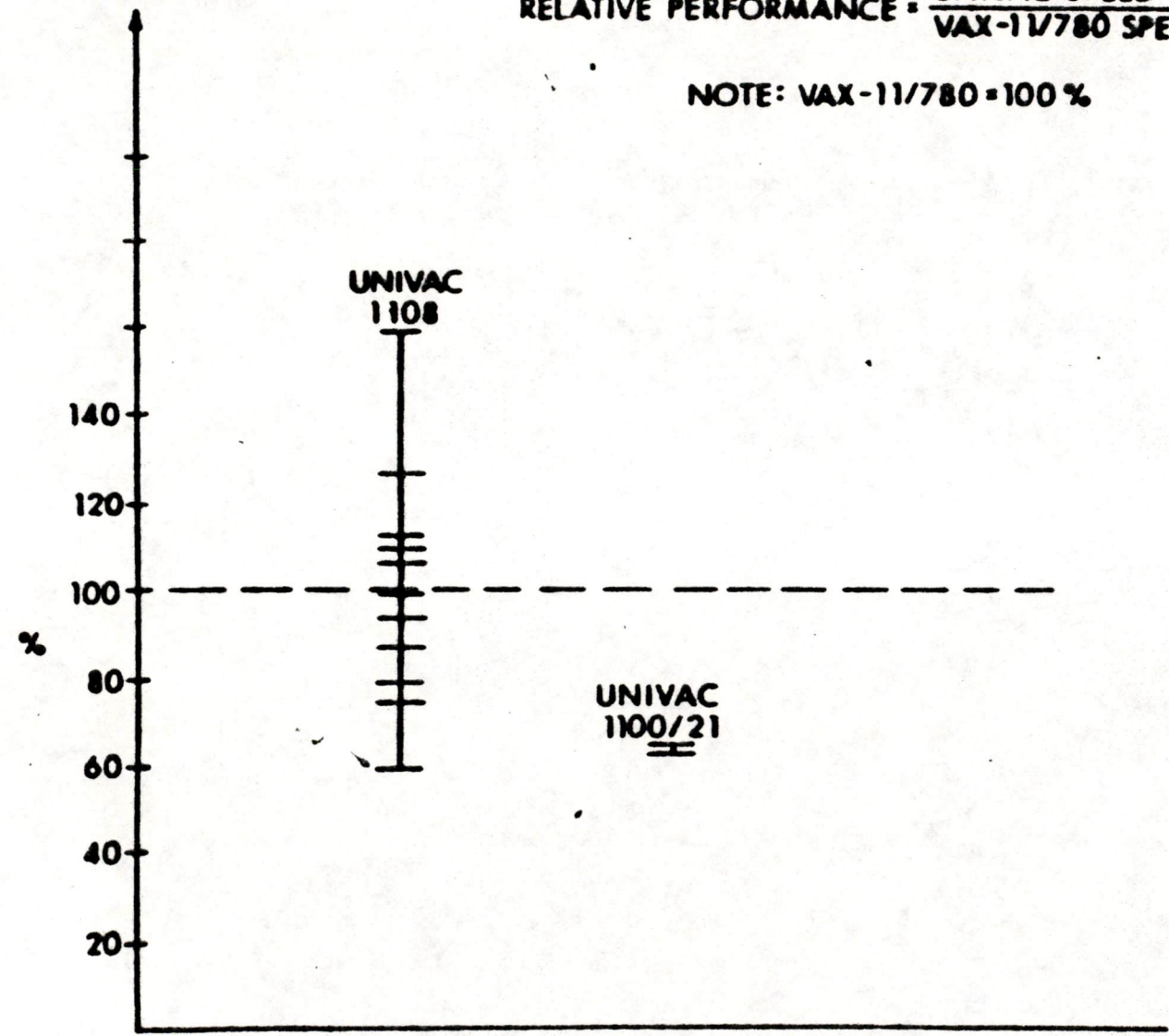


PERFORMANCE: VAX-11/780 vs SIGMA

RELATIVE PERFORMANCE

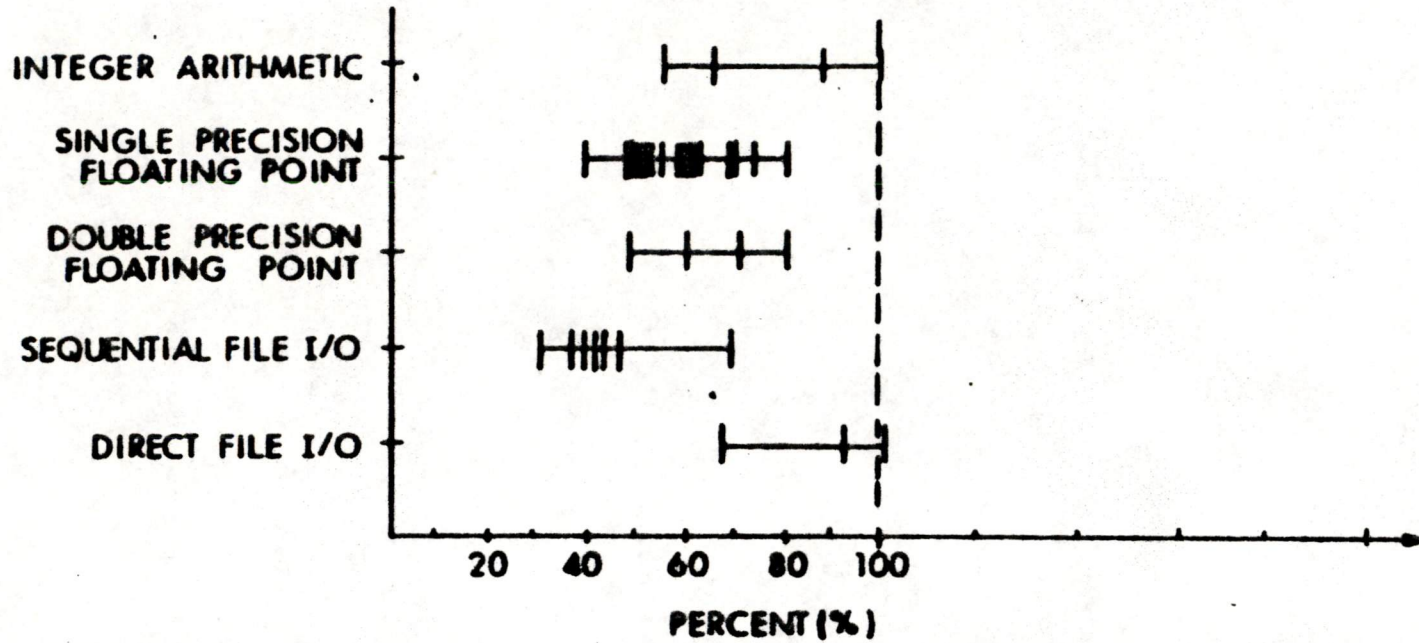
$$\text{RELATIVE PERFORMANCE} = \frac{\text{UNIVAC SPEED}}{\text{VAX-11/780 SPEED}}$$

NOTE: VAX-11/780 = 100 %

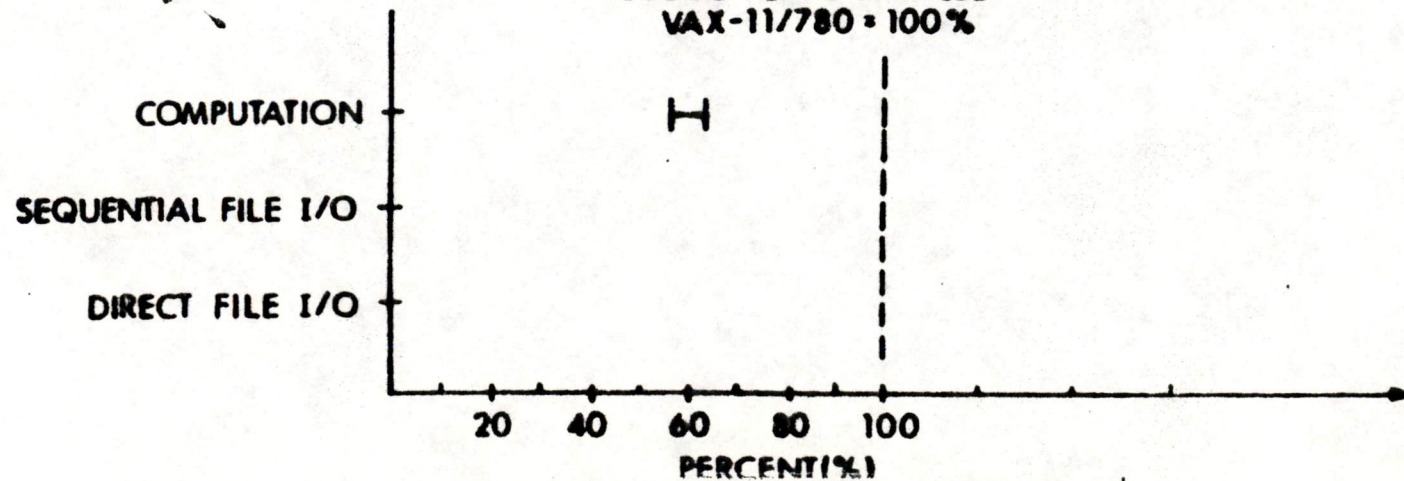


PERFORMANCE: VAX-11/780 vs UNIVAC

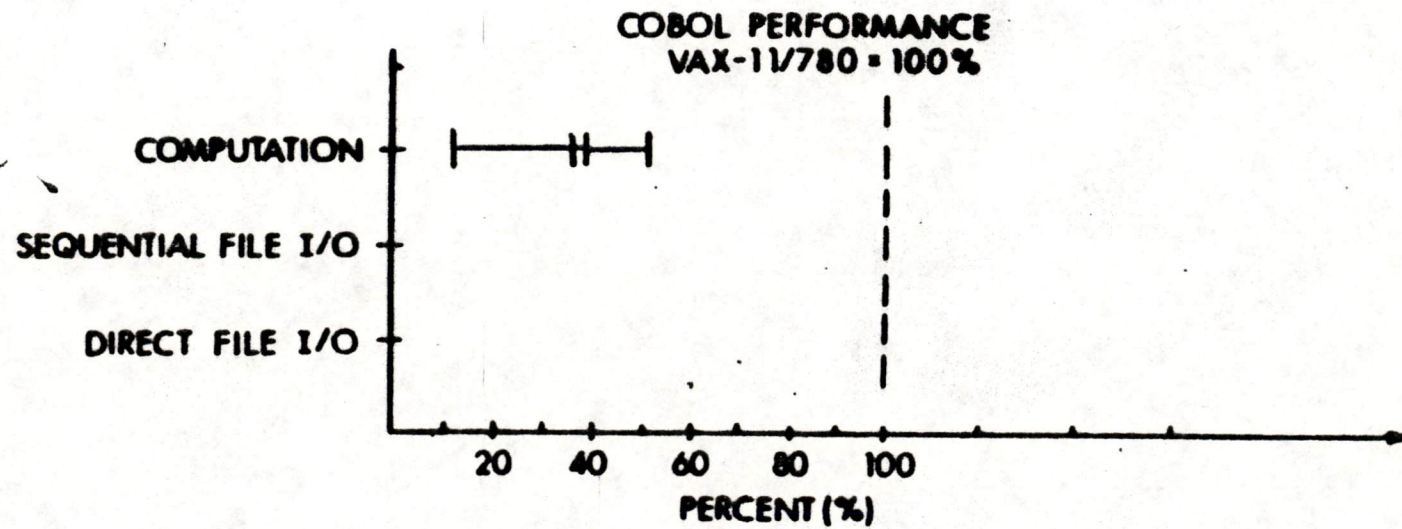
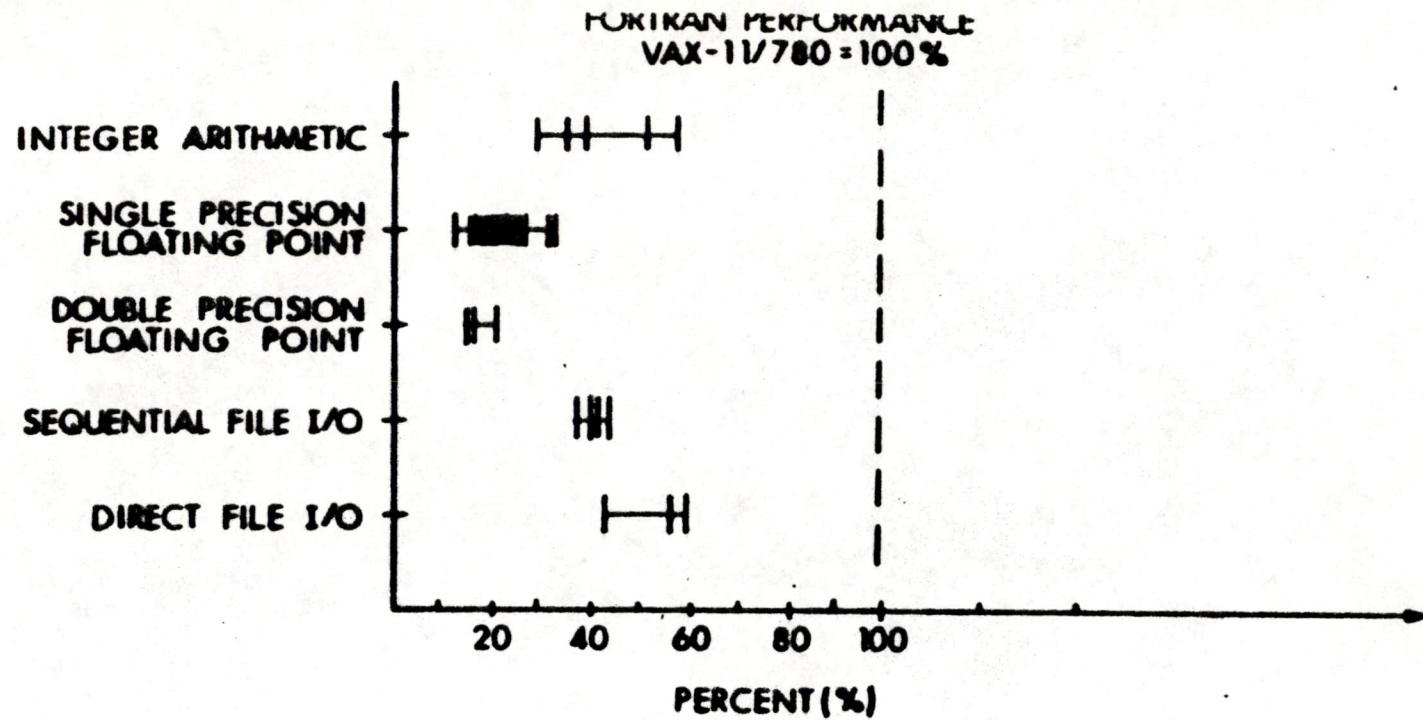
FORTRAN PERFORMANCE
VAX-11/780 = 100%



COBOL PERFORMANCE
VAX-11/780 = 100%

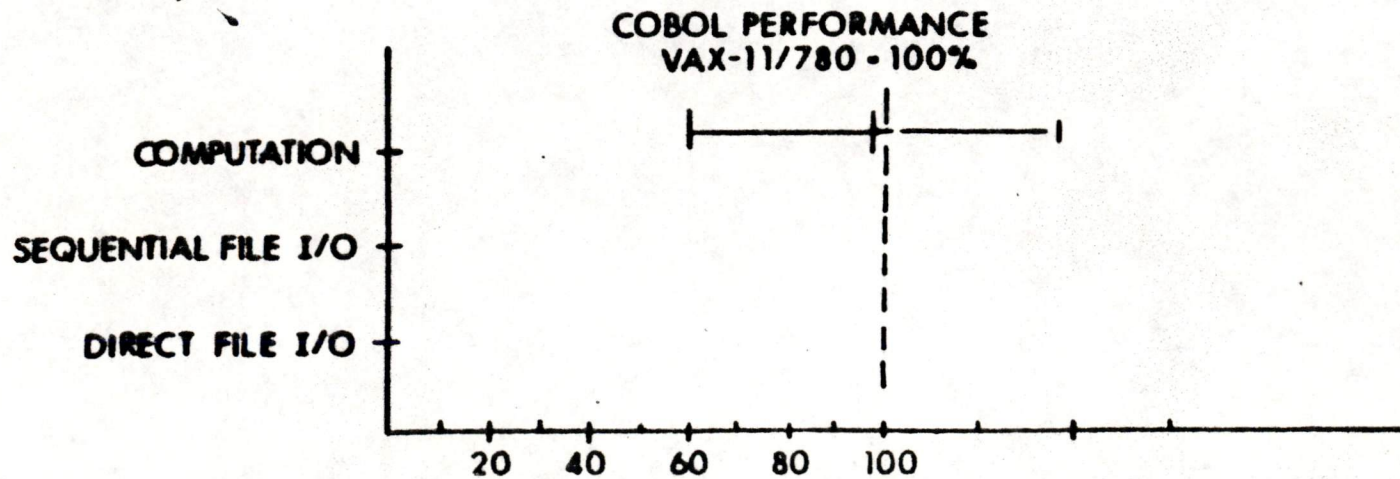
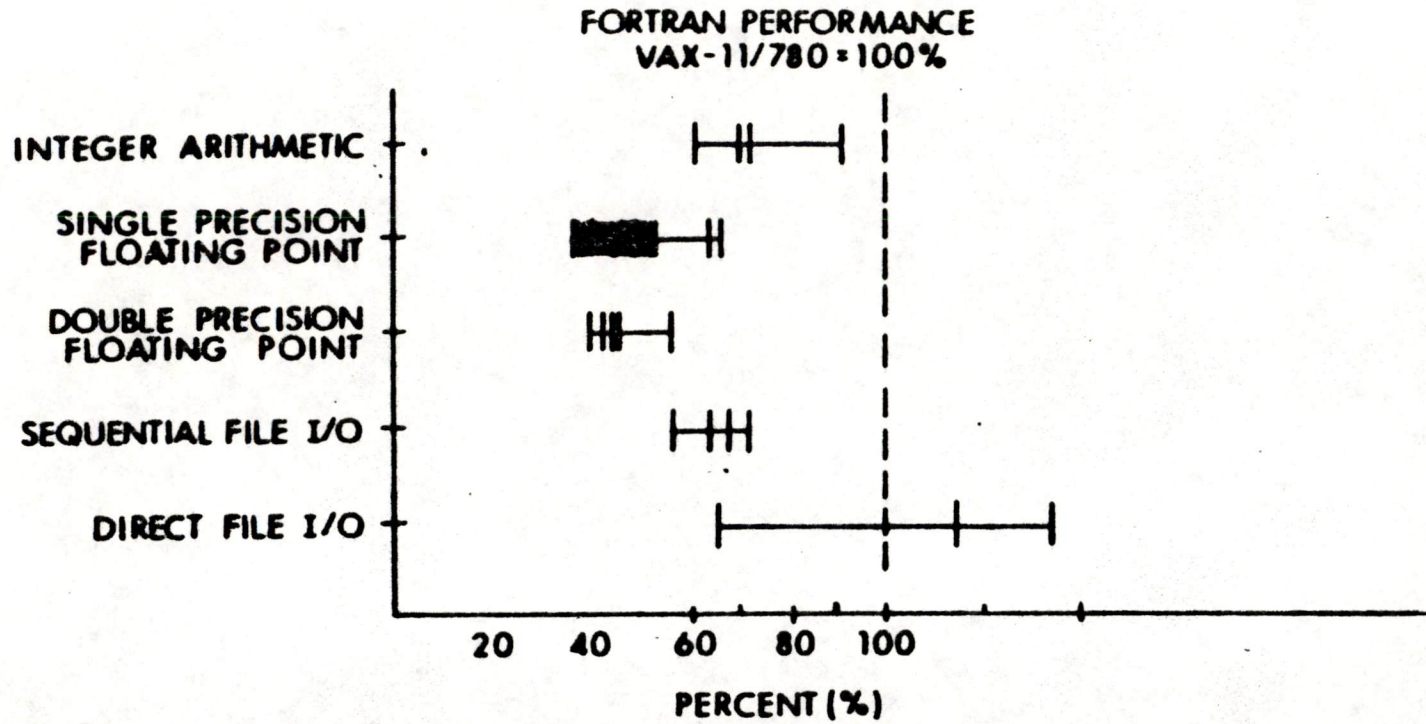


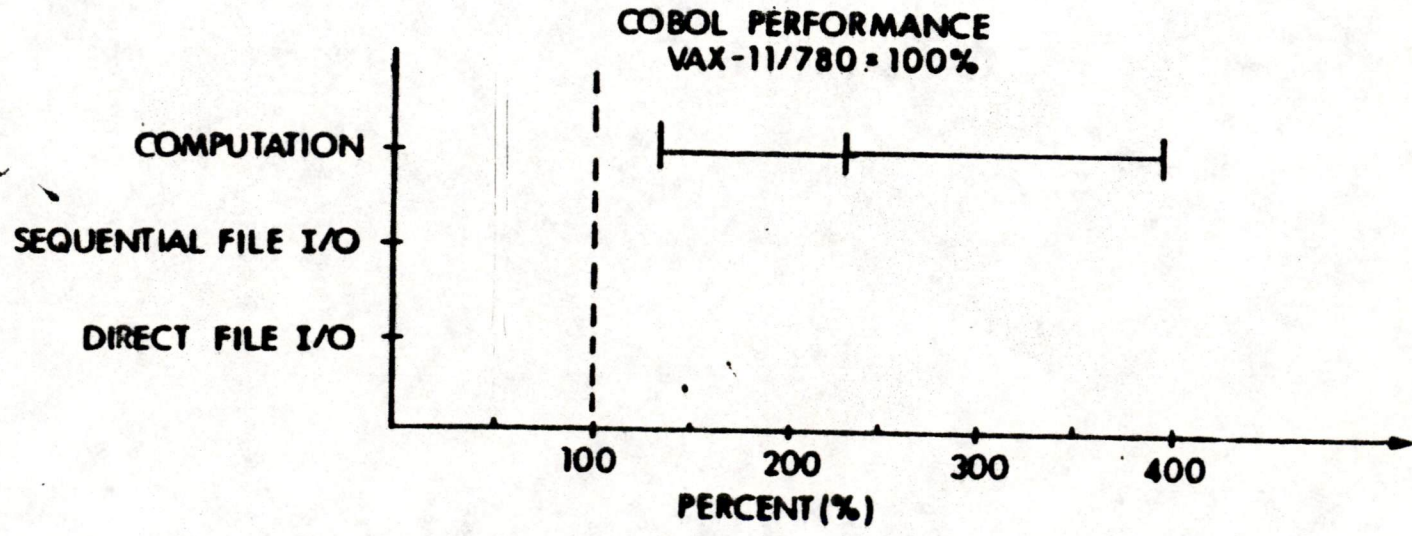
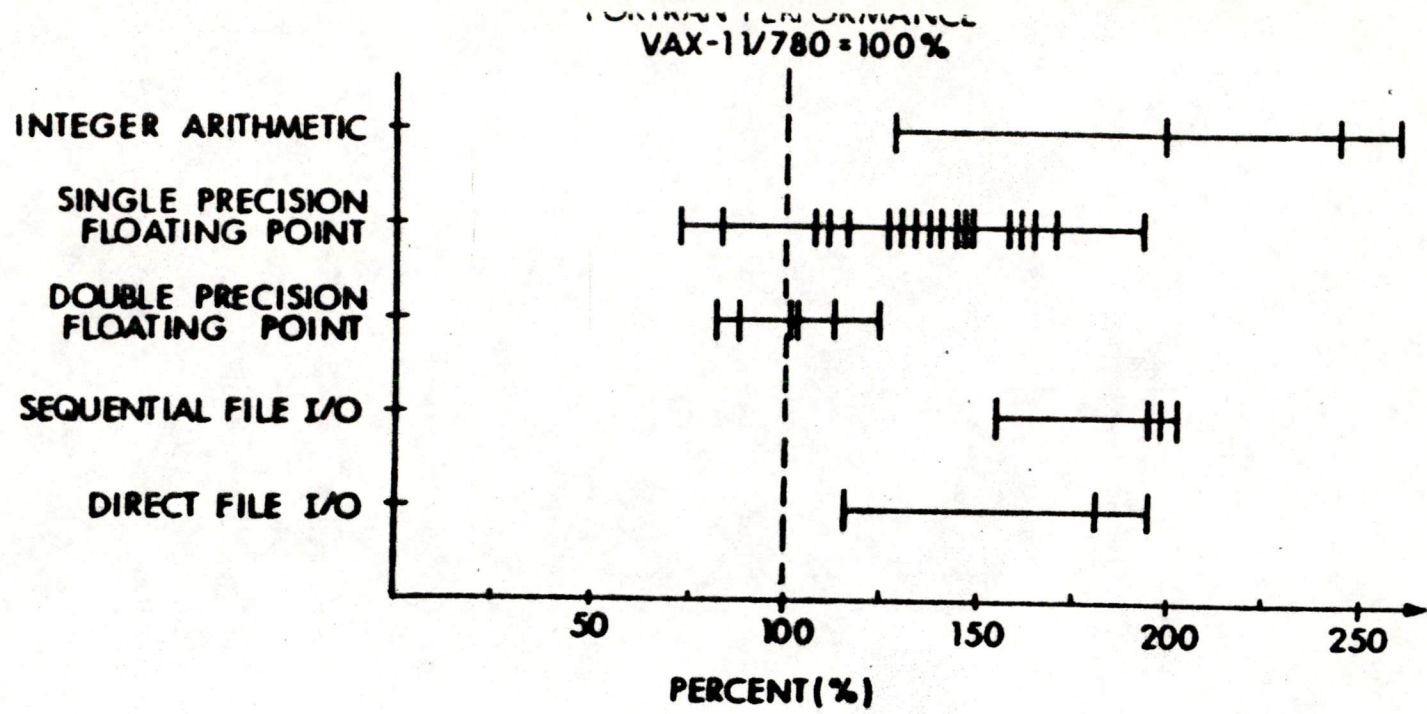
VAX-11/780 vs THE PDP-11/70



VAX-11/780 PERFORMANCE vs THE DECSYSTEM 2020

VAX VS 2040B





VAX-11/780 PERFORMANCE vs THE DEC SYSTEM 2060

	VAX-11	RSX-11M	IAS	RSTS/E	2020	2040	2060
Integer							
Airco-I	9.7	11.2	11.5	36.9	36.0	16.8	7.5
Airco Sub	3.9	6.0	6.1	24.8	8.0	6.9	1.7
Hanoi	18.3	26.2	27.0	145.4	33.4	26.7	7.2
NSC2-I	1.3	2.3	—	—	4.2	—	—
Prime	22.1	23.1	24.0	65.3	62.7	25.2	11.1
Single Precision Float							
Airco-R	0.6	1.1	1.1	4.3	4.3	1.7	0.8
Asea	11.9	22.4	23.2	78.4	65.9	24.6	8.4
CAE1	11.9	19.0	22.0	—	43.1	24.7	6.4
CAE2	101.8	184.0	—	—	195.8	80.0	—
DG3	1.6	4.1	4.2	16.2	9.4	3.8	1.6
Egypt	5.2	9.9	10.2	34.9	27.0	10.4	4.7
FFT45	0.9	1.7	1.7	5.1	5.7	2.0	0.8
Gauss	0.7	1.1	1.1	2.5	4.1	1.5	0.6
Hughes	7.0	14.6	15.1	79.1	28.0	14.1	4.4
Littl	1.4	2.3	2.4	11.4	11.1	3.5	1.7
Lusty	289.0	398.0	433.0	—	—	557.0	203.0
Matrix	1.2	2.5	2.6	10.4	4.7	2.9	0.8
Mflops	0.1	.	.	.	0.5	0.2	0.1
Neff	135.0	231.0	242.0	—	—	296.0	123.0
NSC1A	3.2	5.1	—	—	18.0	—	—
NSC1B	1.3	2.4	—	—	12.2	—	—
RR1	47.0	67.2	69.4	264.2	151.4	109.0	28.9
RR2	36.2	57.9	59.8	155.8	167.7	70.4	27.7
RR3	59.5	96.2	101.1	378.9	256.6	145.7	37.0
RR4	2.8	5.6	5.8	17.8	12.8	5.8	2.0
Single	1.2	1.5	1.5	6.3	5.2	1.9	0.9
Stelos	155.0	226.0	247.0	—	611.0	210.0	92.0
Double Precision Float							
Double	1.6	2.0	2.0	10.6	11.0	2.9	1.4
Egypt-D	8.0	13.1	15.0	—	47.7	17.6	9.0
Jacobi	361.0	.	.	.	—	814.0	344.0
Philco	9.5	18.6	19.3	63.9	52.1	23.9	7.8
SP1111	198.0	.	.	.	—	431.0	194.0
Single Precision Whetstone (thousands of instructions per second)							
	1168	699	694	233	242	512	1136
Double Precision Whetstone (thousands of instructions per second)							
	767	518	500	167	103	304	667

* Too large.

Table 6-12
COBOL Compute Times

	FP11-C	FP11-C	FP11-C	2020	2040	2060
	VAX-11	RSX-11M	IAS	RSTS/E		
Timtst						
Avg. Inst. (msec)	278.0 52	417.0	461.0	438.0	362.0	90.0 41.0
U.S. Steel (secs)	20.4 29	28.7	26.1	26.3	6.6	2.1 1.0
Prod. Index-D	10.0 27	8.0 63	8.0	8.0	32.0	88.0 208.0 213
Prod. Index-C	12.5 165	104			4584	235 702 660



BENCHMARK PROGRAM DESCRIPTIONS

The following section describes the programs referenced in the performance measurement charts. The programs were collected from real benchmark situations, and may be of interest to other customers.

The sizes given include the run time system as well as the compiler output. Hence a 500 byte user program may be shown as a 16 Kb image file.

Airco

- FORTRAN
- Integer, Real, Integer
- Size: 18 Kb
- Three part program:
 - AIRCO-I is:

```
DO 110 L=1,2
DO 110 I=1,591
K=0
DO 110 J=1,1000
ITEM=3*(J-1)/((J-K)*J)
110 K=K+1
```
 - AIRCO-R is:

```
DO 100 L=1,10
DO 100 I=1,43
X=1.0 + (I/500)-(I/750)
DO 100 J=1,1000
X=X+.001
100 Y=3.0*(X + 1.1)/((X + 2.2)*X)
```
 - AIRCO-S is:

```
DO 130 L=1,2
DO 130 I=1,5063
K=1
DO 130 J=1,100
IF (K) 130,112,114
112 IA(J)=J+1
K=1
GO TO 130
114 IA(J)=J
K=0
130 CONTINUE
```

ASEA

- FORTRAN
- Single precision
- Size: 22 Kb
- Matrix manipulation, light floating point

CAE1

- FORTRAN
 - Logical
 - Size: 20 Kb
 - Flight Simulation Benchmark — Most of the statements are logical variable manipulations, simulating the control of airplane control system switches and sensors.
- 528 A.S. - See Value chart*

REVISED MAY 1978

CAE2

- FORTRAN
- Single precision
- Size: 20 Kb
- Flight Simulation Benchmark — The program reproduces the main motion and motion output programs used in an aircraft simulator.

DG3

- FORTRAN
- Single precision
- Size: 28 Kb
- An "aggressive benchmark" by Data General with light single precision floating point and DO loops.

Double

- FORTRAN
- Double precision
- Size: 18 Kb
- Warden Integral Calculator with medium floating point content (double precision version of program SINGLE).

Egypt

- FORTRAN
- Single precision
- Size: 28 Kb
- Matrix Inversion Program — The program generates a 50 X 50 matrix, inverts it twice and checks the result. Round-off errors are reported.

FFT45

- FORTRAN
- Single precision
- Size: 20 Kb
- Fast Fourier Transform Subroutine and a main routine to call it. Uses some library calls. Medium floating point content.

Gauss

- FORTRAN
- Single precision
- Size: 20 Kb
- 10-point Gaussian quadrature numerical integration. Heavy single precision floating point content.

REVISED MAY 1978

Hanoi

- FORTRAN
- Integer
- Size: 18 Kb
- Calculates solution to "Towers of Hanoi" puzzle. Contains all integer arithmetic (add-subtract).

Hughes

- FORTRAN
- Single precision
- Size: 20 Kb
- Matrix manipulation program. Three 10 X 10 matrices are initialized and manipulated.

Jacobi

- FORTRAN
- Double precision
- Size: 186 Kb
- Performs a Jacobi diagonalization of a 100 x 100 double precision matrix of Eigenvectors. The matrix is first created, then diagonalized and later transferred to a second matrix and placed in ascending order. The matrix is checked by forming the product U^*U and tested to see if a unit matrix is obtained.

Littl

- FORTRAN
- Single precision
- Size: 18 Kb
- A little program which computes 100,000 times:

$$A = ((X-1.)/(X+1.))*X*.39 + A$$

The result is compared to the known correct answer as a precision check.

Lusty

- FORTRAN
- Single precision
- Size: 24 Kb
- Main routine and seven subroutines which perform calculations related to atomic particle motion and interactions.

Matrix

- FORTRAN
- Single precision
- Size: 22 Kb
- Matrix Inversion using SSP MINV routine. Program moves A(15,15) to B(15,15), inverts B, and forms product $A*BINV$. The program loops through these steps 10 times. Standard Gauss-Jordan method of matrix inversion is used by the MINV subroutine.

REVISED MAY 1978

Mflops

- FORTRAN
- Single precision
- Size: 67 Kb
- Program evaluates execution rates of FORTRAN DO Loops. All calculations are performed on 2- and 3-dimensional matrices. Fourteen calculations are performed and timed. The resulting time metric is a "time per loop."

NEFF

- FORTRAN
- Double precision
- Size: 20 Kb
- Nuclear equation calculations. The program makes heavy use of trigonometric functions. The program performs an iterative calculation of the effective number of nucleons in a complex nucleus with absorption.

NSC1

- FORTRAN
- Single precision
- Size: 24 Kb
- Three part test of arithmetic capabilities:
 - NSC1A calculates 10,000 times:
 $ABS(SIN + X^{**}(ALOG) + EXP(-ATAN))$
 - NSC1B calculates 100,000 times:
 $(X-1)/(X+1) + X*.39 + A$
 - NSC1C sorts a 1,000 element R*4 array.

NSC2

- FORTRAN
- Mixed
- Size: 22 Kb
- Two-part test of the local and global optimization capabilities of the FORTRAN compiler.
 - NSC2A calculates 327,660 times:
 $C = 2.2*BUF(1000)/113.3 + (999.-BUF(500))$
 - NSC2B calculates 327,660 times:
INTEG = 16^*4
ID = $256/4$
ITEMP = $16*IT/2 + (8-3)$

Philco

- FORTRAN
- Double precision
- Size: 25 Kb
- Matrix inversion program

REVISED MAY 1978

Prime

- FORTRAN
- Integer
- Size: 24 Kb
- Prime number calculation program

RAV1

- FORTRAN
- Double precision and single precision
- Size: 34 Kb
- Program does some geometric calculations including heavy use of trigonometric functions.

RAV2 and RAV3

- FORTRAN
- Integer
- Size: 18 Kb
- Performs 800,000 integer arithmetic operations (add, multiply, and divide).
 - RAV2 uses the machine default integer size (I*4 on the VAX-11/780).
 - RAV3 is identical to RAV2 except that Integer*4 is specified in the program.

RAV4

- FORTRAN
- Single precision
- Size: 31 Kb
- Random program that does real, integer, and logical calculations on scalars and array elements.

RAV5

- FORTRAN
- Double precision
- Size: 45 Kb
- Double precision version of RAV4

RAV6

- FORTRAN
- Single precision
- Size: 18 Kb
- Calculates 10,000 times:
 $A = A + \text{SQRT}(\text{ABS}(\text{SIN}(X))) + X^{**}(\text{IFIX}(\text{ALOG}(X)*0.43429448)) + \text{EXP}(-\text{ATAN}(X/666))$

REVISED MAY 1978

RAV7

- FORTRAN
- Double precision
- Size: 18 Kb
- Calculates 1000 times:
 - $X=1$
 - $Y=2.D0*PI/X$
 - $Y=1.D0/X$
 - $X=DSQRT(DABS(DSIN(X) + DCOS(X) + DATAN(Y)))$

RAV8

- FORTRAN
- Integer
- Size: 44 Kb
- Manipulates eight 1605 element arrays to perform a histogram calculation, accumulating results in a 200 element array.

RR1-RR4

- FORTRAN
- Single precision
- Size: 19-35 Kb
- Series of light to medium single precision floating point bench-marks which prompted the development of the FP11-C and FORTRAN IV-PLUS Version 2 on the PDP-11.

SAAB

- FORTRAN
- Mixed
- Size: 23 Kb
- A four-part test program:
 - SAAB1 — Euler angles transformation between aircraft coordinate system and earth-bound system. Transformation of three angles is done 49130 times. Most of program is SIN, COS, or multiplication.
 - SAAB2 — Simple 5 degrees of freedom aircraft model which is looped through 40,000 times
 - SAAB3 — Interpolation routine which uses a slowly changing variable. 401,000 values are fetched from the routine.
 - SAAB4 — Packing and unpacking of bit flags from a logical variable. A series of AND, OR, NOT operations is repeated 50,000 times. All integer arithmetic.

Scrip

- FORTRAN
- Double precision — FFT
- Size: 282 Kb
- Performs a Fast Fourier Transform ~~twice~~ on a 16,384 element double precision matrix. The root mean square of the result is then computed.

forward and backward of a real ramp. The data consists of

REVISED MAY 1978

Scrip2

- FORTRAN
- Single precision — FFT
- Size: 151 Kb
- Single precision version of Scrip program

Scrip3

- FORTRAN
- Complex — FFT
- Size: 282 Kb
- Complex variable, single precision version of SCRIP program

Single

- FORTRAN
- Double precision
- Size: 18 Kb
- Warden integral calculation with medium floating point content. (Single precision version of the program "Double.")

SP1111

- FORTRAN
- Double precision
- Size: 81 Kb
- An inner loop subroutine from a quantum mechanics application package which solves the Schrodinger Equation for atoms and molecules. SP1111 is the inner loop of a 6-dimension Gaussian integral. It has been produced by a code generator which produces mostly in-line code. DO loops have been removed and replaced by a sequence of separate FORTRAN statements. The subroutine is executed 5000 times.

Stelos

- FORTRAN
- Single precision, heavy trigonometric usage
- Size: 21 Kb
- Astrophysics routine which performs a calculation using many trig functions which produce "Heliocentric distances" as the resulting answers. The answers are stored in a print file.
- Stelos-S and Stelos-D are single and double precision versions of the same program.

Valleau

- FORTRAN
- Single precision
- Size: 70 Kb
- A type of atomic particle physics calculation.

REVISED MAY 1978

Whetstone

- FORTRAN
- Single and double precision versions
- Size: 21 Kb
- The Whetstone program consists of 10 modules, each of which exercises a group of language features. Each module is placed in a loop and the number of times it is executed is adjusted to mimic as closely as possible the statistical profiles of language feature usage (as measured by the British Central Computer Agency). All the loops have been arranged so that an optimizing compiler cannot remove a significant amount of code from them.

Features exercised include simple variable and array addressing, fixed and floating point arithmetic, subroutine calls and parameter passing and standard mathematical functions.

Speed is calculated by executing the program twice using a difference of 10 in the number of loop counts executed and noting the difference in run times. If this time is "T" seconds, then the speed of the machine is expressed as $1000/T$ thousands of Whetstone instructions per second. One Whetstone instruction is approximately equal to two machine instructions.

Subject

Page

Page
ANS to

VAX/VMS Performance and Configuration

Hank Levy
Advanced-11 Engineering
August 15, 1978

COA...

...

...

5

6

42

90

80

200

VAX/VMS Performance and Configuration

Objectives

Our objectives in performing the VMS multi-user measurements were: (1) to exercise VMS under various configurations and loads in order to better understand its response to workload increases, (2) to gain insights into the selection of parameters for generating the VAX/VMS system (3) to feedback performance information to developers, and (4) to provide marketing and configuration data to sales and support personnel.

Measurement Technique

To measure VMS, we used a PDP-11/34 as a remote terminal emulator (RTE). Each DZ11 on the 11/34 was connected to a DZ11 line on VAX through a null modem interface run at 600 baud in each direction. Characters output by one machine appeared as terminal input on the other.

During a measurement session, a script of user commands was associated with each terminal line. The script included all commands to be sent to the VAX system, as well as user-think time distributions, expected responses and response times. In this way, the RTE system generated timesharing "user" inputs and measured the response times to "user" requests.

Definitions

Since measurement of each data point may take an hour or more of stand-alone time, mapping the performance of many configurations is a time consuming and costly operation. Therefore, we measured only one workload, a Fortran development timesharing shop, across several VAX configurations. The three scripts described below represent the kinds of user activity involved in the workload, for example, creating and updating Fortran source programs, performing file manipulations, and compiling, linking and running Fortran programs. Each description is followed by the percentage of users that executed the script in the final workload.

1. New file creation script. User creates a 150 line Fortran program with the SOS editor. Files are then purged, and the procedure is repeated.

Think times are exponential with a 7-second mean. (Since user typing speed was 600 baud, or almost instantaneous, this think time includes the simulated user typing time). Forty percent of the users in the workload execute this script.

2. Update file script. User copies a file from another directory, purges files, performs file editing commands with the SOS editor, and then repeats the procedure. Think times are exponential with a 10-second mean. Forty percent of the users in the workload execute this script.
3. Fortran computational script. User compiles, links, and runs the several hundred line Buchholz Fortran benchmark program. About 80 percent compute bound, this program runs for approximately 50 seconds on a stand-alone system. Think time is 3 seconds between commands. Twenty percent of the users in the workload execute this script.

Each system was loaded with copies of these scripts in the ratios given. For example, on a 20 user system, 8 users were creating new files, 8 were updating old ones, and 4 were compiling, linking, or running the program. We believe this to be a fairly representative development mixture. Note that this is a continuous load; all users are always active. Not only do they type at 600 baud, but they never answer telephones or walk down the hall for a donut and coffee.

Results

The following graphs show the "stretch factor" for various user commands. The stretch factor is a ratio of the response time under loading (that is, when there are many users) to the response time for that request on a stand-alone, one megabyte 2-RP06 system. For example, an insert command with SOS has a stand-alone response time of .3 seconds. If the measured response time for that request with 20 users on the system were .9 seconds, then the stretch factor for that request would be 3.

The graph in Figure 1 shows the stretch factors for the smallest available configuration, a VAX system with 2 RK07 disks, a console, and one DZ11, under a purely editing workload. Also shown is the amount of swapping at each point. The system was loaded with equal numbers of file creation and file update scripts. Stretch factors are shown for two types of editing commands, "insert" and "find". The SOS editors were run in 50 page working sets, and about 20 of those pages were

shareable. From the graph it appears that 8 to 10 small, highly interactive editors could be supported on this configuration.

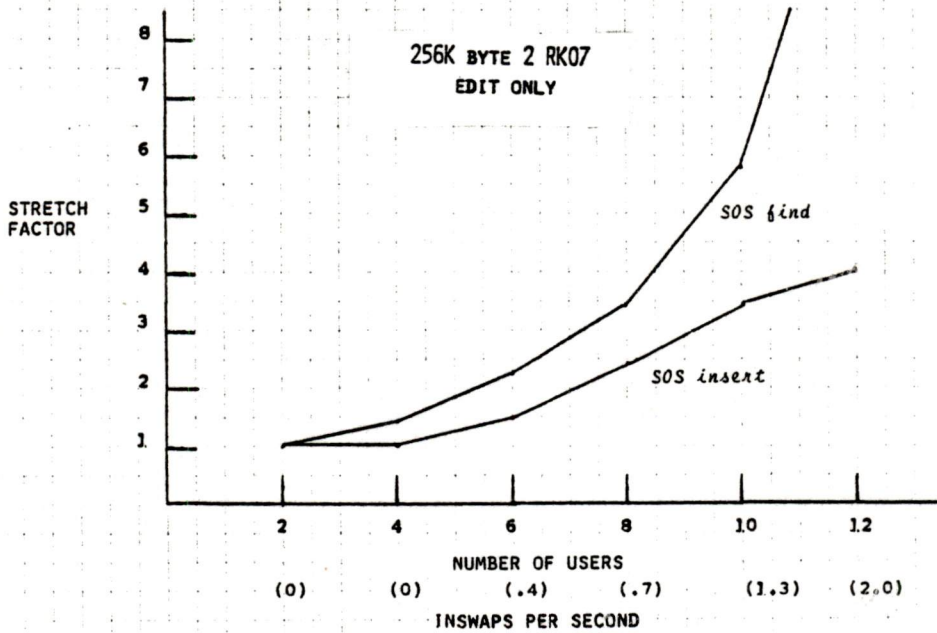


Figure 1

Figure 2 again assumes work on the smallest configuration. It shows stretch factors for up to 7 users, with one user running the Fortran compile, link,

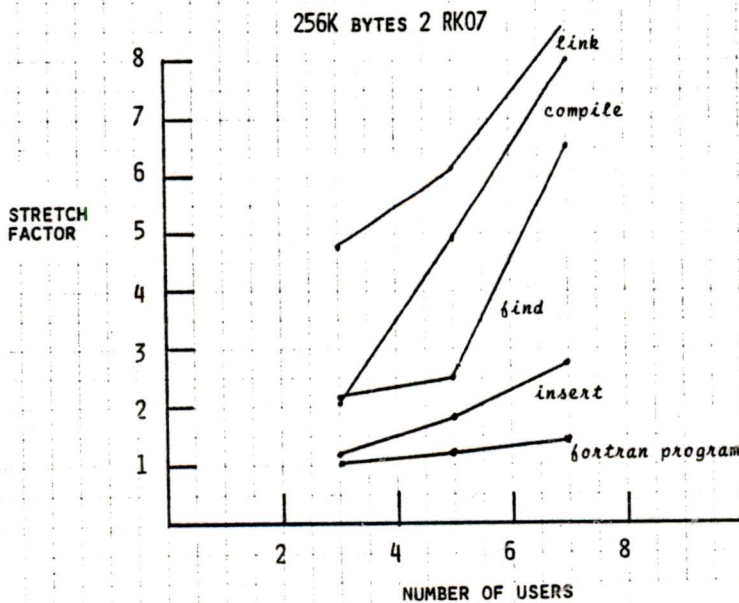


Figure 2

and execute script, and the remaining users editing.

The 256K system has about 320 pages available for process use out of a total of 512 pages. It will not perform well if several users attempt to run large programs simultaneously. Instead of benefiting from the normal I/O overlap in timesharing systems, this configuration is so low on memory that two large programs may not be able to exist in memory together. When this situation occurs, the system will be forced to swap each program out after its quantum expires in order to load the other, causing high degradation. One solution is to create a single batch stream and request users to submit large programs to batch. Serializing the execution of large programs will improve not only interactive response time, but also the response time to large programs even though that response time may include some time spent in the queue.

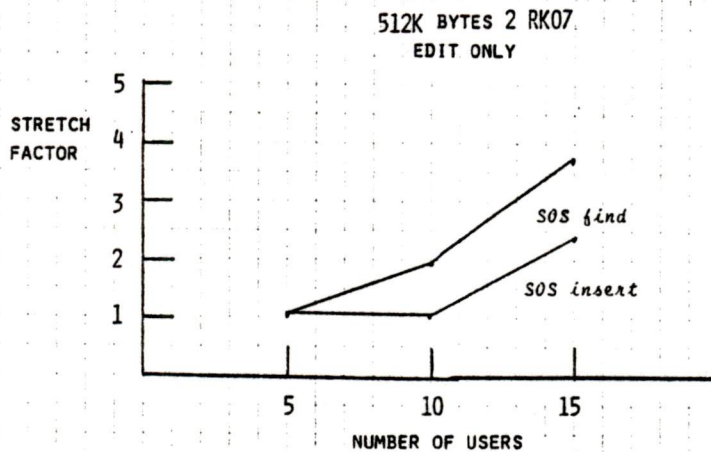


Figure 3

Figure 3 shows the response of a 512K 2 RK07 system to an editing only workload. This system has about 800 pages available for user processes, more than 2 1/2 times that available in the 256K system. In Figure 4, we see this system under the simulated development workload. Although the compile and link degrade by the time 15 users are added, notice, in comparison with Figure 3, that the interactive users still receive essentially the same response as without the added computational load.

Some measurements were performed comparing the 2 RK07 system with a 1 RP06 system in 512K bytes of memory. It was found that the 2 RK07 system provided marginally better response. We would expect the 2 RM03 system to

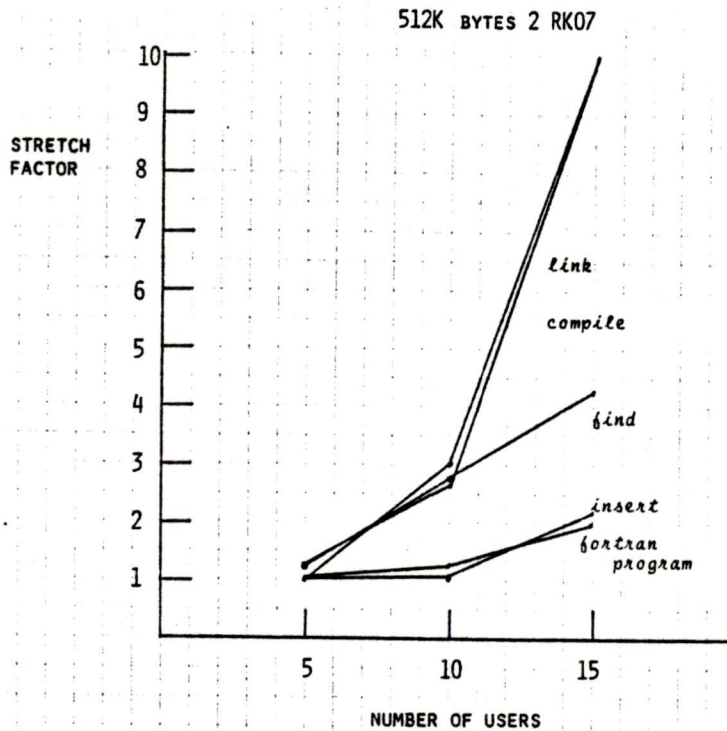


Figure 4

be better than either of the above, but RM03s have not been available to test.

Figure 5 shows the 1 megabyte 2 RP06 system with 5 DZ11s and LP11 line printer, which contains 1670 pages for user processes. We believe this system to be an excellent timesharing performer for up to 35 users.

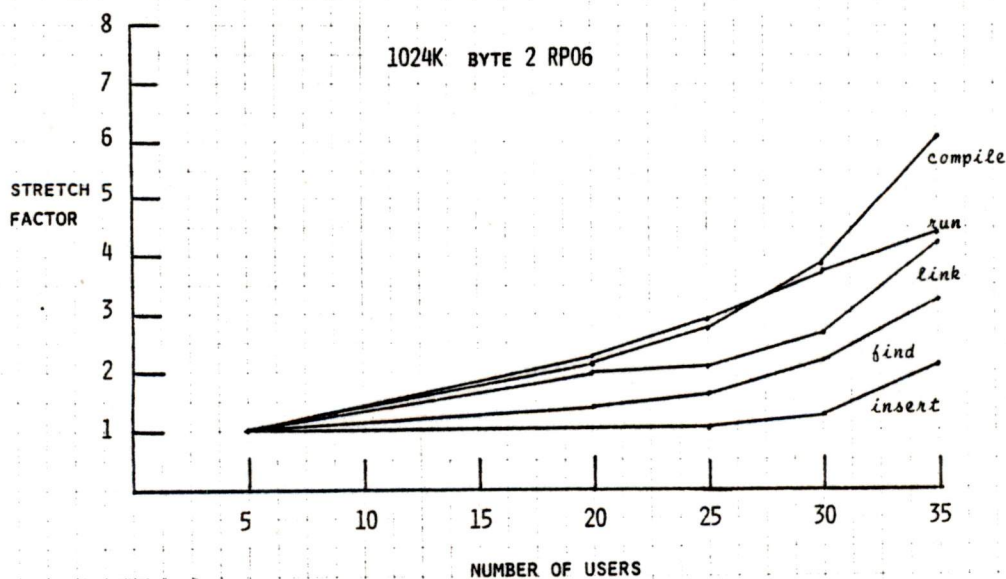


Figure 5

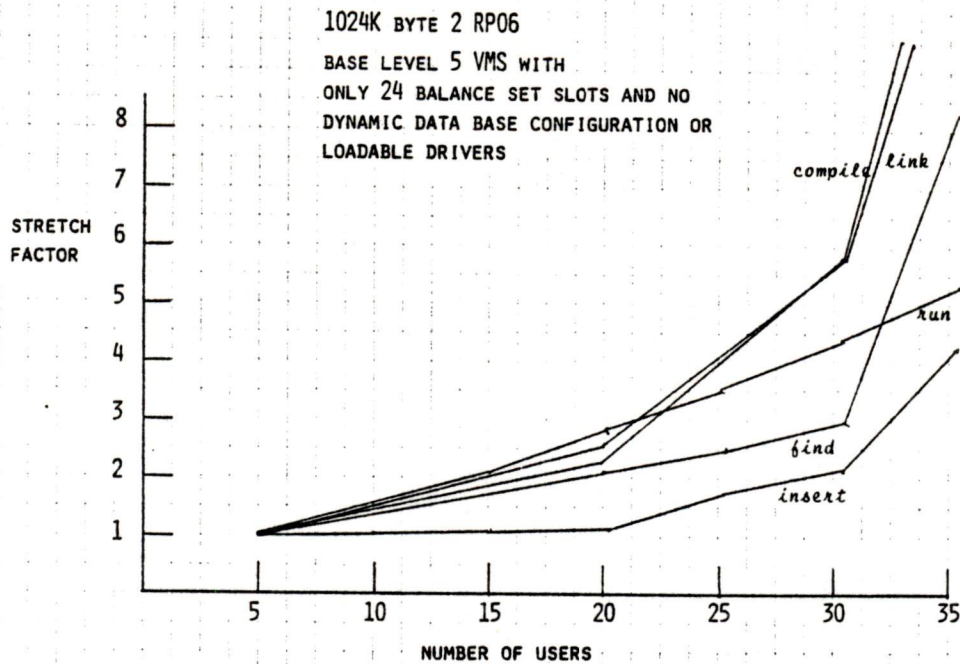


Figure 6

- * Working Set Maximum (WSMAX) - Besides contributing to the process header, the maximum working set determines the amount of swapfile space needed. The swapfile is allocated in segments of size WSMAX. Check to see that the default swap file allows sufficient space for the number of processes you wish to run. The swap file is just a large contiguous file, and you may create one with COPY/CONTIGUOUS.
- * Non-Paged Dynamic Memory (NPAGEDYN) - This parameter controls the size of the non-paged dynamic memory pool. The cost is 516 bytes per page (512 for the page, plus 4 bytes of system page table). When the system is booted, device drivers and device data bases are loaded into the pool. The preallocated list of I/O request packets (IRPCNT) are taken at 80 bytes each from the initial allotment of NPAGEDYN. During system operation, most dynamic data structures and buffers are also taken from the pool. The best way to estimate this requirement is to boot the system and see how much of the allocated space remains using the Display program. It's a good idea to leave about 1 page of pool per process. The cost of non-paged pool is high, so estimate carefully on the low end. Don't be too close to the limit, or the system may hang if it runs out of dynamic memory. On a fully loaded system the pool should have a few thousand bytes left. Just

logging in a user takes 408 bytes of pool and 3 IRPs.

- * I/O Request Packet Count (IRPCNT) - This controls the number of I/O request packets preallocated from the non-paged pool. Allocate about 5 or 6 packets per user. Watch with the Display program to see if the IRP list becomes exhausted when the system is heavily loaded. If so, increase the count. If the list is empty, further requests for packets are filled from the non-paged pool, causing fragmentation.
- * Modified Page Writer (MPW) - There are three parameters that control the allocation of the modified page list and the functioning of the modified page writer:
 - (1) MPW_HILIM - the maximum list size,
 - (2) MPW_LOLIM - the minimum list size, and
 - (3) MPW_WRTCLUSTER - the write cluster size.

The modified page list acts as a cache for "dirty" pages faulted from process working sets on their way to backing store. If a process faults a page that is on the list, it gets a 200us page fault. When the list HILIMIT is reached, the modified page writer begins to write pages to backing store attempting to output contiguous blocks of WRTCLUSTER pages at a time until the list is below LOLIMIT. The parameters should be set so that the list is not emptied completely by the last write. For example, if HILIM, LOLIM, and WRTCLUSTER are 100, 25, and 40, respectively, then the modified page writer will stop with 20 pages left after outputting the second cluster. The cost for the modified page list is HILIM pages. Use the Display program PAGE display to see how many pages are being faulted back from the modified list. If the STATES display shows a high percentage of users in page fault wait (PFW), try increasing the size of the modified- and free-page lists.

- * Free Page Limit (FREELIM) - The free page list is a cache for read-only pages faulted from process working sets. FREELIM is the lower limit for this list and the fixed overhead in pages. The default is 10 pages.

Swapping versus Paging

The "swap ratio" is the ratio of processes which are swapped out to the number resident in physical memory at

Terminal Driver Character Throughput

In an associated measurement, a program was written to drive as many terminals as possible at 9600 baud to determine the rate at which VMS could output characters over DZ11 lines. The program issued QIOs of 1000 byte buffers with the NOFORMAT bit set to reduce character handling overhead. VMS was able to output over 10,000 characters per second, which translates to less than 100us per character.

System Generation and Tuning

The selection of initial parameters for system generation can have a major effect on performance. No hard and fast rules exist for selecting parameters, since each customer's workload and priorities differ. There are general guidelines, however the most important parameters are those that affect the most critical resource -- physical memory. For maximum system performance, as much physical memory must be saved for user processes as is possible without sacrificing system operations or causing bottlenecks. Experimentation and observation are required to arrive at the best values for your configuration and workload.

The parameters described below have the greatest effect on available physical memory.

- * Balance Set Count (BALSETCNT) - This parameter controls the number of balance set slots allocated for process headers. It is the maximum number of processes which can be simultaneously resident in memory, sometimes known as the multi-programming limit. Setting this number too low could cause swapping to occur, even though enough physical memory is available to keep extra processes resident. Compare Figure 6, a base level 5 VMS system with only 24 balance set slots, with Figure 5 which has 35 balance slots. You can clearly see the effect of forced swapping which starts at 20 users.

The cost of a balance slot is fairly high and depends on the maximum process working set size (WSMAX) and the maximum process virtual address space (VIRTUALPAGECNT). The cost is 1 longword for each 128 pages of virtual address space (65K bytes) plus 1 longword for each 128 pages of working set size plus 1 page for the rest of the process header. For a system with a 512 page working set maximum and a 2 megabyte virtual address space (4096 pages), the cost is about 650 bytes for each slot.

any one time. Acceptable system performance is largely determined by the swap frequency -- the number of processes swapped in and out of memory per second -- rather than the swap ratio itself. On the heavily swapping system, an I/O bottleneck of the system disk causes performance degradation. Consider that a VAX process of a hundred pages may take 100 or more milliseconds to swap. A process with a 1000 page working set may take more than a full second of disk time to transfer!

The Display program will show the number of inswaps per second on the system. Multiply this number by 2 to get the number of actual swap transfers, since there is normally an outswap for each inswap. A system which shows 3 inswaps per second on RP or RM disks, or 2 inswaps per second on RK disks, over a long period of time is probably too heavily loaded. Figure 1 shows the number of inswaps per second and the associated performance degradation on a 256K byte system.

Once the system is properly configured, one must decide how to divide the available physical memory among the user processes. The system manager controls this by the setting of working set quotas in the authorization record of each user and the setting of the maximum allowable working set WSMAX. However, within the allowable working set limits, large performance differences can be achieved by choosing correct working set sizes for different tasks. Reducing working set sizes allows us to keep more processes resident, but by reducing the working set size beyond a certain point, we begin to trade paging for swapping. The Display program will show the paging I/O rate (READ I/Os plus WRITE I/Os) as well as the inswap rate to give you some idea of where most of the load is coming from.

For the scripts run in the multiuser tests, all SOS editing tasks were performed in 50-page working sets. Experimentation showed that the response time achieved with 50 pages was better than that achieved with either 40- or 60-page working sets. The Fortran benchmark was run in a 90-page working set, and the compile and link steps used 200 pages on the megabyte system and 150 pages on the smaller ones. In general, it is more efficient to keep as many processes in memory as possible, which not only reduces swapping, but increases the probability that the CPU will be able to find an executable process to service. Although there is a point of diminishing returns when reducing working sets, we generally believe that paging is better than swapping because the modified- and free-page lists act as page caches to reduce the number of disk I/Os that are needed.

Effect of Large Programs

We included one experiment to determine the effect of running a large, compute-bound, heavily paging program on the response to timesharing users. A 15-user timesharing system in 1 megabyte was measured with and without a large Fortran matrix inversion program. Run in a 256 page working set, the matrix inversion inverted a 500 by 500 matrix using more than two megabytes of virtual space for program and data. The performance of the inversion program was not measured. Table 1 shows stretch factors for several commands with and without the inversion program (INV).

Command Type	Stretch Factor Without INV	Stretch Factor With INV
Insert	0.98	1.86
Find	1.58	3.26
Compile	1.71	3.68
Link	1.65	4.89

Table 1

Note that the INV program caused some degradation in the user response times, but we believe the results to be quite good considering the size and nature of the benchmark.

Multiple ACPs

When mounting public disks, it is possible to specify a separate ACP process (/process=unique) to control file system accesses to that volume. The ACP comes into play only on requests which require file system intervention, e.g., space allocations, directory handling, or I/Os which cause a window miss. Since a single ACP handling several volumes serializes the requests, under certain workloads more overlap may be obtained with multiple ACPs. In the few experiments performed so far with RTE, the second ACP has not helped performance, probably because the memory requirement is more valuable than any additional overlap. Multiple ACPs may improve performance on large memory and disk configurations with lots of user I/O spread across several volumes, but not on the 256K or 512K systems.

Conclusions

Benchmarks have already demonstrated the speed of the

VAX processor and VMS and its associated software. In the multiuser timesharing environment, disks and memory are much more critical than raw CPU speed. VMS is a sophisticated system with many performance enhancing features, most of which require physical memory to function properly. VMS can be a great performer in the multiuser world if configurations are selected and adjusted with a knowledge of the customer's workload. The 256K byte system in particular requires some amount of tuning to reach optimal performance. On the megabyte system, be generous with parameter settings for buffers, caches, etc. to allow the system to function with all the performance optimizations. Although we've only been able to test 35 users due to RTE hardware limitations, we have every confidence that VMS will support many more users with the addition of more memory.

The following table summarizes our recommendations for the minimum configurations that will support the number of users shown. We consider the RM03 and RP06 disks to be interchangeable in the configurations, and the smaller systems could use one RP06 or RM03 in place of two RK07s.

number of users	configuration	
	memory	disks
1-7	256K	2 RK07s
8-15	512K	2 RK07s
15-20	512K	2 RM03s
20-35	1024K	2 RP06s
35-48	1536K	2 RP06s
48-64	2048K	2 RP06s

Table 2



For Further Information:
Richard O. Berube
(617) 897-5111 Ext. 3046

VAX11 780

NEWS RELEASE

DIGITAL EQUIPMENT CORPORATION UNVEILS ITS
FIRST 32-BIT COMPUTER SYSTEM: VAX-11/780

Called by Ken Olsen "probably the most significant interactive computer of the last decade...a milestone equal to the original PDP-11..."

BOSTON, Mass.--October 25, 1977--Digital Equipment Corporation introduced a new 32-bit computer system today which combines the full power and performance of conventional large mainframes with the interactive strength, flexibility and low cost of a minicomputer.

Called the interactive VAX-11/780, the new multi-user system is an extension of Digital's PDP-11 family and is said to be the industry's fastest computer system priced below \$200,000. It features a new virtual memory operating system (VAX/VMS) which provides multi-users a direct addressing capability of over four billion bytes!

Prices for the VAX-11/780 system start at \$130,000 for a minimum configuration. Deliveries are scheduled to begin in early 1978.

-more-

In announcing the new system to 400 shareholders and 100 members of the press at the company's Annual Meeting here, Digital President Kenneth H. Olsen called it "probably the most significant interactive computer of the last decade. Indeed, we think it is a milestone equal to the original PDP-11 in terms of the long-range impact it will have on the way people use computers."

In addition to its extensive interactive timesharing capabilities -- 64 separate users can work with the system simultaneously -- the VAX-11/780 also has impressive multi-user batch processing capabilities, according to Olsen. "We designed the 11/780 to be a general purpose machine which can handle a wide variety of applications in science, industry, education, and in the commercial world. And we think it will be popular with our OEMs, as well," he said.

"The best of what we've learned about interactive computers in our first twenty years has gone into this machine," he continued. "We have spent over 300 man years of intensive engineering effort in its development, and during that time I have sensed more excitement and enthusiasm among the developers of VAX than I remember seeing at any other time in Digital's short history."

Olsen described the VAX-11/780 system as an upward extension of Digital's popular PDP-11 family. "And yet, it is an entirely new and exciting computer in its own right. We have gone to great lengths to design a system that will be useful to the largest possible number of users, starting with those who use the 50,000 PDP-11s we have already installed worldwide and the thousands more we expect to sell in the future.

"Our PDP-11 customers have invested enormous resources in the form of trained personnel, peripheral devices and data bases, all built up to support PDP-11 systems. To the extent that we found it possible, we wanted to be certain that these resources would be compatible with our new VAX system," he said. "As a result, the 11/780, with its 32-bit wordlength and comprehensive operating system software, provides a natural upward migration for PDP-11 users whose applications require additional address space and functionality.

Olsen noted that Digital's PDP-11 family has the widest scope of any family of computers available today. "Starting with our LSI-11 microcomputer at the low end, the PDP-11 family forms a continuous spectrum of full system functionality which now ranges up to the new VAX-11/780," he said.

#####



For Further Information:
Richard O. Berube
(617) 897-5111 Ext. 3046

**VAX11
780**
NEWS RELEASE

DIGITAL'S NEW VAX-11/780 SYSTEM
TARGETED FOR DIVERSE MARKETS

BOSTON, Mass.--October 25, 1977--The 32-bit VAX-11/780 System introduced today by Digital Equipment Corporation will have widespread use among end users and OEMs in industrial, commercial, scientific and educational environments. It is aimed at satisfying the need for extended wordlength in many data acquisition and process control applications. Its unique hardware and software architecture are designed to maximize its performance of interactive, time-critical and computational tasks.

Here is what the managers of some of Digital's major market areas have to say about the VAX-11/780 system:

EDWARD A. KRAMER, VICE PRESIDENT, LABORATORY AND MEDICAL PRODUCTS GROUPS:

"Government, university, and private research laboratories represent a major market for the VAX-11/780. Many scientific applications require two main capabilities in a computer system: very fast FORTRAN and the ability to handle extremely large programs. Crystallography and molecular research are just two fields where the ability to manipulate large data arrays easily is absolutely essential. The VAX-11/780 system provides these capabilities -- and more -- at a price which will make it very attractive to our customers. It is easy to use, enabling the

-more-

scientist to get his application up and running quickly. It can be connected to other PDP-11 systems and employed as a central development system in a laboratory network. The VAX-11/780's high-performance FORTRAN and floating-point accelerator option enable it to outperform any other machine in its class -- and several selling for twice the price.

"The VAX-11/780 continues Digital's leadership in scientific computing."

WILLIAM H. LONG, VICE PRESIDENT, OEM: "The new system's capabilities for rapid response and high total throughput are of significant value to original equipment manufacturers for such tasks as aircraft simulation, power monitoring and commercial processing.

"The VAX-11/780 is especially strong for time-critical applications," Long said. "Aircraft simulation, a growing market for OEMs, requires the computer system to supply all aircraft responses to pilot action in a flight simulator, using manufacturer-supplied data on several hundred elements of flight behavior and systems performance. The extended wordlength and raw speed of the VAX-11/780 are well suited to manipulating such large amounts of data in real time.

"For commercial OEM applications, the VAX-11/780's optimized design for multi-user interaction is well suited for high-volume transaction processing, while virtual memory features simplify writing and processing of large programs for inventory, bills of materials, material requirements planning and database management.

"In power monitoring, the VAX-11/780 can function as an upper level system in a hierarchical network monitoring power distribution for electric utilities," Long continued. "In this position, it would gather information from smaller systems at many distribution points, calculate load flows and compile energy distribution and system status reports for engineering and management use."

JULIUS MARCUS, VICE PRESIDENT, INFORMATION SYSTEMS: "The VAX-11/780 is a major technical achievement and milestone in computer systems design. It exemplifies our philosophy of product compatibility and provides an architectural extension to the high end of our PDP-11 family of systems. We see initial applications in our telephone industry and government markets and selected high-performance, data communications-intensive applications in the banking and transportation markets.

"Because of the critical nature of the intended applications, we placed much emphasis on development of the Reliability And Maintainability Program (RAMP) during system design. This feature enables both local and remote diagnostic analysis to expedite maintenance and repair. The expected result is substantially higher than average uptime for systems of this size.

"The initial software offering on the VAX-11/780 is extensive, providing operating systems, languages, and file systems. Major development programs are currently underway for extension of this software capability to include additional commercial software, which will round out the VAX-11/780's overall strength and applicability to all our markets involving general-purpose computing, EDP and transaction processing."

JERRY WITMORE, PRODUCT LINE MANAGER, EDUCATION PRODUCTS GROUP: "Colleges and universities are primary prospects for the VAX-11/780 educational installations. We see this system assuming a prominent role in university computation centers as the principal timesharing facility for student use. In addition, it would handle batch-oriented tasks in FORTRAN-IV-PLUS. In this way the VAX-11/780 could increase both the quantity and quality of total computer services, through its ability to serve a large number of interactive users, and its cost advantages when compared with the expense of enlarging a central mainframe. For college data processing centers the VAX-11/780 could act as the primary facility. It would

Digital Equipment Corporation
Digital's New VAX-11/780 System
Targeted For Diverse Markets
Page Four

satisfy requirements for high-performance, general purpose timesharing, executing large FORTRAN programs, and running high-volume COBOL tasks for administrative processing and program development."

#####

OPENING REMARKS

Before I start diving into some of the more technical characteristics of the VAX-11/780 system I would like to make a few general comments.

1. We, as a company have invested a great amount of effort and resources in this program. Over 300 man years of effort. The industry is still technology driven. DIGITAL's customers have come to expect certain things from us; products with either a higher level of functionality and performance at the same price as existing products or new products with the same functionality at lower cost.
2. The VAX-11/780 system is not the average 32 bit computer. The system architecture, hardware and software is the result of a careful and lengthy design process. Just as for the PDP-8, the DEC 10/20 and the PDP-11, the VAX-11 architecture has to withstand the years; it must be adaptable to many user environments, some of which are not ever foreseen today. Our customers depend on this type of stability and flexibility.
3. We believe that the new VAX-11/780 is a milestone in the field of interactive computers because we have combined in one system LARGE COMPUTER FUNCTIONALITY AND PERFORMANCE, RELIABILITY, MAINTAINABILITY, AVAILABILITY AND COMPATIBILITY WITH the most popular 16 bit computer, the PDP-11.
4. We have done this by taking a fresh look at total system design. In one sense we started from scratch, but in an other sense we started with the experience of 100,000 computers behind us.

A SYSTEM APPROACH

We started designing or perhaps more precisely architecting the machine from scratch with a team of hardware and software engineers working together right from the beginning.

We have made trade offs on a system wide basis.

The most obvious manifestation of this ground up, integrated design effort is reflected in the instruction set of the machine. I took two examples out of many to illustrate how hardware and software influenced each other.

ACBL, (ADD, COMPARE and BRANCH) translates exactly into one machine instruction, higher level languages constructs such as a FORTRAN DO LOOP. The PDP-11/70 which has one of the fastest FORTRAN compiler in the 16 bit computer world takes 7 instructions to translate the same DO LOOP. The result is more compact and faster compiler generated code.

INSQUE and its counterpart REMQUE lets the operating system scheduler insert or remove an entry in a doubly linked queue in one single instruction. RSX-11M an industry leader and our fastest REAL TIME MULTIPROGRAMMING operating system, takes 8 instructions to accomplish the same function. The result is faster program scheduling and less operating system overhead.

FUNCTIONALITY

The functionality of a computer system is often characterized by its operating system and associated software.

We have developed VAX/VMS, the VAX-11/780 operating system as a Single general purpose operating system to satisfy a broad range of functions.

For instance VAX/VMS supports

TIME CRITICAL APPLICATIONS
64 INTERACTIVE USERS
MULTIPLE BATCH STREAMS
MULTIPLE LANGUAGES

We have applied mainframe software technology to VAX-11/780. VAX/VMS is a virtual memory operating system; it allows programs larger than physical memory to run in a fashion transparent to application programmers. I have been asked very often what is virtual memory? VIRTUAL MEMORY is memory that is not there, but the programmer does not know it! We have solved the inherent addressing limitations of the 16 bit computer by providing a very large addressing space of 4 BILLION Bytes. As a point of comparison a 16 bit architecture provides only 64K bytes or 128K bytes of addressing space.

We have built a hardware engine with a new architecture and 32 bits everywhere.

The machine is articulated around a high speed (13.3Mb/Sec) synchronous bus (a backplane in reality); some of the key features are

- 32 BIT INTERNAL BUS
- 32 BIT ARITHMETIC AND DATA PATH
- 243 BASIC INSTRUCTIONS
- 9 FUNDAMENTAL ADDRESSING MODES
- FLOATING POINT INSTRUCTIONS
- PACKED DECIMAL AND STRING INSTRUCTIONS
- PAGING WITH 4 HIERARCHICAL PROTECTION MODES
- 16 32 BIT REGISTERS
- 2Mb OF ECC MOS MEMORY (THIS IS REAL MEMORY NOW; IT'S THERE)
- UP TO 32 DISK DRIVES WITH 176Mb OF STORAGE EACH
- 800/1600 BPI TAPE
- CARD READER, LINE PRINTERS, TERMINALS

PERFORMANCE

The VAX-11/780 system can perform many functions as we have just seen. It can perform these functions very fast.

Let's review a few of the factors which contribute to performance.

- A very fast internal bus
- Multiple caches for data, address translation, I/O buffering and operating system data bases
- A powerful instruction set
- A controllable and tunable paging scheme which allows time critical program to be swapped entirely (rather than paged) or locked in memory
- A scheduler with fixed and system optimized priorities
- A highly optimized FORTRAN compiler and a very fast optional floating point accelerator
(1.4 usec for double precision ADD).
- The VAX-11/780 is positioned at the high end of the PDP-11 family in terms of performance and price, and below the DEC 2050 in terms of both price and performance.
- On average, 32 bit program execution and system throughput is roughly twice that of a comparably configured PDP-11/70.

RELIABILITY, AVAILABILITY, MAINTAINABILITY

VAX-11/780 serves a broad range of function, is fast and by design is RELIABLE, AVAILABLE, MAINTAINABLE.

Reliability, availability and maintainability features are found in the hardware architecture, the software architecture, the individual components and board design and in the packaging.

The objective: Keep the system running

If it fails find the fault quickly, fix it, get the machine up and running again

Protect the data

The list of features is long and impressive

Parity on buses, data path, control store, caches

ECC on memory

Consistency checks in hardware and software

History of bus activity

LSI-11 microcomputer for console operation, local and remote diagnostics

Floppy diskette for microdiagnostics loading and software updates distribution

Packaging with fixed back plane, cable troughs, modular power supplies, air flow and temperature sensors. On line diagnostics.

Error logging

We included these features by design. They cost money.

We built them into the system, not because it is FUN, but because our customers expect it. This is the business we are in. It's just inappropriate for our machine to go down or stay down very long.

COMPATIBLE WITH THE PDP-11

VAX-11/780 is a new 32 bit machine; VAX/VMS is a new Virtual Memory Operating System.

By design the new system is enormously compatible with the other PDP-11's.

It is compatible where it counts: PEOPLE, DATA, USER PROGRAMS. Compatibility is designed in, from the innermost to the outermost layers.

The new instruction set is not bit for bit compatible with the PDP-11's because we wanted performance and efficiency but both the PDP-11 and VAX-11 are

byte addressable machines
stack oriented
they have the same data types (VAX-11 has more)
the same instruction mnemonics
the same UNIBUS and MASSBUSES
the same PERIPHERALS

VAX-11 includes a 16 bit instruction set in its compatibility mode. The PDP-11/70 instructions are there with the exception of some privileged instructions such as HALT, I/O RESET etc.. Such instructions are not normally used by application programmers. The software or outer layer of the system has the same high degree of compatibility with the PDP-11 as the hardware.

- The ON DISK STRUCTURE is the same as RSX-11 and IAS.
(VAX/VMS also has implemented extensions to it for more performance)
- The file access methods RMS are the same as RSX-11, IAS, RSTS/E
- The command languages DCL and MCR are the same as the ones found in IAS, RT-11 and RSX-11M
- The higher level languages are source compatible with their PDP-11's counterparts.
- The RSX-11M Application Migration Executive which exploits the 16 bit compatibility mode instruction set runs concurrently with other jobs under the control of VAX/VMS.
- The Application Migration Executive allows non privileged RSX-11M Tasks to execute on the VAX-11/780 with little or no modification.

We are taking advantage of this feature, ourselves - extensively. COBOL-11 and BASIC+2 as well as many utilities (perhaps as many as 200,000 lines of code) execute in compatibility mode and generate PDP-11 code.

- VAX-11/780 can be used as a host development system for RSX-11M and RSX-11S. All but the final debugging can be done on VAX-11/780.

To Conclude:

For the hundred of thousands of persons who have worked with and know the PDP-11 family, the new VAX-11/780 will be the simplest new system to learn.

For those who need more power than a PDP-11 can offer the VAX-11/780 system offers performance, functionality, up time, easy migration and great compatibility.

- A word about prices and delivery.

System prices start at \$128,000 for a system with 128Kb of memory, 2 disk drives of 14Mb each, 8 asynchronous lines, a console terminal and the VAX/VMS operating system. A typical system configuration which includes 512Kb of memory. One 176Mb disk drive, an 800/1600 bpi magnetic tape, 8 asynchronous lines, a console terminal and the VAX/VMS operating system is priced at \$185,000.

Deliveries are scheduled to start in early 1978; volume production is expected to be reached in mid 1978.



For Further Information:
Richard O. Berube
(617) 897-5111 Ext. 3046

DIGITAL'S VAX-11/780 SYSTEM:

A New Direction in Computer Development

BOSTON, Mass.--October 25, 1977--The new VAX-11/780 system introduced today by Digital Equipment Corporation combines the functionality, capacity and performance usually found only on large mainframe systems with the best features of minicomputers, and offers them at a remarkably low price.

Highlights of the new system include 32-bit wordlength, four billion bytes of virtual addressing space, a new Virtual Memory operation system, compatibility with Digital's 16-bit PDP-11 family and built-in reliability and maintenance design innovations.

According to Andrew C. Knowles, Digital Vice President and Group Manager, the new system "provides the high capacity, wordlength and throughput of conventional mainframes together with the interactive capabilities, design innovation and price/performance features of a minicomputer. With this level of designed-in flexibility, the VAX-11/780 is suited to a wide variety of applications in scientific/time-critical, computational, control, data processing and interactive timesharing. And it has impressive multi-user batch capabilities as well," he said.

-more-

VAX-11/780: THE HARDWARE

- o 32-bit wordlength provides up to 4.3 billion bytes of virtual addressing space.
- o Main memory subsystem is ECC MOS memory using 4K MOS RAM chips. Minimum system configuration provides 128K bytes of physical memory which is expandable up to 2 million bytes.
- o Memory controller includes request buffer which increases system throughput and eliminates most of the need for interleaving.
- o Complete and powerful instruction set consists of 243 instructions, 9 addressing modes and 5 data types. Designed for the generation of fast, efficient compiled code, the instruction set includes integral floating point, packed decimal arithmetic, character string manipulation and context switching instructions.

As an example of efficient code generation, a FORTRAN DO loop translates into one instruction. Calls to subroutines, and return to main program combine up to 15 operations in just one instruction.

- o 8K byte write-through cache memory yields effective memory access time of 290 nanoseconds.
- o Optional floating point accelerator performs double precision floating point 64-bit addition in 1.4 microseconds.
- o Paging memory management is supported with 4 hierarchical protection modes, each with read-write access control.
- o Sixteen 32-bit general registers and 32 interrupt priority levels, 16 each for hardware and software.
- o Two standard clocks: programmable real time clock and time-of-year clock with battery backup for automatic system restart.
- o A Synchronous Backplane Interconnect (SBI) serves as the main control and data transfer path. It is capable of aggregate throughput rate of 13.3 million bytes per second.
- o MASSBUS interfacing adapter permits connection of high-speed PDP-11 Peripheral devices (e.g. RP06 Disks and TE16 Mag Tapes); UNIBUS interfacing adapter allows connection of conventional PDP-11 peripherals (e.g. smaller disks, CRTs and printers). One UNIBUS adapter and up to four MASSBUS adapters can be connected to the backplane.

VAX-11/780: THE HARDWARE (continued)

- o MASSBUS connects to the SBI via a buffered adapter and permitting the interfacing of high performance mass storage peripherals with parity checking. Throughput rate here is two million bytes per second.
- o Adapter pathway between UNIBUS and SBI has a throughput rate of 1.5 million bytes per second.
- o Console subsystem incorporates intelligent LSI-11 microcomputer with 16K bytes of read-write memory and 8K bytes of read-only memory, single floppy disk and LA36 teleprinter.
- o The console permits simplified bootstrapping, improved distribution of software updates and fast on-line diagnosis, either local or remote.

VAX-11/780: THE SOFTWARE

VAX-11/780 boasts a new virtual memory operating system, VAX/VMS, which applies mainframe software technology by allowing programs much larger than the physical memory to be run in a way that is transparent to the programmer.

Essentially, the new system will take any size program.

- o Single virtual memory operating system for multiple functions.
- o Full demand paging operation permits programs as large as 32 million bytes.
- o Memory management facilities can be controlled by the user, who can lock pages of a program in memory never to be paged out or can lock an entire program in memory never to be swapped out. This feature is particularly important for time-critical applications.
- o System supports 64 interactive users simultaneously.
- o Program development capabilities include two editors, language processors, symbolic debugger, librarian, and utilities.
- o Languages include VAX-11 FORTRAN IV PLUS, VAX-11 MACRO, PDP-11 COBOL and PDP-11 BASIC-PLUS-2, with FORTRAN and MACRO generating 32-bit native code on the VAX-11/780.
- o Operating system provides file and record management facility allowing users to create, access and maintain data files and records with full protection.

VAX-11/780: THE SOFTWARE (continued)

- o New operating system supports networking capabilities for task-to-task, access and file transfer and down-line loading.
- o Batch capabilities include job control, multi-stream, spooled input and output, operator control, conditional command branching and accounting.
- o DIGITAL command language (DCL) and MCR command languages provided.
- o 32 levels of software process priority for fast scheduling.
- o Record and file management facilities include sequential and relative file organization, sequential and random access.
- o Applications Migration Executive allows RSX-11M/S non-privileged tasks to run with minimal or no modification.

RELIABILITY, AVAILABILITY AND MAINTAINABILITY PROGRAM

The VAX-11/780 system is designed to be the most reliable, available and maintainable computer system of its class built to date, through the inclusion of reliability and maintenance design innovations. These features have been designed into the hardware architecture and software architecture, individual component and board designs and in the cabinetry.

A diagnostic console contains an LSI-11 microcomputer which provides automatic consistency and error checking to detect abnormal instruction uses or illegal machine conditions. Integral fault detection and maintenance features detect errors on memory or disks, record recent bus activity, detect hung machine conditions and allow restart recovery.

Among the several monitoring activities performed automatically are parity checking for data integrity on the synchronous backplane interconnect, MASSBUS and UNIBUS adaptors, memory cache, address translation buffer, and error detection and correction (ECC) on memory. Also performed are operating system consistency checks, redundant recording of critical information, uniform exception handling, on-line error logging, on-line diagnostics and unattended automatic restart.

-more-

PDP-11 COMPATIBILITY

According to Digital's Bernard LaCroute, product manager for the VAX-11/780, "one of the design goals for the new systems was compatibility with other PDP-11's. The result is an instruction set for the new 32-bit system that is extremely rich, through the use of microprogrammed logic. The new instruction set has the same mnemonics as the PDP-11. The system also includes a compatibility mode which provides the PDP-11 instruction set, with the exception of privileged and floating point instructions.

Like other PDP-11s, VAX-11/780 uses both DCL and MCR command languages and implements the same FORTRAN-IV-PLUS, BASIC-PLUS-2 and COBOL languages. FORTRAN generates native 32-bit code on the VAX-11/780, and can concurrently execute a subset of the PDP-11 instruction set in its "compatibility" mode.

The VAX-11/780 system can also be used as a host development system for RSX-11M and RSX-11S operating systems running on PDP-11s. Like other PDP-11s, the new 11/780 uses a UNIBUS for connecting to peripherals, and like the PDP-11/70 it uses integrated MASSBUS adapters for interfacing high-speed peripherals.

The on disk structure is the same as RSX-11, and IAS; the RMS file access methods are the same as RSX-11, IAS and RSTS/E.

"In sum, our conscious design of the new system to be compatible with the 50,000 PDP-11 systems installed throughout the world will make it simple for the hundreds of thousands of people who have worked with the PDP-11 family to make an easy migration up to the VAX-11/780 system," he said.

SYSTEM CONFIGURATIONS

For end users, the VAX-11/780 system will be offered in three standard system configurations:

- o Minimum system configuration consists of VAX-11/780 CPU with 128K bytes of ECC MOS memory, LA36 DECwriter II console terminal, two RK06 14-megabyte disk drives and a multiplexer that provides eight EIA terminal connections and VAX/VMS operating system. Price: \$128,000.
- o CPU with 256K bytes of ECC MOS memory, one RM03 67-megabyte high performance disk, one TE16 mag tape drive, and 8-line multiplexer and VAX/VMS operating system. Price: \$153,000.
- o CPU with 512K bytes of ECC MOS memory, one RP06 176-megabyte high performance disk drive and one TE16 800/1600bpi magnetic tape drive, an 8-line multiplexer and VAX/VMS operating system.
- o All systems have provisions for additional memory and peripherals.
- o System components will be available to OEMs.

VAX-11/780 QUESTIONS and ANSWERS

For Press Conference and Stockholder Meeting

PDP-11 and DEC 20

Q - How is this machine going to impact the PDP-11 business?
What is the future of the PDP-11.

A - The VAX-11/780 is an extension of the PDP-11; it offers more functionality and performance for those customers who need it while complementing the PDP-11 offerings. PDP-11 hardware and software will continue to be aggressively enhanced to maintain their price performance leadership in the 16 bit world.

Q - Do you expect in five years to have the same ratio of business between VAX and the PDP-11 as you have today between the PDP-11 and the PDP-8?

A - The ratio is not that important; what counts is the right set of products at the right time to meet our customers needs.

Q - How is this machine going to impact the DEC 10/20 business?
What is the future of the DEC 10/20?

A - The VAX-11/780 is an extension to the PDP-11 and will be sold for the same type of applications as well as new ones which require more functionality. Just as for the PDP-11 the DEC 10/20 will continue to be enhanced. VAX-11/780 complements the PDP-11 and DEC 10/20 offerings.

Q - Who would want to buy a PDP-11/70 when there is only a \$30,000 difference between the two systems?

A - Those customers who do not need the 32 bit functionality of the VAX-11/780. Why would somebody pay more to get something they don't need?

PDP-11 and DEC 20 (cont.)

Q - Why didn't you choose a 36 bit architecture for your new machine? Is 36 bit obsolete?

A - We wanted the new machine to key off the PDP-11 for compatibility reasons. This has nothing to do with our 36 bit architecture obsolescence.

Q - What do you anticipate the average VAX configuration to be? How does it relate to the PDP-11/70 and the DEC 20?

A - The VAX-11/780 system configurations start at \$128,000. A configuration with $\frac{1}{2}$ Mb memory, a 176 Mb disk drive and tape is priced at \$185,000.

The PDP-11/70 average system is around \$200,000; the average DEC 20 system is around \$450,000. We expect the average VAX-11/780 configuration to be in the \$250,000 - \$300,000 range.

Q - Did you cut your PDP-11/70 prices to make room for this new machine?

A - No - PDP-11/70 price reduction reflects our increased manufacturing efficiency; so does the pricing of the VAX-11/780.

VAX-11/780 Specific

Q - Are you now spending most of your R & D dollars on VAX now?

A - No. No comments about any figures.

Q - You have acknowledged spending a lot of resources developing this machine. When is the next VAX machine to be expected and where is it going to be positioned?

A - Just as for the PDP-8, PDP-11, DEC 20 we will have the right product at the right time.

Q - Is this machine a minicomputer or a mainframe?

A - It is what you want it to be; mainframe capabilities at minicomputer prices.

Q - How many machines are you building in the first year?

A - The number is expressed in hundreds.

Q - What is the anticipated yearly number of machines at volume production?

A - As many as we need to satisfy our customers demand.

Q - Who do you anticipate your customers/markets to be?

A - The traditional PDP-11 customers and applications plus those new applications which need 32 bit word length in the OEM, Scientific, Realtime computation market place.

Q - Are you going after the commercial market with this machine?

A - In the same sense as we are with the PDP-11 today.

Q - When are you going to add COBOL?

A - We do have PDP-11 COBOL on the machine.

Q - What is the performance of the machine?

A - On average twice the speed of the PDP-11/70 for 32 bit programs and operations.

Q - Is there any new technology in the machine (ECL, LSI etc.)?

A - No, we are using conventional Schotky TTL logic.

Q - When are you shipping your first machine?

A - We will be shipping our first machine to a customer testing environment this coming month. Production machines will follow in early calendar 1978 and full production in mid-calendar 1978. (Do not disclose customer names).

VAX-11/780 Specific (Con't'd.)

Q - Have you identified your first customers?

A - Yes, as part of our test marketing effort. (do not disclose customer names).

Q - How many machines have you built so far?

A - Nine. Several more are to follow shortly.

Q - Where is the machine going to be manufactured?

A - In our New Hampshire facilities.

Q - Will you manufacture VAX in Europe? Where? When?

A - The machine will be built initially in New Hampshire. We will consider manufacturing it in Europe later if it makes sense.

Q - Why are you making such a big announcement? It is not the traditional DEC approach.

A - To make sure that we get you the right information about the significance of VAX.

Q - Why didn't you use the acronym PDP?

A - The PDP acronym is implied in the name; VAX-11 stands for Virtual Address Extension to the (PDP) 11.

IBM and COMPETITION

Q - Does this new machine signal DEC's entry in the "IBM World"?

A - NO. VAX-11/780 is an interactive computer designed to serve our traditional markets as well as new applications which require greater word length functionality.

Q - Which IBM machine are you competing (and/or comparing) with?

A - VAX-11/780 is an interactive computer designed to serve our traditional markets as well as new applications which require greater word length functionality. IBM machines are primarily Batch oriented; we can't calibrate ourselves against them.

Q - Is this machine going to replace installed IBM equipment?

A - See answer to previous question.

Q - How do you stack up against the IBM 370 line in terms of raw computer power - or which IBM 370?

A - Very well; but our customers don't use the machine the same way. Ours are interactive, IBM is Batch oriented

Q - The VAX-11/780 has a PDP-11 emulation capability. Are you also planning to emulate the IBM 370?

A - NO; we have never considered this in our design.

Q - Why are you so late with your 32 bit machine, some of your competitors (SEL, INTERDATA) have had one for two years?

A - We don't think we are late; we have the right product at the right time. If you want to compare let's discuss performance and functionality!

Q - When do you expect Data General to follow suit with their 32 bit machine?

A - We don't know; we have the product today, they don't.

Q - How does this relate to the recent 32 bit WANG announcement?

A - We have not had time to look at it.

A G E N D A

VAX Press Conference/Stockholders Meeting

Tuesday, October 25, 1977

Dorothy Quincy Suite, John Hancock Building

VAX PRESS CONFERENCE (Conference Room)

- 9:45am -- Welcome and Introduction: WIN HINDLE
- 9:50am -- DEC product philosophy, family evolution
VAX design goals: GORDON BELL
- 10:05am -- VAX Technical presentation: BERNIE LA CROUTE
- 10:25am -- VAX Markets and Applications: ANDY KNOWLES
- 10:35am -- Questions and Answers (Hindle, Knowles, Bell, LaCroute, Demmer,
et al)
- 10:55am -- Press adjourn to Shareholders Meeting

SHAREHOLDERS MEETING (Dorothy Quincy Suite)

- 11:00am -- Call to order:)
)
11:15am -- Informal Remarks:) KEN OLSEN
)
11:30am -- Stockholder Q&A:)

12 noon -- Adjourn Annual Meeting
 VAX Demo for Stockholders: TOM RARICH, et al

12:10pm -- KHO press conference (conference room)

12:40pm -- Adjourn KHO press conference
 (Press return to the Dorothy Quincy Suite for VAX Demo)

1:15pm -- Press luncheon (Top of the Hub at the Prudential)

JULY 6, 1978

GORDON

A NEW CONSUMER INDUSTRY: ELECTRONIC PHONES/81

C-MOS erasable PROM uses single power supply/ 106

A guide to thermal resistance measurements in ICs/ 121

JUL 07 1978

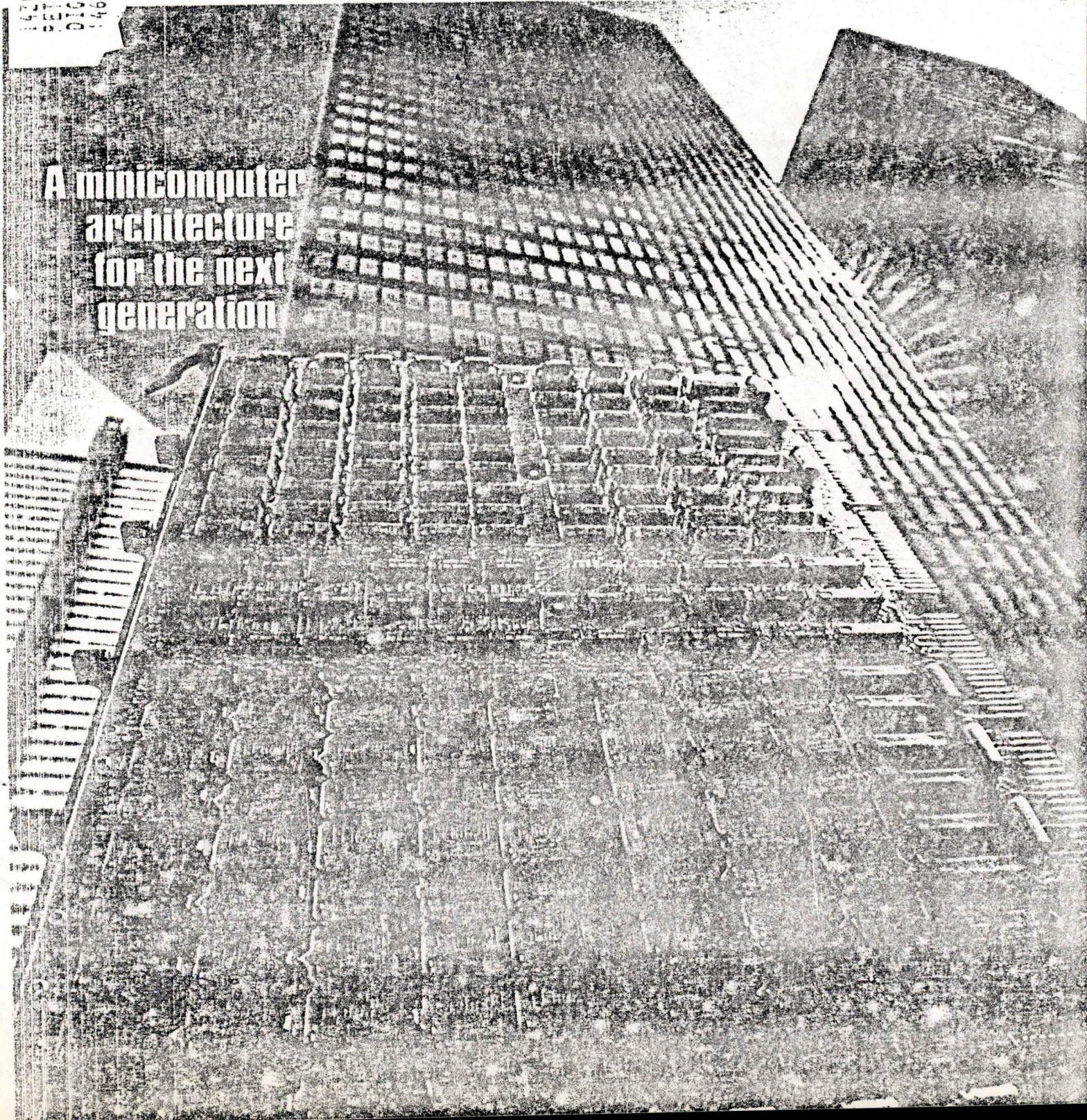
01754

FOUR DOLLARS A MCGRAW HILL PUBLICATION

Electronics

1470 CR-146 M-1217 NOV78
PETER CHRISTY ML12-3 A62
DIGITAL EQUIPMENT CP
146 MAIN ST
HARD MA

A minicomputer
architecture
for the next
generation



Technical articles

Minicomputer architecture links past and future generations

by Peter Christy, *Digital Equipment Corp., Maynard, Mass.*

□ The design and planning of a new series of minicomputers is a difficult problem, especially for a company with a large installed base of a highly successful family. And the problem is greatly magnified when the proposal is for the new family to overstep what were earlier regarded as a minicomputer's limits.

Thus, when Digital Equipment Corp. decided to extend its line into 32-bit mainframe territory, it set in motion a series of complex design decisions requiring a thoroughgoing reexamination of minicomputer architecture in the light both of likely user needs through the 1980s and of likely technological progress through the same period.

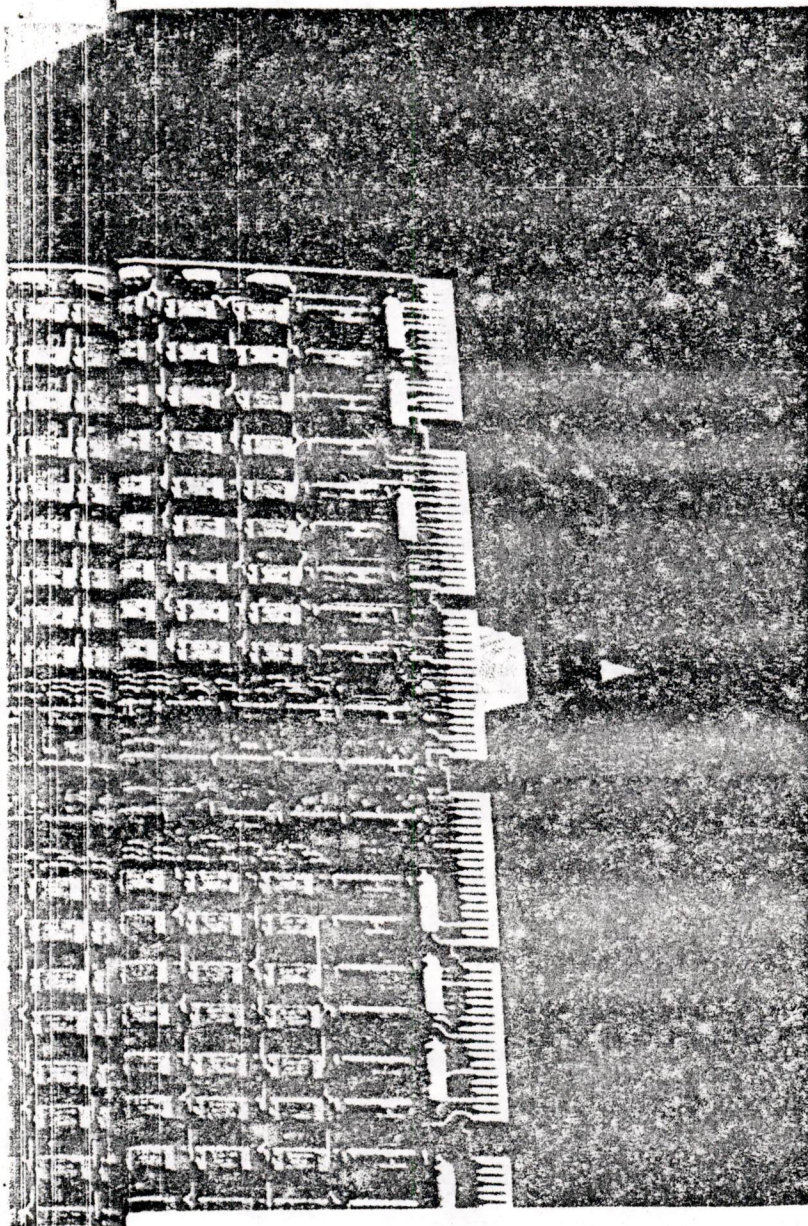
This article addresses some of the issues that guided the development of the VAX-11 architecture, the VAX-11/780 computer system, and the VAX/VMS operating system. The goal was to preserve compatibility with the existing large software investment in the PDP-11 minicomputer family yet to reflect future system needs, in particular by enlarging virtual address space to a huge 4 gigabytes. VAX, in fact, stands for virtual address exten-

sion, and VMS stands for virtual memory system.

The VAX-11/780 32-bit minicomputer system is the high end of the new family. A typical configuration costs between \$150,000 and \$200,000. Initial benchmarks show the machine's Fortran performance, using its fast-floating-point option, to be comparable to that of a modern upper-to-middle-range mainframe costing several times as much. Yet processor, optional floating-point unit, up to 1 megabyte of metal-oxide-semiconductor random-access memory, a Unibus medium-speed input/output controller, and two high-speed (Massbus) I/O controllers come in a single cabinet measuring 47 by 60 by 30 inches (Fig. 1). More memory and various options can be added in extender cabinets.

Parts

All this was implemented with conventional Schottky transistor-transistor logic and standard large-scale integrated memory circuits. Indeed, it was the ready availability of fast, high-density read-only memory that made it possible to design a complex processor without resort-



the
costs
marks
fast-
of a
sev-
ing-
duc-
peed
bus)
7 by
rious

otky
inte-
aila-
made
sort-

ing to anything more expensive than microcoding techniques. Besides the ROM control store, the central processing unit includes 12 kilobytes of RAM control store, which is used for diagnostic functions, some special instructions, and field microcode changes. A further 12 kilobytes of RAM control store is available as an option.

Other RAM parts are used throughout the system to increase performance. The CPU includes some in the form of an 18-kilobyte cache, which keeps the most recently used instructions and data quickly accessible to the processor. Also included in the CPU is a 128-entry address-translation buffer, which is functionally analogous to memory-mapping hardware: it keeps the most recently used translations between virtual and physical memory in high-speed registers, greatly reducing the memory management overhead. RAM is also used throughout the memory busing and I/O subsystem to increase the efficiency of the major busing mechanisms.

As for the new VAX/VMS operating system, it provides the VAX-11/780 minicomputer with the kind of func-

tions previously available only to mainframe computers. Examples are full virtual-memory management, demand paging, indexed data-access methods, and extensive interjob protection and sharing capabilities. VAX/VMS supports up to 64 on-line users simultaneously developing and executing programs in assorted high-level languages. In particular, a compiler for DEC's Fortran IV-Plus language (a superset of ANSI Fortran) has been developed to take full advantage of the extended instruction set of the VAX-11 architecture.

Compatibility with the PDP-11

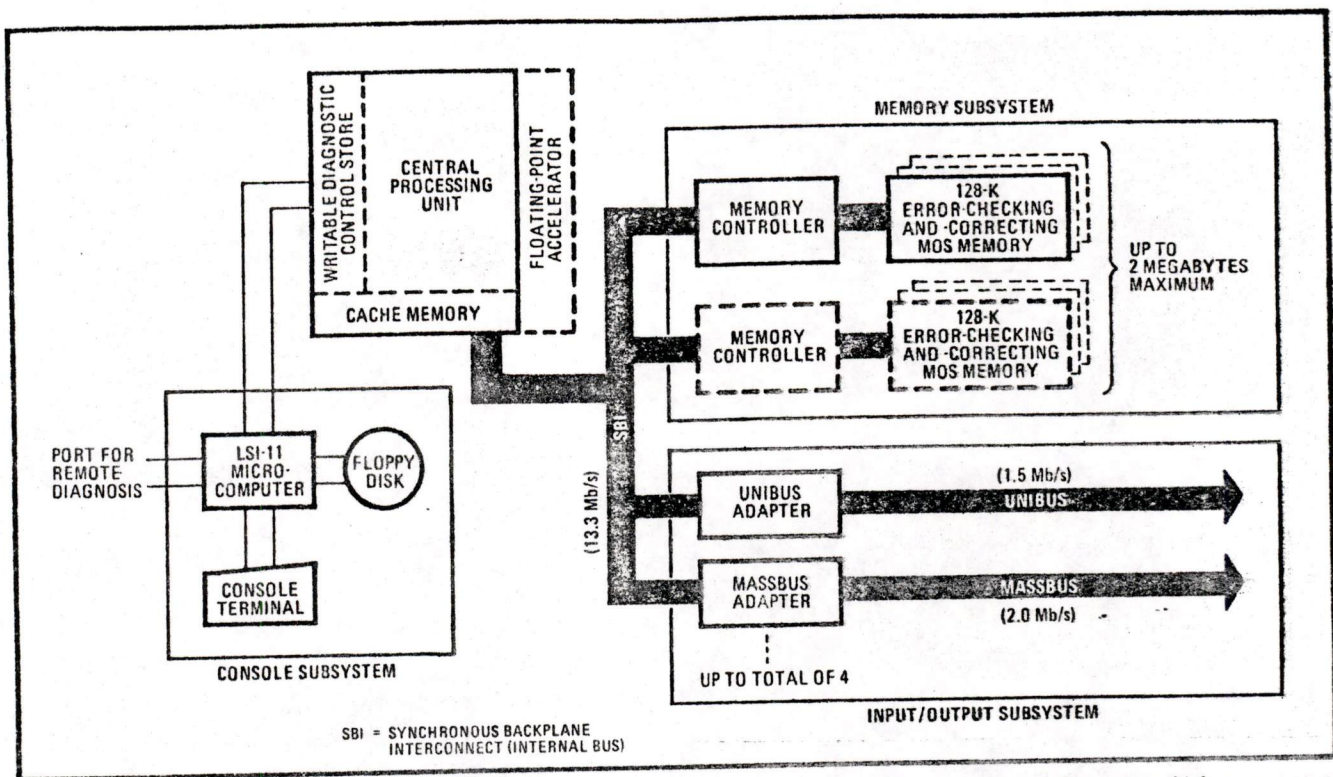
Though the VAX-11/780 is not a 32-bit PDP-11, for reasons that will shortly be gone into at length, cost-saving compatibility with the PDP-11 and its associated software has been achieved as desired at the six most relevant user levels:

- Cultural compatibility, if such a term may be used to describe the stylistic similarity of the machines. Because of it, PDP-11 programmers can produce high-quality VAX-11 native-mode code with little training, and language compiler designs that generate PDP-11 code can be adapted to generate efficient VAX-11 code.
- Operating-system compatibility. Many of the VAX/VMS operating-system functions were modeled after the PDP-11 RSX operating systems (for example, in the type and form of system calls, the way in which tasks synchronize with each other and exchange data, etc.). Key VAX/VMS functions, such as the file system and record management facilities, are functionally identical to their RSX counterparts.
- High-level language compatibility. VAX-11 languages are designed to be compatible with the existing PDP-11 compilers. A calculation program written in Fortran IV-Plus for the PDP-11 runs unchanged on VAX-11.
- Direct processor support for user-mode PDP-11 programs. The VAX-11 architecture includes a PDP-11 compatibility mode, in which the processor behaves just like a user-mode PDP-11, except that it can simultaneously run 32-bit code jobs.
- Data compatibility. All PDP-11 data formats were brought forward to the VAX-11 architecture.
- Data-file compatibility. VAX/VMS is able to create and access disk files that are compatible with the PDP-11 RSX-11 operating systems.

Why not a 32-bit PDP-11?

During planning of the VAX-11, one idea that received serious consideration was in fact a 32-bit PDP-11. Most of the PDP-11 architecture is already independent of word length, and recent architectural studies had demonstrated that the PDP-11 is a bit-efficient architecture, even compared with mainframe architectures. In short, a PDP-11-like machine with an extended virtual address space would evidently be an attractive computer, today and tomorrow.

The most distinctive attribute of the PDP-11, and the basis for its architectural power, is the flexible way in which its registers can be used to form addresses. This flexibility permits the machine to be used effectively for many different types of computing, unlike most previous architectures, which tended to be good for one style of



1. New generation. The VAX-11/780 computer system consists of the central processing unit, the console subsystem, which serves as an operating system terminal and system or diagnostic console, the main memory subsystem, and the I/O subsystem. All major hardware components, implemented by Schottky TTL and standard MOS memory devices, are connected through the SBI, an internal synchronous bus.

processing but poor for another. For example, a design that has many central registers but not stack-like characteristics is good for scientific calculation but poor for complex subroutine structures. Conversely, a machine designed around a stack architecture is good for program control but inefficient for intensive calculation. But the PDP-11 is able to take on either set of attributes, and others, whenever a task demands it.

However, a 32-bit PDP-11 would have meant extending the register width to 32 bits but keeping the instruction formats and encodings unchanged, and this turned out to be an impossibility. The idea would have been a machine that could execute existing PDP-11 machine code intermingled with 32-bit code that made full use of the 4-gigabyte virtual address space. But a careful examination of a 32-bit version of the PDP-11 uncovered some unsurmountable obstacles.

There turn out to be many ways in which a programmer can implicitly design the address length into a program. For example, before control is passed to a subroutine, parameters may be pushed onto the stack. The subroutine call itself leaves the return address on the top of the stack. Within the subroutine, the parameters are accessed with respect to a known displacement from the top of the stack. But unfortunately, changing the address length from 2 to 4 bytes makes these known offsets invalid. This and many similar problems ruled out the possibility of executing 16-bit code unchanged in a larger address space, or of automatically translating 16-bit programs into a 32-bit form.

Given the difficulties of directly extending the PDP-11 design to a 32-bit form, the next alternative was to see what improvements could be gained by a bit-level-in-

compatible, but otherwise highly similar, design. The result was the VAX-11, a substantially better design that, though not precisely like the PDP-11, is "culturally compatible" with it. Hardware and software were also developed that permit a large subset of existing PDP-11 programs to execute without any changes on a VAX-11 system, as described earlier.

Architectures and word lengths

The description of the VAX-11 as a 32-bit minicomputer and the PDP-11 as a 16-bit minicomputer implies that the essential difference between them is their word length. But any significant difference in the architectures would presumably be measurable in terms of their comparative bit efficiencies on important applications. As it turns out, the bit efficiency of the PDP-11 is excellent, and in most respects the PDP-11 is not restricted to a 16-bit word length.

The problem is that the term "word length" has too many meanings to be useful without qualification. In a typical computer system, many different word lengths can be identified. In this context, therefore, it is necessary to eliminate from consideration the word lengths that represent engineering decisions for specific implementations and to consider only those that are intrinsic to an architecture and affect all its implementations in the family.

Instruction length is a possible candidate here. But both the PDP-11 and the VAX-11 have instructions of variable length, ranging from 16 to 48 bits and 8 to 296 bits respectively. In both cases, the variable-length instruction format offers better bit efficiency than an equal-length format because common instructions can

The importance of bit efficiency

A good architecture is reflected in a computer's static and dynamic bit efficiency. Bit efficiency is a quantifiable measure of how well the investment in the computer system's components pays off in application-level throughput. In other words, if two systems are built with the same technology and the same complexity, then the one with the greater bit efficiency will be more cost-effective (assuming that the bit-efficient instructions can still be rapidly decoded and executed by the processor).

Static bit efficiency is the relative size of a program compared with the size of a program coded for an architecture defined as a standard. A good static bit efficiency reduces the requirements for central memory and program file storage and streamlines the tasks involved in program-moving overhead, such as initial program loading, fetching overlays, paging, or swapping.

Dynamic bit efficiency is a comparative measure of how many program bits must be fetched from memory to the processor to execute a program. If all machine instructions were used with the same frequency, then static and dynamic bit efficiency would be the same. In practice, some instructions and data types occur often and others occur rarely. Good dynamic bit efficiency reflects the fact that the most frequent instructions (such as loop control

instructions) have particularly good encoding.

The ideal way to compare bit efficiencies would be to take a specific set of application programs and measure their actual bit efficiencies on different architectures. Unfortunately, such an approach is impractical for a computer vendor because customers have many disparate applications and many architectures of interest are hypothetical.

Fortunately, there are ways to characterize typical applications. Those coded in common high-level languages, such as Fortran, Cobol, and Basic, may be related to studies of typical program behavior, which show that in each of these languages different statements and data types have a characteristic frequency of occurrence. With these statistics and with an understanding of the machine code generated for each common statement, it is possible to estimate the bit efficiency of real or hypothetical architectures.

Bit efficiency is a good general test of architectural effectiveness, since it diminishes with any difficulty in machine-level programming or compiler code generation. Good static bit efficiency reflects effective use of system components; good dynamic bit efficiency reflects effective use of memory system bandwidth.

have shorter encodings. No architecturally useful definition of word length can be derived from instruction length, therefore.

Both the PDP-11 and VAX-11 are byte-address machines, since all data types are addressed in main memory by the byte address at the beginning of the data item, regardless of whether the data is a 1-byte character or an 8-byte double-precision, floating-point number. So memory addressing is also no help in defining the architectural difference between the machines.

But the PDP-11 has 16-bit general registers, whereas the VAX-11 has 32-bit general registers. In both architectures the registers can be used for arithmetic on data items that are shorter than the register size (8-bit integers on the PDP-11, 8- and 16-bit integers on VAX-11) or can be used in multiples for data items that are longer than the register size. Register length as used in arithmetic is not an invariable word length, therefore.

Register length

However, register length as used in instruction address formation is another matter: it does define the essential difference between the PDP-11 and the VAX-11, since it determines the size of logical storage that a program can instantaneously address—the size of the virtual address space. Thus the PDP-11's 16-bit byte address creates a virtual address space of 65,536 bytes, whereas the VAX-11's 32-bit byte address creates more than 4 billion bytes of virtual address space. What's more, virtual address space limitations can affect bit efficiency.

At the time the PDP-11 was designed, it seemed unlikely that any minicomputer would need more than 65 kilobytes of *physical* (as opposed to virtual) memory. In retrospect, the designers realize they failed to anticipate how rapidly central memory costs would decline.

Early in the evolution of the PDP-11 family, hardware memory mapping was added to the top-range machines. PDP-11 mapping logically divides the 64-K virtual address space into eight 8-K pages, each of which can be located independently in physical memory and protected independently (Fig. 2).

The addition of mapping offered two major benefits:

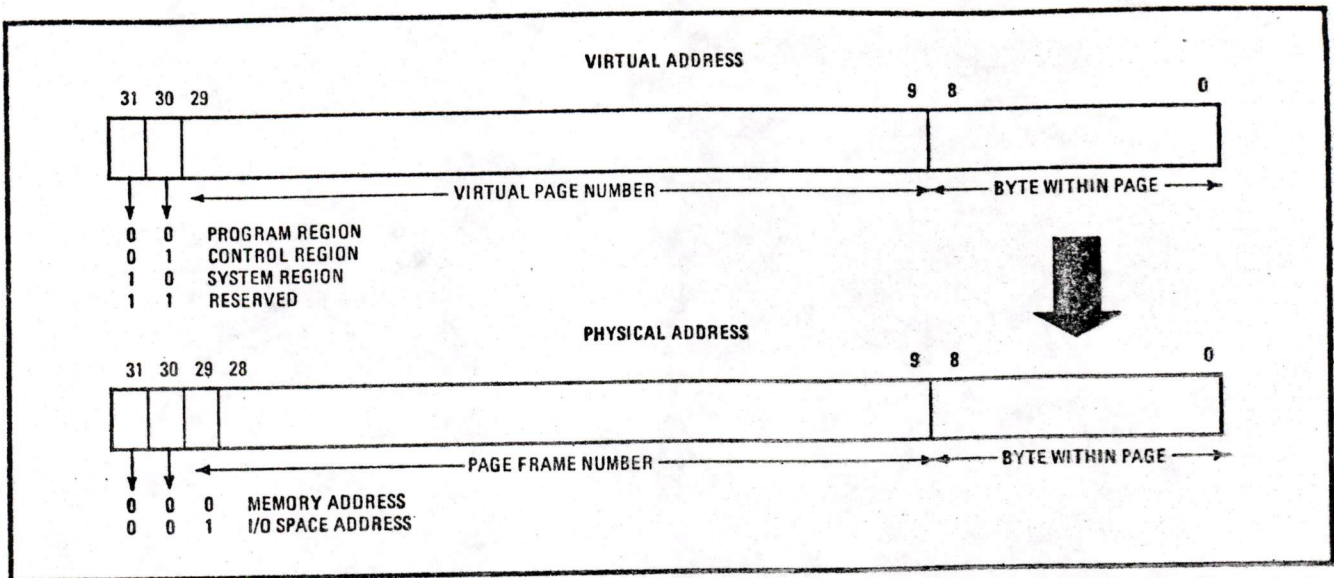
- It permitted the design of multiprogramming software systems in which a user program is prevented from damaging another user program or the operating system code, since each program can address only those parts of central memory allocated to it.
- It permitted the design of configurations with more than 65 kilobytes of physical memory, since the mapping hardware can translate 16-bit addresses into physical memory addresses of arbitrary length. Thus a PDP-11/70 may have in excess of 4 million bytes of central memory by developing 22-bit addresses.

For most applications, the remapping overhead is insignificant. But there are calculation applications in which it induces noticeable bit inefficiency. Also, although today most minicomputer programs and their data fit naturally into a 65-kilobyte address space, the trend to larger central memories will surely lead to larger program sizes as well.

The need for large address space

Consequently, though the immense marketplace success of the PDP-11 and other minicomputer architectures demonstrates that limited address space has not so far been felt as a restriction, it might become one in the future. Since the need for large address space is felt first in large configurations, the VAX-11/780 was designed as a top-of-the-line minicomputer.

The fundamental need of VAX-11 was to solve the



2. Address formats. Virtual addresses are 32 bits long in format shown at top. The virtual address space (memory space as it appears to a given process) is mapped onto the physical address space (lower format), which is the actual memory space existing in hardware. Memory-management logic controls the mapping and maintains in physical memory only portions of virtual memory being used.

addressing problem. Given the byte orientation of the PDP-11 and the need to do address arithmetic conveniently, the obvious address lengths to explore were 24 bits (16+8), 32 bits (16+8+8), etc.

Memory costs are roughly halving each year, and 1 more bit of physical address per year is needed. A 24-bit virtual address seemed too small. For VAX-11 the choice was 32 bits, representing a 4-gigabyte virtual address space (see Fig. 2 again). The next decision was to make this address space linearly addressable, meaning that there would be no further segmentation (many mainframes offer comparably large virtual address spaces but break them into many segments). This is large by any standard today, including mainframes, and should also allow a good decade of growth before the size of typical physical memories comes anywhere near the virtual address space. The very large, linear, virtual address space will also permit flexible evolution in software system design techniques, should that prove needed, for example, in advanced file or data-base management system designs.

Picking the virtual address size was a key decision, but a simple one compared with the total question of virtual memory design. Other issues included:

- Feasibility. It had to be possible to implement without adding a large cost penalty to the processor.
- Functionality. It had to be able to support sophisticated data-processing applications.
- Efficient use of control memory. The VAX-11 architecture had to exhibit a wide range of system performance without requiring large amounts of central memory for control tables.
- System efficiency. A virtual memory design would have to work well in real-time applications, yet be able to switch rapidly from one program to another while responding to external requests.

A 32-bit virtual memory design is totally different from a 16-bit one. For example, mapping a 65-kilobyte PDP-11 program consists of initializing eight internal processor registers. In contrast, mapping a 4-gigabyte

address space would mean specifying 8 million page relocations (on VAX-11, each page or separately mappable unit of virtual memory is 512 bytes). Obviously, no VAX-11 processor will have 8 million internal relocation registers!

Similarly, the operating system on a PDP-11 does little in the way of virtual memory management. Because programs are relatively small, they are typically swapped in and out in their entirety. But the large virtual address space on VAX-11 encourages programmers to build large, logically connected programs that may well be much bigger than the physical memory of the system. For example, a large computation that had previously been structured in overlays will simply be a big program on VAX-11. But making the virtual memory useful to the application programmer means that the operating system has to be much more active in deciding which pieces of an application program should be kept in central memory and which should be kept on disk-backing storage (called working-set management). This in turn means that the mapping mechanism must be efficiently controlled by operating system code and must provide as much useful information about the dynamic usage of virtual memory as possible.

Sharing virtual memory space

A key feature of the VAX-11 virtual memory design, which leads to efficient use of central memory and low overhead during execution, is the way in which the operating system shares virtual memory space with user processes. The entire 4-gigabyte virtual address space is logically divided into halves. The user process is limited to the use of the bottom half, and most of the operating-system code resides in the top half (Fig. 3). (Remember that this is just virtual memory; the decision about what virtual memory is made resident in central memory, including operating-system code, is made dynamically on the basis of actual need.) The mapping of each user process may be unique, or user processes may share program and data pages with one another, but in either

case the same virtual operating system is mapped with each user process.

Putting the operating system into a single address space rather than having pieces of the system code in multiple address spaces minimizes the need for system mapping control tables and makes intersystem communication more efficient. Having the operating system share the virtual address space of each user process simplifies requesting services from the operating system. The high-speed processor translation caches, which store the most recently used mapping translations, treat system and user mapping separately so that system mapping translations stay in the cache when the operating system switches to another user process, but the user translations are flushed out.

In a simple memory scheme, it would be risky to put the user programs in the same address space as the operating system, since their malfunctions could affect the system operation. In VAX-11, the system is totally protected from this by a separate access control mechanism. The processor executes in one of four modes:

- Kernel, for interrupt processing, physical I/O control, processor scheduling, and the like.
- Executive, for file management and similar functions.
- Supervisor, for functions such as interactive command processing.
- User, in which user programs are executed.

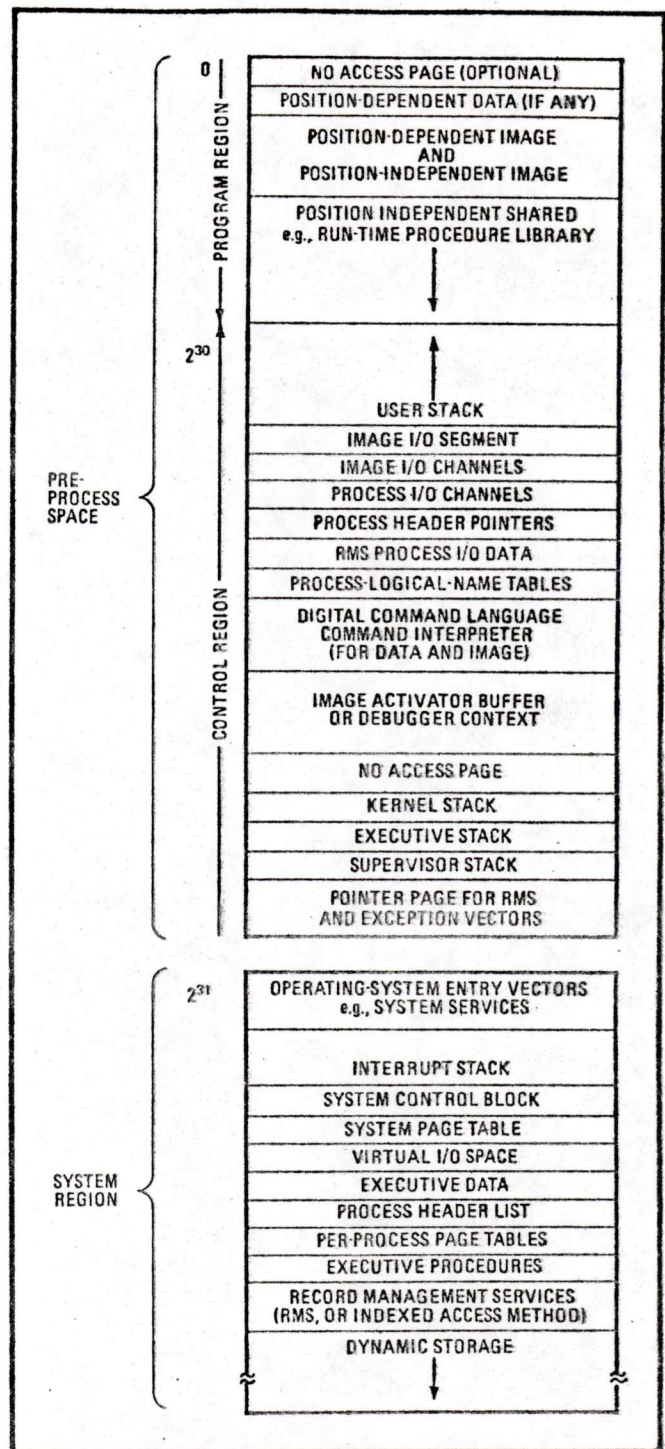
The accessibility of each page in the virtual memory space (whether the page can be read, written, or both) can be controlled for each of the processor execution modes. Thus the operating system can keep critical data in the user's address space and can access that data freely during execution of an operating system service (during which the processor runs in a more privileged mode), and yet the user's program may be restricted from reading the data, if that is inappropriate.

Compatibility mode

The VAX-11 user mode can be put into a compatibility mode, which makes it capable of executing many PDP-11 programs often faster than the PDP-11/70, the top-of-the-line PDP-11. The efficiency of PDP-11 emulation is due to the strong cultural compatibility between VAX-11 and the PDP-11. Thus a processor designed to perform VAX-11 instructions efficiently can also perform PDP-11 instructions well. Instruction execution of the VAX-11/780 is implemented with microcode (as is true of most computers today); PDP-11 compatibility-mode emulation was primarily implemented with a 10% increment of microcode.

This VAX-11/PDP-11 compatibility mode is worth exploring in some detail. Some earlier "compatible" emulation modes required that the computer be used only in one mode at a time. But although 16- and 32-bit code cannot be freely mingled within a single program on the VAX-11, compatibility-mode jobs and native jobs can run at the same time, sharing the resources of the VAX/VMS multiprogramming system. The two kinds of jobs can even cooperate with each other by exchanging messages or by sharing files.

Emulation modes have also been used in the past in the place of software development for the new architec-



3. Allocation. Virtual address space is divided into halves. The lower address locations are limited to user processes, the upper to operating-system code. User processes may share program and data pages, but virtual operating system is always the same.

ture. However, in the case of VAX-11, the power of the enormous virtual address space and new instructions were intrinsic to the value of the system. So there seemed to be little value in a hardware-supported compatibility mode that would execute a complete PDP-11 operating system. Instead, as already indicated, the compatibility mode is limited to user-mode programs. Those operating-system utilities that are insensitive to the size of the address space have been taken from the earlier family's

DATA TYPES HANDLED BY VAX-11			
Data type	Size	Range (decimal)	
Integer		Signed	Unsigned
Byte	8 bits	-128 to +127	0 to 255
Word	16 bits	-32,768 to +32,767	0 to 65,535
Long word	32 bits	-2^{31} to $+2^{31}-1$	0 to $2^{32}-1$
Quad word	64 bits	-2^{63} to $+2^{63}-1$	0 to $2^{64}-1$
Floating point		$\pm 2.9 \times 10^{-37}$ to 1.7×10^{38}	
Floating	32 bits	approximately 7-decimal-digit precision	
Double floating	64 bits	approximately 16-decimal-digit precision	
Packed decimal string	0 to 16 bytes (31 digits)	numeric, two digits per byte sign in low half of last byte	
Character string	0 to 65,535 bytes	1 character per byte	
Variable-length bit field	0 to 32 bits	dependent on interpretation	

RSX-11M operating system and execute in compatibility mode transparently to the user.

The application migration executive is a subroutine package provided with VAX/VMS that emulates RSX-11M operating support for PDP-11 programs running in the VAX-11/PDP-11 compatibility mode. VAX/VMS has been designed to transfer control to the AME within 50 microseconds on the VAX-11/780 when a PDP-11 compatibility-mode program requests operating-system services. The AME executes as a VAX-11 program, in a 32-bit address space that includes the 16-bit address space of the PDP-11 program. The AME determines which RSX-11M system call is being requested by a PDP-11 program, translates it into VAX/VMS format, and issues the request to VAX/VMS. When control returns to the AME, it translates the results into RSX-11M format, stores the result in the compatibility-mode program data, and then returns control to the PDP-11 program via a VAX/VMS service.

Translators

Use of the AME permits a large collection of the RSX-11M programs to run unchanged on VAX-11 systems under VAX/VMS. Although the AME translates RSX-11M system calls, a similar program could be written to translate calls of other PDP-11 operating systems. A single VAX/VMS system could, theoretically, have translators for multiple PDP-11 operating systems.

The efficiency of compatibility-mode program execution under such a translator depends on how heavily the program uses operating-system facilities and how different the emulated operating system was from RSX-11M and VAX/VMS. Although the translation adds some overhead, the typical VAX/VMS service is faster than RSX-11M (run on a PDP-11/70) because of the increased functionality of the VAX-11. On balance, emulated PDP-11 programs run about as fast as they would in the PDP-11/70 under RSX-11M.

The PDP-11 was designed with 8 general-purpose registers. Since then, the cost increment of additional processor registers has gone down dramatically, and the

VAX-11 was given 16. Apart from cost, the penalties for additional registers are a need for extra system overhead to perform context switching and a reduction in bit-efficiency, since more bits are required to address a register. Nevertheless, these extra registers do provide better compiler optimization of generated code, lower overhead in subroutine usage, and efficient design of complex instructions.

The strength of the PDP-11 architecture is its inclusion of the best features of stack, multiple-register, and memory-to-memory designs because of the versatile way in which its general registers can be used to develop addresses. VAX-11 added to these addressing modes to increase the efficiency of program indexing into those tables that list multiple-byte data items like 4-byte floating-point values or 8-byte integers.

In the last decade the processing capacity of minicomputer systems has increased to the point where they are patently unsuited for very few applications. To support efficiently all likely forms of processing, new data types (forms of data for which processor instructions exist) were added to VAX-11, as shown in the table. VAX-11 implicitly does 32-bit address arithmetic, and instructions were added for explicit 32-bit integer arithmetic and Boolean logic.

Decimal arithmetic

Thirdly, VAX-11 permits arithmetic to be done directly in decimal form, instead of requiring that it be converted to binary form, to ensure that full data precision is retained, and because such data is more often moved intact between data records than used for calculation. Still other new data types are test string manipulation (where strings of characters can be moved, translated, and searched with specific instructions) and a complex editing instruction (to provide for the kinds of manipulations common in generating the typical data-processing report—for example, editing out leading zeros or adding a dollar sign). On VAX-11, direct processor support for 1- to 32-bit data fields has been implemented, increasing the bit efficiency of critical

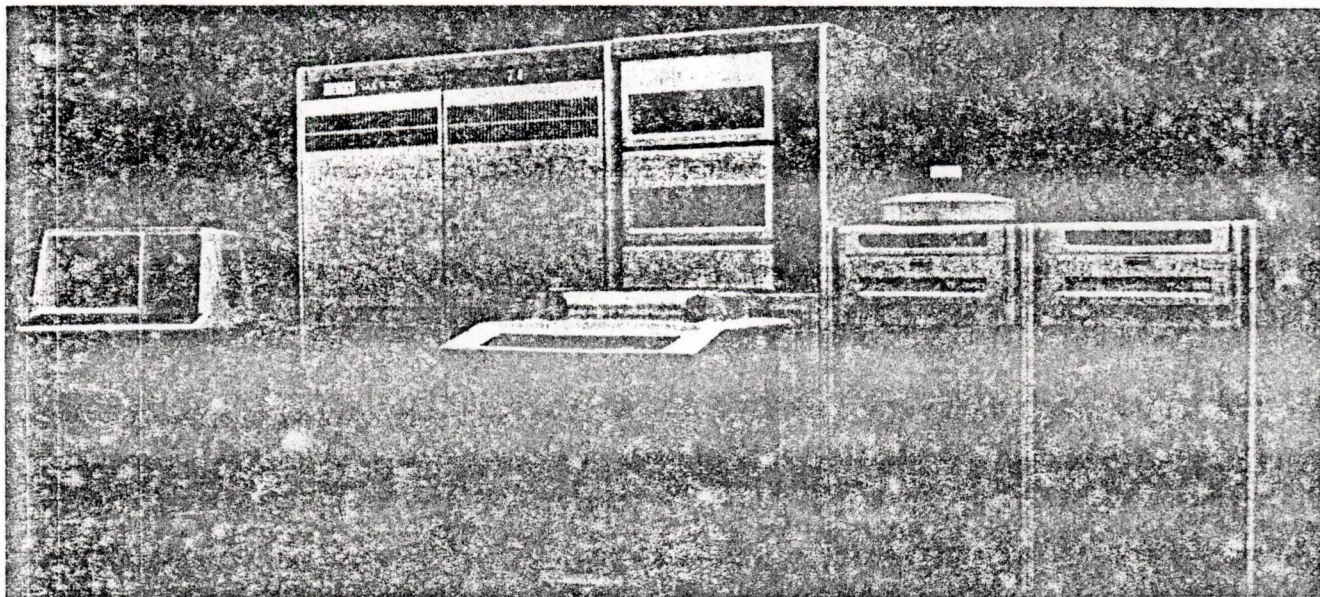
The economics of compatibility

The last thing a computer user wants to do is to rewrite an existing program for a new hardware design. He would rather develop new software for all the profitable new applications it opens up. After all, if the hardware is costing him less, the cost of programming is still as high as ever. Worse yet, it may even be rising, since the number of computers to be programmed appears to be growing faster than the population of skilled programmers.

In view of this, a major design focus for the VAX-11 architecture was compatibility with the PDP-11. Accurate statistics are not available, but if as little as \$20,000 has been spent for software for each of the 50,000 PDP-11s produced this far, then the total investment is \$1 billion. Surely, this is a conservative estimate when it is remem-

bered that \$20,000 buys only a small amount of code.

The size of the entire existing investment in software is incomparably larger, and the need to preserve it might already have halted innovation in the computer industry were it not for the phenomenal rapidity with which computer technology is evolving. The new markets and applications continuously being opened up force architectural changes that compel some level of innovation even at the price of a devalued software investment. Nevertheless, the point has seemingly been reached at which no new computer—mainframe, mini or micro—can be designed without careful examination of the compatibility issue. Barring some remarkable breakthrough in software engineering, compatibility will continue to grow in importance.



Compatibility. Design goal of the VAX-11/780 32-bit minicomputer system shown here was to preserve compatibility with the software developed for the existing PDP-11 family while anticipating future needs. It features greatly extended virtual address space.

operating-system code and like programs. Since field-bit position is specified by a 32-bit integer, very large (512-megabyte) structures can be linearly bit-addressed.

Although most of the added instructions were in support of the data types listed above, many other special instructions were added for operating system support, user programming support, and specific computation needs. For example, an instruction, POLY, has been provided to compute polynomial equations of the form:

$$y = C_1 + C_2x + C_3x^2 + \dots$$

in a single instruction, with the loop overhead handled in microcode. This instruction substantially speeds up the calculation of standard numerical approximation, such as the calculation of sine and cosine functions within a Fortran run-time library.

Future minicomputers

The implications of all these capabilities for the future of the VAX-11 minicomputer family can readily be assessed. The yardstick may be inferred from the appar-

ently constant rates of improvements in base technologies like semiconductors and magnetics and from computer designers' rules of thumb. For example, in a typical system the number of bytes of central memory is about equal to the number of instructions per second by the central processor. Such a system must also be capable of performing 1 bit of I/O for each instruction executed (these rules are sometimes attributed to Gene Amdahl). For any year in the near future, the technologies' price prediction for computer subsystems may be combined with the designers' predictions of the appearance of a balanced system, the constant price definition of minicomputers applied, and the range of expectable system configurations thus delimited.

Following such logic, a minicomputer priced at \$50,000 in the early 1980s should look much like today's mainframe in gross capability, having on the order of a million bytes of central memory, hundreds of millions of bytes of disk storage, and so on. That prediction, made some years ago, led to the VAX-11 design project and the development of the VAX-11/780. □

REPRINTED FROM

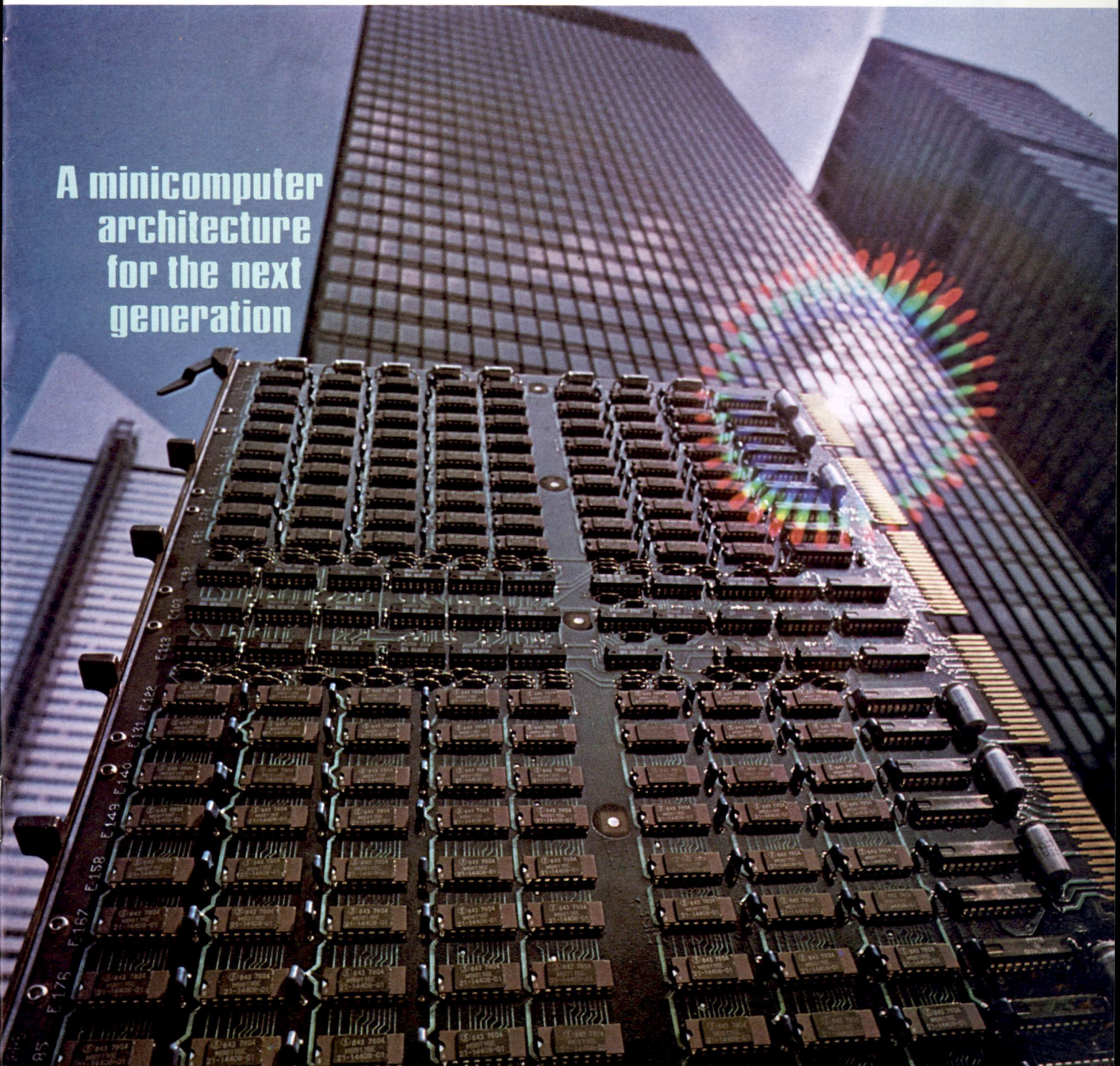
JULY 6, 1978

A MCGRAW-HILL PUBLICATION

Electronics®

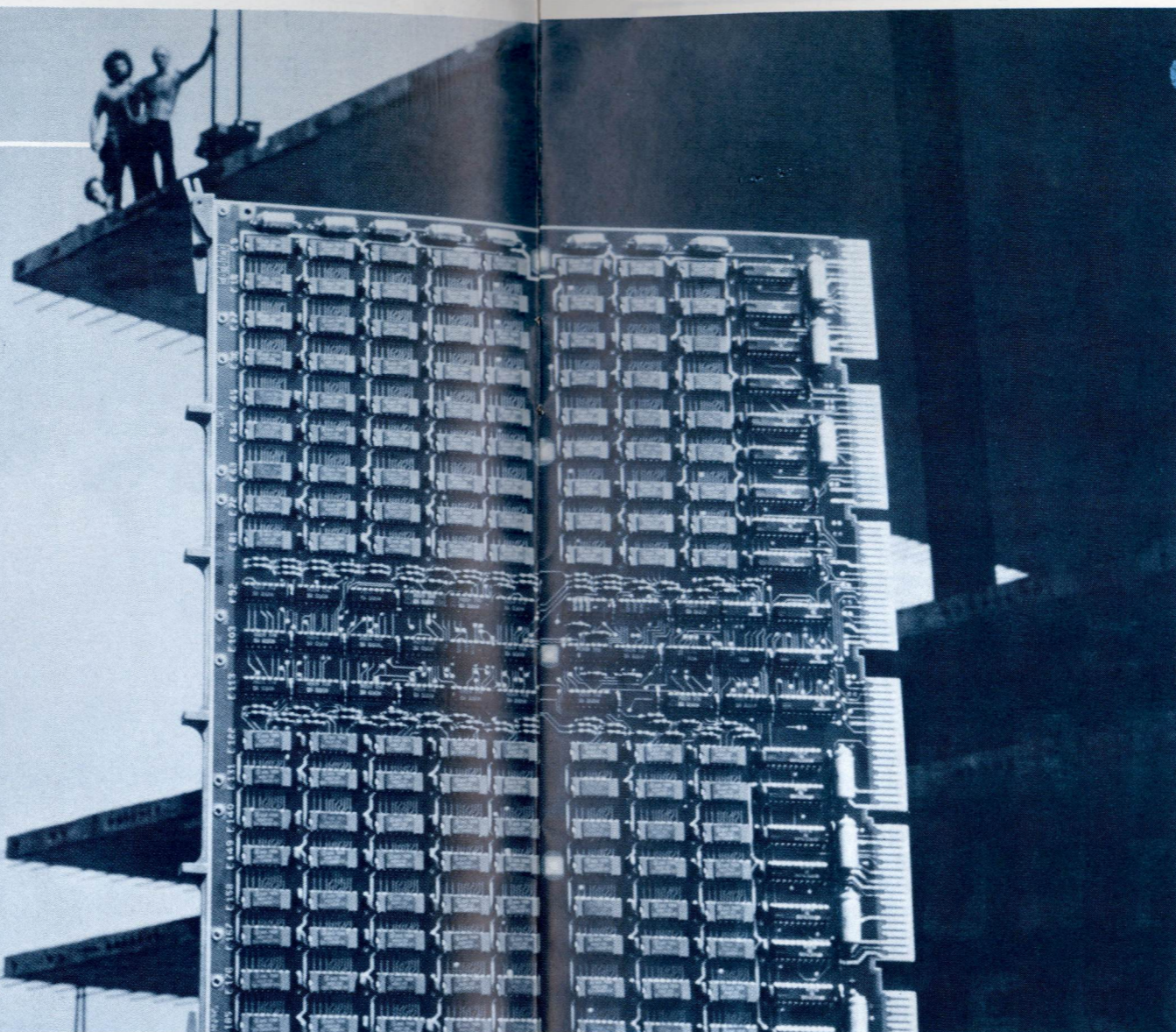
Reprinted from ELECTRONICS, July 6, 1978, copyright 1978 by McGraw-Hill, Inc., with all rights reserved.

**A minicomputer
architecture
for the next
generation**



Minicomputer architecture links past and future generations

by Peter Christy, Digital Equipment Corp., Maynard, Mass.



□ The design and planning of a new series of minicomputers is a difficult problem, especially for a company with a large installed base of a highly successful family. And the problem is greatly magnified when the proposal is for the new family to overstep what were earlier regarded as a minicomputer's limits.

Thus, when Digital Equipment Corp. decided to extend its line into 32-bit mainframe territory, it set in motion a series of complex design decisions requiring a thoroughgoing reexamination of minicomputer architecture in the light both of likely user needs through the 1980s and of likely technological progress through the same period.

This article addresses some of the issues that guided the development of the VAX-11 architecture, the VAX-11/780 computer system, and the VAX/VMS operating system. The goal was to preserve compatibility with the existing large software investment in the PDP-11 minicomputer family yet to reflect future system needs, in particular by enlarging virtual address space to a huge 4 gigabytes. VAX, in fact, stands for virtual address exten-

sion, and VMS stands for virtual memory system.

The VAX-11/780 32-bit minicomputer system is the high end of the new family. A typical configuration costs between \$150,000 and \$200,000. Initial benchmarks show the machine's Fortran performance, using its fast-floating-point option, to be comparable to that of a modern upper-to-middle-range mainframe costing several times as much. Yet processor, optional floating-point unit, up to 1 megabyte of metal-oxide-semiconductor random-access memory, a Unibus medium-speed input/output controller, and two high-speed (Massbus) I/O controllers come in a single cabinet measuring 47 by 60 by 30 inches (Fig. 1). More memory and various options can be added in extender cabinets.

Parts

All this was implemented with conventional Schottky transistor-transistor logic and standard large-scale integrated memory circuits. Indeed, it was the ready availability of fast, high-density read-only memory that made it possible to design a complex processor without resort-

ing to anything more expensive than microcoding techniques. Besides the ROM control store, the central processing unit includes 12 kilobytes of RAM control store, which is used for diagnostic functions, some special instructions, and field microcode changes. A further 12 kilobytes of RAM control store is available as an option.

Other RAM parts are used throughout the system to increase performance. The CPU includes some in the form of an 18-kilobyte cache, which keeps the most recently used instructions and data quickly accessible to the processor. Also included in the CPU is a 128-entry address-translation buffer, which is functionally analogous to memory-mapping hardware: it keeps the most recently used translations between virtual and physical memory in high-speed registers, greatly reducing the memory management overhead. RAM is also used throughout the memory bus and I/O subsystem to increase the efficiency of the major bus mechanisms.

As for the new VAX/VMS operating system, it provides the VAX-11/780 minicomputer with the kind of func-

tions previously available only to mainframe computers. Examples are full virtual-memory management, demand paging, indexed data-access methods, and extensive interjob protection and sharing capabilities. VAX/VMS supports up to 64 on-line users simultaneously developing and executing programs in assorted high-level languages. In particular, a compiler for DEC's Fortran IV-Plus language (a superset of ANSI Fortran) has been developed to take full advantage of the extended instruction set of the VAX-11 architecture.

Compatibility with the PDP-11

Though the VAX-11/780 is not a 32-bit PDP-11, for reasons that will shortly be gone into at length, cost-saving compatibility with the PDP-11 and its associated software has been achieved as desired at the six most relevant user levels:

- Cultural compatibility, if such a term may be used to describe the stylistic similarity of the machines. Because of it, PDP-11 programmers can produce high-quality VAX-11 native-mode code with little training, and language compiler designs that generate PDP-11 code can be adapted to generate efficient VAX-11 code.

- Operating-system compatibility. Many of the VAX/VMS operating-system functions were modeled after the PDP-11 RSX operating systems (for example, in the type and form of system calls, the way in which tasks synchronize with each other and exchange data, etc.). Key VAX/VMS functions, such as the file system and record management facilities, are functionally identical to their RSX counterparts.

- High-level language compatibility. VAX-11 languages are designed to be compatible with the existing PDP-11 compilers. A calculation program written in Fortran IV-Plus for the PDP-11 runs unchanged on VAX-11.

- Direct processor support for user-mode PDP-11 programs. The VAX-11 architecture includes a PDP-11 compatibility mode, in which the processor behaves just like a user-mode PDP-11, except that it can simultaneously run 32-bit code jobs.

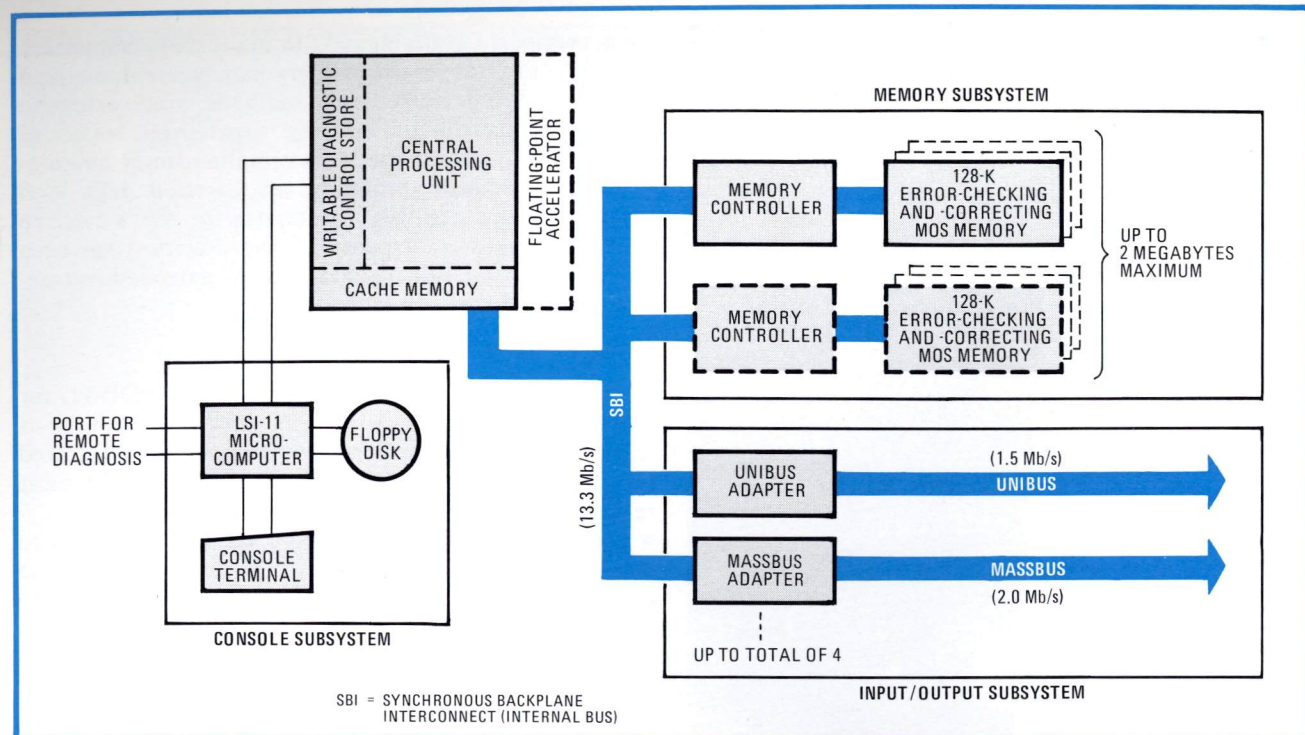
- Data compatibility. All PDP-11 data formats were brought forward to the VAX-11 architecture.

- Data-file compatibility. VAX/VMS is able to create and access disk files that are compatible with the PDP-11 RSX-11 operating systems.

Why not a 32-bit PDP-11?

During planning of the VAX-11, one idea that received serious consideration was in fact a 32-bit PDP-11. Most of the PDP-11 architecture is already independent of word length, and recent architectural studies had demonstrated that the PDP-11 is a bit-efficient architecture, even compared with mainframe architectures. In short, a PDP-11-like machine with an extended virtual address space would evidently be an attractive computer, today and tomorrow.

The most distinctive attribute of the PDP-11, and the basis for its architectural power, is the flexible way in which its registers can be used to form addresses. This flexibility permits the machine to be used effectively for many different types of computing, unlike most previous architectures, which tended to be good for one style of



1. New generation. The VAX-11/780 computer system consists of the central processing unit, the console subsystem, which serves as an operating system terminal and system or diagnostic console, the main memory subsystem, and the I/O subsystem. All major hardware components, implemented by Schottky TTL and standard MOS memory devices, are connected through the SBI, an internal synchronous bus.

processing but poor for another. For example, a design that has many central registers but not stack-like characteristics is good for scientific calculation but poor for complex subroutine structures. Conversely, a machine designed around a stack architecture is good for program control but inefficient for intensive calculation. But the PDP-11 is able to take on either set of attributes, and others, whenever a task demands it.

However, a 32-bit PDP-11 would have meant extending the register width to 32 bits but keeping the instruction formats and encodings unchanged, and this turned out to be an impossibility. The idea would have been a machine that could execute existing PDP-11 machine code intermingled with 32-bit code that made full use of the 4-gigabyte virtual address space. But a careful examination of a 32-bit version of the PDP-11 uncovered some unsurmountable obstacles.

There turn out to be many ways in which a programmer can implicitly design the address length into a program. For example, before control is passed to a subroutine, parameters may be pushed onto the stack. The subroutine call itself leaves the return address on the top of the stack. Within the subroutine, the parameters are accessed with respect to a known displacement from the top of the stack. But unfortunately, changing the address length from 2 to 4 bytes makes these known offsets invalid. This and many similar problems ruled out the possibility of executing 16-bit code unchanged in a larger address space, or of automatically translating 16-bit programs into a 32-bit form.

Given the difficulties of directly extending the PDP-11 design to a 32-bit form, the next alternative was to see what improvements could be gained by a bit-level-in-

compatible, but otherwise highly similar, design. The result was the VAX-11, a substantially better design that, though not precisely like the PDP-11, is "culturally compatible" with it. Hardware and software were also developed that permit a large subset of existing PDP-11 programs to execute without any changes on a VAX-11 system, as described earlier.

Architectures and word lengths

The description of the VAX-11 as a 32-bit minicomputer and the PDP-11 as a 16-bit minicomputer implies that the essential difference between them is their word length. But any significant difference in the architectures would presumably be measurable in terms of their comparative bit efficiencies on important applications. As it turns out, the bit efficiency of the PDP-11 is excellent, and in most respects the PDP-11 is not restricted to a 16-bit word length.

The problem is that the term "word length" has too many meanings to be useful without qualification. In a typical computer system, many different word lengths can be identified. In this context, therefore, it is necessary to eliminate from consideration the word lengths that represent engineering decisions for specific implementations and to consider only those that are intrinsic to an architecture and affect all its implementations in the family.

Instruction length is a possible candidate here. But both the PDP-11 and the VAX-11 have instructions of variable length, ranging from 16 to 48 bits and 8 to 296 bits respectively. In both cases, the variable-length instruction format offers better bit efficiency than an equal-length format because common instructions can

The importance of bit efficiency

A good architecture is reflected in a computer's static and dynamic bit efficiency. Bit efficiency is a quantifiable measure of how well the investment in the computer system's components pays off in application-level throughput. In other words, if two systems are built with the same technology and the same complexity, then the one with the greater bit efficiency will be more cost-effective (assuming that the bit-efficient instructions can still be rapidly decoded and executed by the processor).

Static bit efficiency is the relative size of a program compared with the size of a program coded for an architecture defined as a standard. A good static bit efficiency reduces the requirements for central memory and program file storage and streamlines the tasks involved in program-moving overhead, such as initial program loading, fetching overlays, paging, or swapping.

Dynamic bit efficiency is a comparative measure of how many program bits must be fetched from memory to the processor to execute a program. If all machine instructions were used with the same frequency, then static and dynamic bit efficiency would be the same. In practice, some instructions and data types occur often and others occur rarely. Good dynamic bit efficiency reflects the fact that the most frequent instructions (such as loop control

instructions) have particularly good encoding.

The ideal way to compare bit efficiencies would be to take a specific set of application programs and measure their actual bit efficiencies on different architectures. Unfortunately, such an approach is impractical for a computer vendor because customers have many disparate applications and many architectures of interest are hypothetical.

Fortunately, there are ways to characterize typical applications. Those coded in common high-level languages, such as Fortran, Cobol, and Basic, may be related to studies of typical program behavior, which show that in each of these languages different statements and data types have a characteristic frequency of occurrence. With these statistics and with an understanding of the machine code generated for each common statement, it is possible to estimate the bit efficiency of real or hypothetical architectures.

Bit efficiency is a good general test of architectural effectiveness, since it diminishes with any difficulty in machine-level programming or compiler code generation. Good static bit efficiency reflects effective use of system components; good dynamic bit efficiency reflects effective use of memory system bandwidth.

have shorter encodings. No architecturally useful definition of word length can be derived from instruction length, therefore.

Both the PDP-11 and VAX-11 are byte-address machines, since all data types are addressed in main memory by the byte address at the beginning of the data item, regardless of whether the data is a 1-byte character or an 8-byte double-precision, floating-point number. So memory addressing is also no help in defining the architectural difference between the machines.

But the PDP-11 has 16-bit general registers, whereas the VAX-11 has 32-bit general registers. In both architectures the registers can be used for arithmetic on data items that are shorter than the register size (8-bit integers on the PDP-11, 8- and 16-bit integers on VAX-11) or can be used in multiples for data items that are longer than the register size. Register length as used in arithmetic is not an invariable word length, therefore.

Register length

However, register length as used in instruction address formation is another matter: it does define the essential difference between the PDP-11 and the VAX-11, since it determines the size of logical storage that a program can instantaneously address—the size of the virtual address space. Thus the PDP-11's 16-bit byte address creates a virtual address space of 65,536 bytes, whereas the VAX-11's 32-bit byte address creates more than 4 billion bytes of virtual address space. What's more, virtual address space limitations can affect bit efficiency.

At the time the PDP-11 was designed, it seemed unlikely that any minicomputer would need more than 65 kilobytes of *physical* (as opposed to virtual) memory. In retrospect, the designers realize they failed to anticipate how rapidly central memory costs would decline.

Early in the evolution of the PDP-11 family, hardware memory mapping was added to the top-range machines. PDP-11 mapping logically divides the 64-K virtual address space into eight 8-K pages, each of which can be located independently in physical memory and protected independently (Fig. 2).

The addition of mapping offered two major benefits:

- It permitted the design of multiprogramming software systems in which a user program is prevented from damaging another user program or the operating system code, since each program can address only those parts of central memory allocated to it.

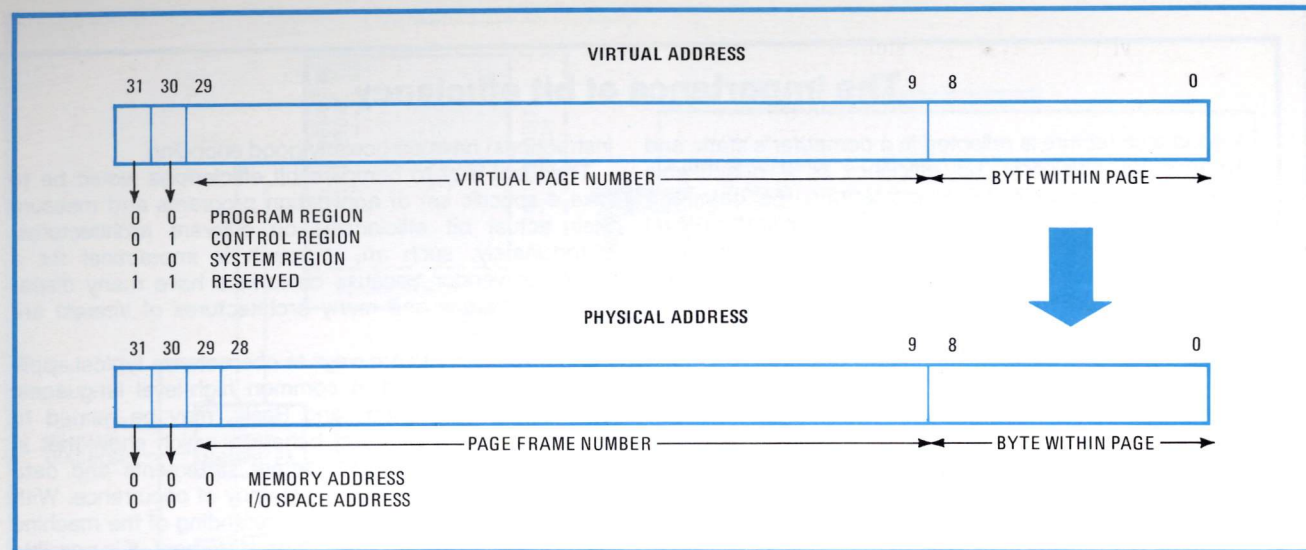
- It permitted the design of configurations with more than 65 kilobytes of physical memory, since the mapping hardware can translate 16-bit addresses into physical memory addresses of arbitrary length. Thus a PDP-11/70 may have in excess of 4 million bytes of central memory by developing 22-bit addresses.

For most applications, the remapping overhead is insignificant. But there are calculation applications in which it induces noticeable bit inefficiency. Also, although today most minicomputer programs and their data fit naturally into a 65-kilobyte address space, the trend to larger central memories will surely lead to larger program sizes as well.

The need for large address space

Consequently, though the immense marketplace success of the PDP-11 and other minicomputer architectures demonstrates that limited address space has not so far been felt as a restriction, it might become one in the future. Since the need for large address space is felt first in large configurations, the VAX-11/780 was designed as a top-of-the-line minicomputer.

The fundamental need of VAX-11 was to solve the



2. Address formats. Virtual addresses are 32 bits long in format shown at top. The virtual address space (memory space as it appears to a given process) is mapped onto the physical address space (lower format), which is the actual memory space existing in hardware. Memory-management logic controls the mapping and maintains in physical memory only portions of virtual memory being used.

addressing problem. Given the byte orientation of the PDP-11 and the need to do address arithmetic conveniently, the obvious address lengths to explore were 24 bits (16+8), 32 bits (16+8+8), etc.

Memory costs are roughly halving each year, and 1 more bit of physical address per year is needed. A 24-bit virtual address seemed too small. For VAX-11 the choice was 32 bits, representing a 4-gigabyte virtual address space (see Fig. 2 again). The next decision was to make this address space linearly addressable, meaning that there would be no further segmentation (many mainframes offer comparably large virtual address spaces but break them into many segments). This is large by any standard today, including mainframes, and should also allow a good decade of growth before the size of typical physical memories comes anywhere near the virtual address space. The very large, linear, virtual address space will also permit flexible evolution in software system design techniques, should that prove needed, for example, in advanced file or data-base management system designs.

Picking the virtual address size was a key decision, but a simple one compared with the total question of virtual memory design. Other issues included:

- Feasibility. It had to be possible to implement without adding a large cost penalty to the processor.
- Functionality. It had to be able to support sophisticated data-processing applications.
- Efficient use of control memory. The VAX-11 architecture had to exhibit a wide range of system performance without requiring large amounts of central memory for control tables.
- System efficiency. A virtual memory design would have to work well in real-time applications, yet be able to switch rapidly from one program to another while responding to external requests.

A 32-bit virtual memory design is totally different from a 16-bit one. For example, mapping a 65-kilobyte PDP-11 program consists of initializing eight internal processor registers. In contrast, mapping a 4-gigabyte

address space would mean specifying 8 million page relocations (on VAX-11, each page or separately mappable unit of virtual memory is 512 bytes). Obviously, no VAX-11 processor will have 8 million internal relocation registers!

Similarly, the operating system on a PDP-11 does little in the way of virtual memory management. Because programs are relatively small, they are typically swapped in and out in their entirety. But the large virtual address space on VAX-11 encourages programmers to build large, logically connected programs that may well be much bigger than the physical memory of the system. For example, a large computation that had previously been structured in overlays will simply be a big program on VAX-11. But making the virtual memory application programmer means that the operating system has to be much more active in deciding which pieces of an application program should be kept in central memory and which should be kept on disk-backing storage (called working-set management). This in turn means that the mapping mechanism must be efficiently controlled by operating system code and must provide as much useful information about the dynamic usage of virtual memory as possible.

Sharing virtual memory space

A key feature of the VAX-11 virtual memory design, which leads to efficient use of central memory and low overhead during execution, is the way in which the operating system shares virtual memory space with user processes. The entire 4-gigabyte virtual address space is logically divided into halves. The user process is limited to the use of the bottom half, and most of the operating-system code resides in the top half (Fig. 3). (Remember that this is just virtual memory; the decision about what virtual memory is made resident in central memory, including operating-system code, is made dynamically on the basis of actual need.) The mapping of each user process may be unique, or user processes may share program and data pages with one another, but in either

case the same virtual operating system is mapped with each user process.

Putting the operating system into a single address space rather than having pieces of the system code in multiple address spaces minimizes the need for system mapping control tables and makes intersystem communication more efficient. Having the operating system share the virtual address space of each user process simplifies requesting services from the operating system. The high-speed processor translation caches, which store the most recently used mapping translations, treat system and user mapping separately so that system mapping translations stay in the cache when the operating system switches to another user process, but the user translations are flushed out.

In a simple memory scheme, it would be risky to put the user programs in the same address space as the operating system, since their malfunctions could affect the system operation. In VAX-11, the system is totally protected from this by a separate access control mechanism. The processor executes in one of four modes:

- Kernel, for interrupt processing, physical I/O control, processor scheduling, and the like.
- Executive, for file management and similar functions.
- Supervisor, for functions such as interactive command processing.
- User, in which user programs are executed.

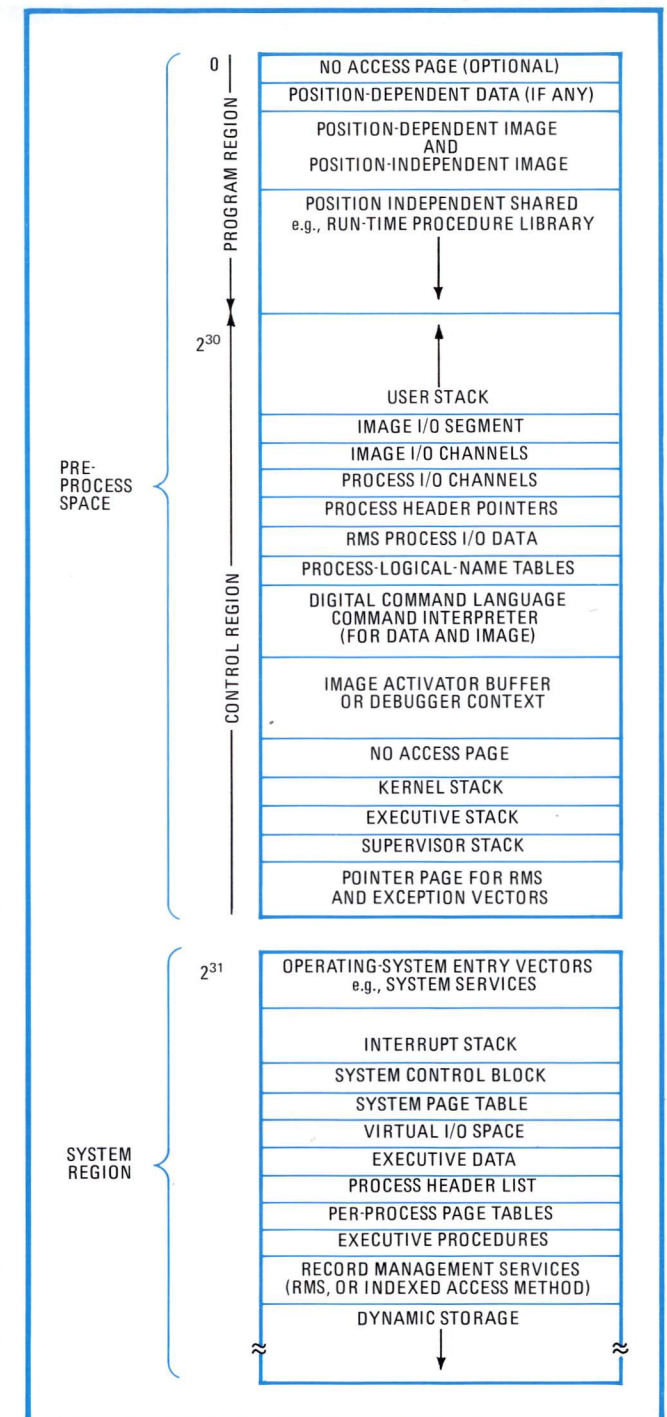
The accessibility of each page in the virtual memory space (whether the page can be read, written, or both) can be controlled for each of the processor execution modes. Thus the operating system can keep critical data in the user's address space and can access that data freely during execution of an operating system service (during which the processor runs in a more privileged mode), and yet the user's program may be restricted from reading the data, if that is inappropriate.

Compatibility mode

The VAX-11 user mode can be put into a compatibility mode, which makes it capable of executing many PDP-11 programs often faster than the PDP-11/70, the top-of-the-line PDP-11. The efficiency of PDP-11 emulation is due to the strong cultural compatibility between VAX-11 and the PDP-11. Thus a processor designed to perform VAX-11 instructions efficiently can also perform PDP-11 instructions well. Instruction execution of the VAX-11/780 is implemented with microcode (as is true of most computers today); PDP-11 compatibility-mode emulation was primarily implemented with a 10% increment of microcode.

This VAX-11/PDP-11 compatibility mode is worth exploring in some detail. Some earlier "compatible" emulation modes required that the computer be used only in one mode at a time. But although 16- and 32-bit code cannot be freely mingled within a single program on the VAX-11, compatibility-mode jobs and native jobs can run at the same time, sharing the resources of the VAX/VMS multiprogramming system. The two kinds of jobs can even cooperate with each other by exchanging messages or by sharing files.

Emulation modes have also been used in the past in the place of software development for the new architec-



3. Allocation. Virtual address space is divided into halves. The lower address locations are limited to user processes, the upper to operating-system code. User processes may share program and data pages, but virtual operating system is always the same.

ture. However, in the case of VAX-11, the power of the enormous virtual address space and new instructions were intrinsic to the value of the system. So there seemed to be little value in a hardware-supported compatibility mode that would execute a complete PDP-11 operating system. Instead, as already indicated, the compatibility mode is limited to user-mode programs. Those operating-system utilities that are insensitive to the size of the address space have been taken from the earlier family's

DATA TYPES HANDLED BY VAX-11

Data type	Size	Range (decimal)	
Integer		Signed	Unsigned
Byte	8 bits	-128 to +127	0 to 255
Word	16 bits	-32,768 to +32,767	0 to 65,535
Long word	32 bits	-2^{31} to $+2^{31}-1$	0 to $2^{32}-1$
Quad word	64 bits	-2^{63} to $+2^{63}-1$	0 to $2^{64}-1$
Floating point		$\pm 2.9 \times 10^{-37}$ to 1.7×10^{38}	
Floating	32 bits	approximately 7-decimal-digit precision	
Double floating	64 bits	approximately 16-decimal-digit precision	
Packed decimal string	0 to 16 bytes (31 digits)	numeric, two digits per byte sign in low half of last byte	
Character string	0 to 65,535 bytes	1 character per byte	
Variable-length bit field	0 to 32 bits	dependent on interpretation	

RSX-11M operating system and execute in compatibility mode transparently to the user.

The application migration executive is a subroutine package provided with VAX/VMS that emulates RSX-11M operating support for PDP-11 programs running in the VAX-11/PDP-11 compatibility mode. VAX/VMS has been designed to transfer control to the AME within 50 microseconds on the VAX-11/780 when a PDP-11 compatibility-mode program requests operating-system services. The AME executes as a VAX-11 program, in a 32-bit address space that includes the 16-bit address space of the PDP-11 program. The AME determines which RSX-11M system call is being requested by a PDP-11 program, translates it into VAX/VMS format, and issues the request to VAX/VMS. When control returns to the AME, it translates the results into RSX-11M format, stores the result in the compatibility-mode program data, and then returns control to the PDP-11 program via a VAX/VMS service.

Translators

Use of the AME permits a large collection of the RSX-11M programs to run unchanged on VAX-11 systems under VAX/VMS. Although the AME translates RSX-11M system calls, a similar program could be written to translate calls of other PDP-11 operating systems. A single VAX/VMS system could, theoretically, have translators for multiple PDP-11 operating systems.

The efficiency of compatibility-mode program execution under such a translator depends on how heavily the program uses operating-system facilities and how different the emulated operating system was from RSX-11M and VAX/VMS. Although the translation adds some overhead, the typical VAX/VMS service is faster than RSX-11M (run on a PDP-11/70) because of the increased functionality of the VAX-11. On balance, emulated PDP-11 programs run about as fast as they would in the PDP-11/70 under RSX-11M.

The PDP-11 was designed with 8 general-purpose registers. Since then, the cost increment of additional processor registers has gone down dramatically, and the

VAX-11 was given 16. Apart from cost, the penalties for additional registers are a need for extra system overhead to perform context switching and a reduction in bit-efficiency, since more bits are required to address a register. Nevertheless, these extra registers do provide better compiler optimization of generated code, lower overhead in subroutine usage, and efficient design of complex instructions.

The strength of the PDP-11 architecture is its inclusion of the best features of stack, multiple-register, and memory-to-memory designs because of the versatile way in which its general registers can be used to develop addresses. VAX-11 added to these addressing modes to increase the efficiency of program indexing into those tables that list multiple-byte data items like 4-byte floating-point values or 8-byte integers.

In the last decade the processing capacity of minicomputer systems has increased to the point where they are patently unsuited for very few applications. To support efficiently all likely forms of processing, new data types (forms of data for which processor instructions exist) were added to VAX-11, as shown in the table. VAX-11 implicitly does 32-bit address arithmetic, and instructions were added for explicit 32-bit integer arithmetic and Boolean logic.

Decimal arithmetic

Thirdly, VAX-11 permits arithmetic to be done directly in decimal form, instead of requiring that it be converted to binary form, to ensure that full data precision is retained, and because such data is more often moved intact between data records than used for calculation. Still other new data types are test string manipulation (where strings of characters can be moved, translated, and searched with specific instructions) and a complex editing instruction (to provide for the kinds of manipulations common in generating the typical data-processing report—for example, editing out leading zeros or adding a dollar sign). On VAX-11, direct processor support for 1- to 32-bit data fields has been implemented, increasing the bit efficiency of critical

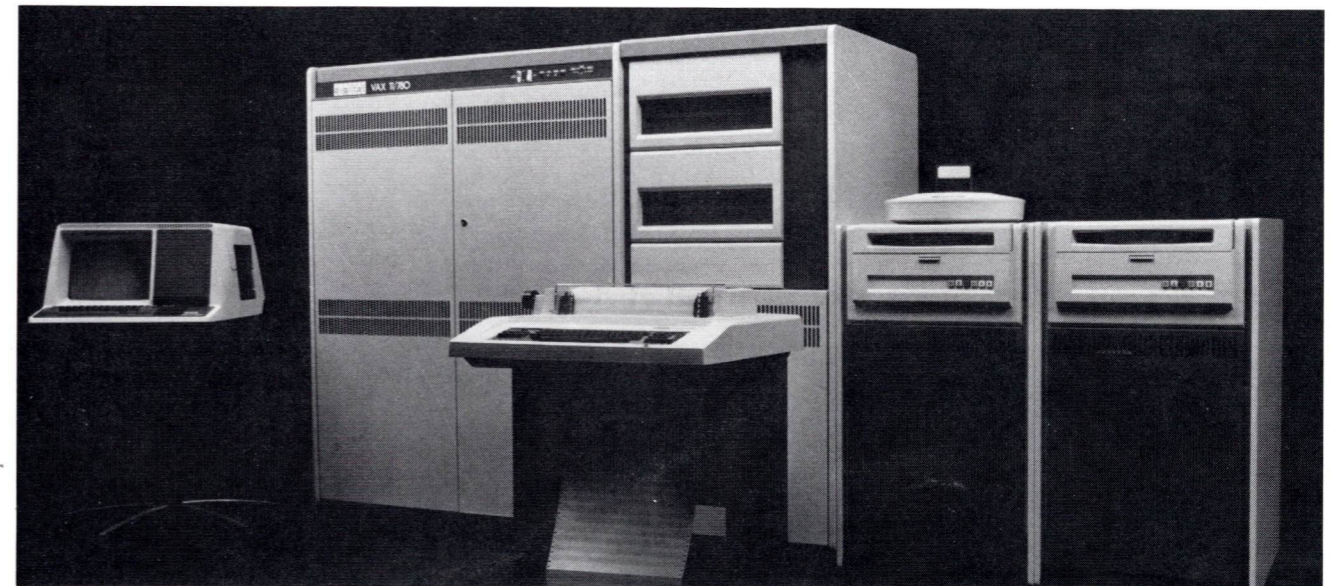
The economics of compatibility

The last thing a computer user wants to do is to rewrite an existing program for a new hardware design. He would rather develop new software for all the profitable new applications it opens up. After all, if the hardware is costing him less, the cost of programming is still as high as ever. Worse yet, it may even be rising, since the number of computers to be programmed appears to be growing faster than the population of skilled programmers.

In view of this, a major design focus for the VAX-11 architecture was compatibility with the PDP-11. Accurate statistics are not available, but if as little as \$20,000 has been spent for software for each of the 50,000 PDP-11s produced this far, then the total investment is \$1 billion. Surely, this is a conservative estimate when it is remem-

bered that \$20,000 buys only a small amount of code.

The size of the entire existing investment in software is incomparably larger, and the need to preserve it might already have halted innovation in the computer industry were it not for the phenomenal rapidity with which computer technology is evolving. The new markets and applications continuously being opened up force architectural changes that compel some level of innovation even at the price of a devalued software investment. Nevertheless, the point has seemingly been reached at which no new computer—mainframe, mini or micro—can be designed without careful examination of the compatibility issue. Barring some remarkable breakthrough in software engineering, compatibility will continue to grow in importance.



Compatibility. Design goal of the VAX-11/780 32-bit minicomputer system shown here was to preserve compatibility with the software developed for the existing PDP-11 family while anticipating future needs. It features greatly extended virtual address space.

operating-system code and like programs. Since field-bit position is specified by a 32-bit integer, very large (512-megabyte) structures can be linearly bit-addressed.

Although most of the added instructions were in support of the data types listed above, many other special instructions were added for operating system support, user programming support, and specific computation needs. For example, an instruction, POLY, has been provided to compute polynomial equations of the form:

$$y = C_1 + C_2x + C_3x^2 + \dots$$

in a single instruction, with the loop overhead handled in microcode. This instruction substantially speeds up the calculation of standard numerical approximation, such as the calculation of sine and cosine functions within a Fortran run-time library.

Future minicomputers

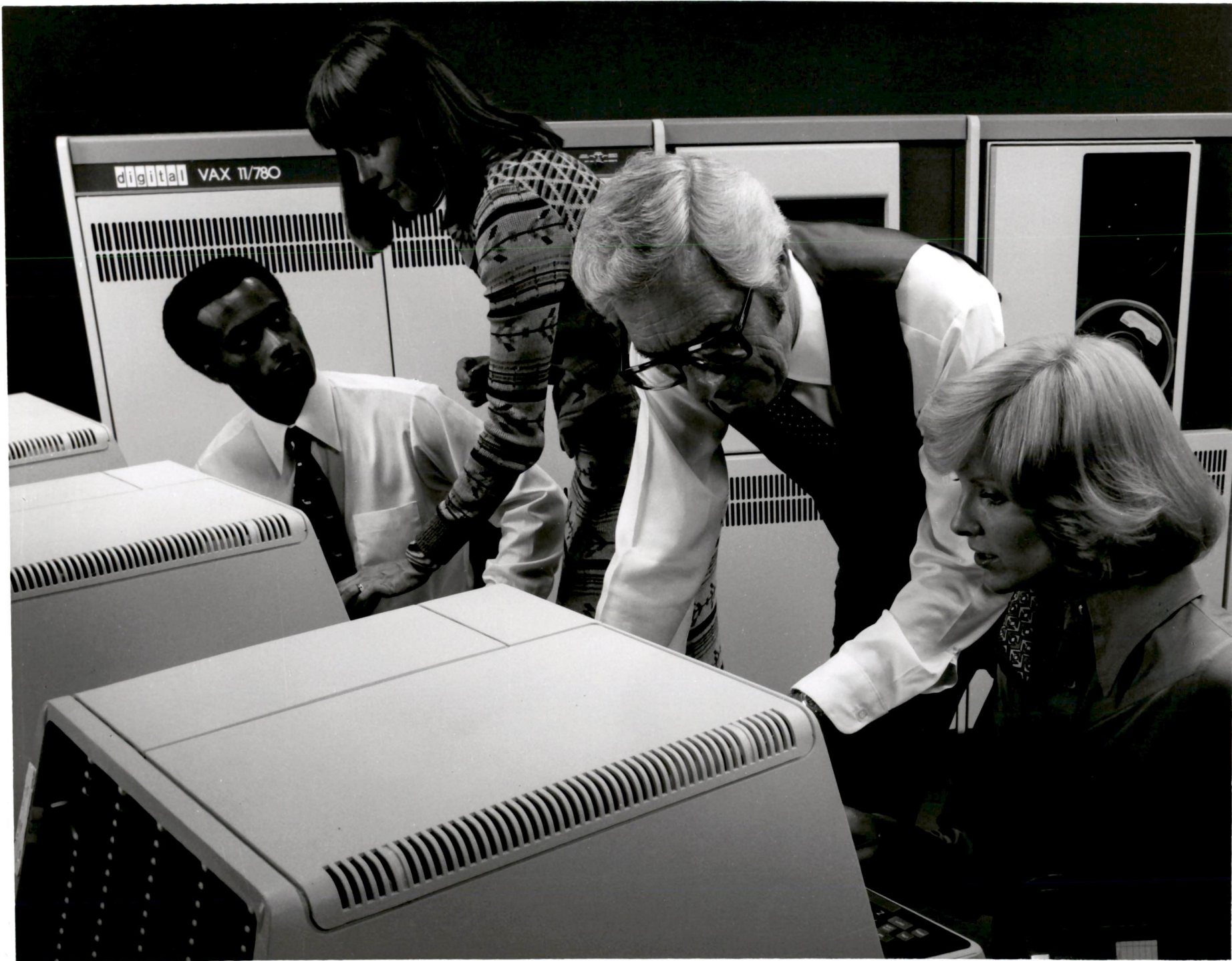
The implications of all these capabilities for the future of the VAX-11 minicomputer family can readily be assessed. The yardstick may be inferred from the appar-

ently constant rates of improvements in base technologies like semiconductors and magnetics and from computer designers' rules of thumb. For example, in a typical system the number of bytes of central memory is about equal to the number of instructions per second by the central processor. Such a system must also be capable of performing 1 bit of I/O for each instruction executed (these rules are sometimes attributed to Gene Amdahl). For any year in the near future, the technologies' price prediction for computer subsystems may be combined with the designers' predictions of the appearance of a balanced system, the constant price definition of minicomputers applied, and the range of expectable system configurations thus delimited.

Following such logic, a minicomputer priced at \$50,000 in the early 1980s should look much like today's mainframe in gross capability, having on the order of a million bytes of central memory, hundreds of millions of bytes of disk storage, and so on. That prediction, made some years ago, led to the VAX-11 design project and the development of the VAX-11/780. □

digital

DIGITAL EQUIPMENT CORPORATION, Corporate Headquarters: Maynard, Massachusetts 01754, Telephone (617) 897-5111—SALES AND SERVICE OFFICES; UNITED STATES—ALABAMA, Birmingham, Huntsville • ARIZONA, Phoenix, Tucson • CALIFORNIA, El Segundo, Oakland, Sacramento, San Diego, San Francisco, Santa Ana, Santa Barbara, Santa Clara • COLORADO, Denver • CONNECTICUT, Fairfield, Meriden • FLORIDA, Miami, Orlando, Tampa • GEORGIA, Atlanta • HAWAII, Honolulu • ILLINOIS, Chicago, Peoria, Rolling Meadows • INDIANA, Indianapolis • IOWA, Bettendorf • KENTUCKY, Louisville • LOUISIANA, New Orleans • MARYLAND, Baltimore, Odenton • MASSACHUSETTS, Springfield, Waltham • MICHIGAN, Detroit • MINNESOTA, Minneapolis • MISSOURI, Kansas City, St. Louis • NEBRASKA, Omaha • NEW HAMPSHIRE, Manchester • NEW JERSEY, Cherry Hill, Fairfield, Princeton, Somerset • NEW MEXICO, Albuquerque, Los Alamos • NEW YORK, Albany, Buffalo, Long Island, Manhattan, Rochester, Syracuse, Westchester • NORTH CAROLINA, Chapel Hill, Charlotte • OHIO, Cincinnati, Cleveland, Columbus, Dayton • OKLAHOMA, Tulsa • OREGON, Portland • PENNSYLVANIA, Harrisburg, Philadelphia, Pittsburgh • RHODE ISLAND, Providence • SOUTH CAROLINA, Columbia • TENNESSEE, Knoxville, Nashville • TEXAS, Austin, Dallas, El Paso, Houston • UTAH, Salt Lake City • VERMONT, Burlington • VIRGINIA, Richmond • WASHINGTON, Seattle • WASHINGTON, D.C. • WEST VIRGINIA, Charleston • WISCONSIN, Milwaukee • INTERNATIONAL—ARGENTINA, Buenos Aires • AUSTRALIA, Adelaide, Brisbane, Canberra, Darwin, Hobart, Melbourne, Perth, Sydney, Tasmania • AUSTRIA, Vienna • BELGIUM, Brussels • BOLIVIA, La Paz • BRAZIL, Rio de Janeiro, Sao Paulo • CANADA, Calgary, Edmonton, Halifax, London, Montreal, Ottawa, Toronto, Vancouver, Winnipeg • CHILE, Santiago • DENMARK, Copenhagen • EGYPT, Cairo • FINLAND, Helsinki • FRANCE, Lyon, Paris • HONG KONG • INDIA, Bombay • INDONESIA, Djakarta • IRAN, Tehran • IRELAND, Dublin • ISRAEL, Tel Aviv • ITALY, Milan, Rome, Turin • JAPAN, Osaka, Tokyo • MALAYSIA, Kuala Lumpur • MEXICO, Mexico City • NETHERLANDS, Amsterdam, Hague, Utrecht • NEW ZEALAND, Auckland, Christchurch • NORTHERN IRELAND, Belfast • NORWAY, Oslo • PERU, Lima • PUERTO RICO, San Juan • SAUDI ARABIA, Jeddah • SCOTLAND, Edinburgh • SINGAPORE • SOUTH KOREA, Seoul • SPAIN, Madrid • SWEDEN, Gothenburg, Stockholm • SWITZERLAND, Geneva, Zurich • TAIWAN, Taipei • UNITED KINGDOM, Birmingham, Bristol, Ealing, Epsom, Leeds, Leicester, London, Manchester, Reading, Welwyn • VENEZUELA, Caracas • WEST GERMANY, Berlin, Cologne, Frankfurt, Hamburg, Hannover, Munich, Nurnberg, Stuttgart • YUGOSLAVIA, Belgrade, Ljubljana •



digital

EQUIPMENT
CORPORATION
MAYNARD, MASS. 01754

Digital Equipment's new VAX-11/780^(TM) 32-bit computer system supports up to 64 interactive users as well as multi-stream batch processing. The virtual memory operating system permits simultaneous processing of multiple large programs for scientific computation, simulation, commercial and control applications.

#####

For Further Information:
McLaren Harris
(617) 897-5111 Ext. 2857



digital

EQUIPMENT
CORPORATION
MAYNARD, MASS. 01754

Interactivity, reliability and maintainability are emphasized in the design of Digital's new VAX-11/780^(TM) computer system. The high-speed, multi-program system uses the 32-bit VAX-11 FORTRAN IV-Plus language as well as existing PDP-11 software including BASIC-Plus-2 and COBOL.

#####

For Further Information:
McLaren Harris
(617) 897-5111 Ext. 2857



digital


EQUIPMENT
CORPORATION
MAYNARD, MASS. 01754

Digital Equipment's new VAX-11/780^(TM) 32-bit computer system supports up to 64 interactive users as well as multi-stream batch processing. The virtual memory operating system permits simultaneous processing of multiple large programs for scientific computation, simulation, commercial and control applications.

#####

For Further Information:
McLaren Harris
(617) 897-5111 Ext. 2857

digital VAX 11/780



Part #: 674-2367AJ
Description: Lxtoran Engine 230-0P crankshaft
Number on hand - 27
Inventory lower limit - 6
Number of 674-2367AJ required: 1
Charge to: John G. Coarson/acct 234323
Part #: 670-23488K
Description: Lockwashers for Lxtoran Engine 230-0P
Number on hand - 2168
Inventory lower limit - 127
Number of 670-23488K required: 21
Charge to:

digital EQUIPMENT
CORPORATION
MAYNARD, MASS. 01754

BOSTON, Mass. -- October 25, 1977 -- "Probably the most significant interactive computer of the decade" was the description given by Digital Equipment Corporation President Kenneth H. Olsen of his firm's new VAX-11/780^(TM) computer unveiled today at a press conference prior to the company's annual meeting of shareholders in Boston. The company says the new 32-bit computer system combines the full power and performance of conventional large computers with the interactive strength, flexibility and low cost of a minicomputer. The multi-user system is said to be the industry's fastest computer system priced under \$200,000. Anticipated early markets for the computer are original equipment manufacturers (OEMs) and end-users in industrial, commercial, research and academic environments.

#####

For Further Information:
Edward J. Canty
(617) 897-5111 Ext. 2268



digital

EQUIPMENT
CORPORATION
MAYNARD, MASS. 01754

Digital's new 32-bit computer features 4 billion byte virtual addressing, a virtual memory operating system, and extended compilers. Called VAX-11/780^(TM), it is a high-performance, multi-user, multi-program system specifically designed for interactive applications. As the fastest-operating 32-bit computer system priced below \$200,000, VAX-11/780 is suitable for scientific, commercial, control, and simulation applications. System prices start at \$130,000 with deliveries scheduled to begin in early 1978.

#####

For Further Information:
Stephen A. Kallis, Jr.
(617) 897-5111 Ext. 2777