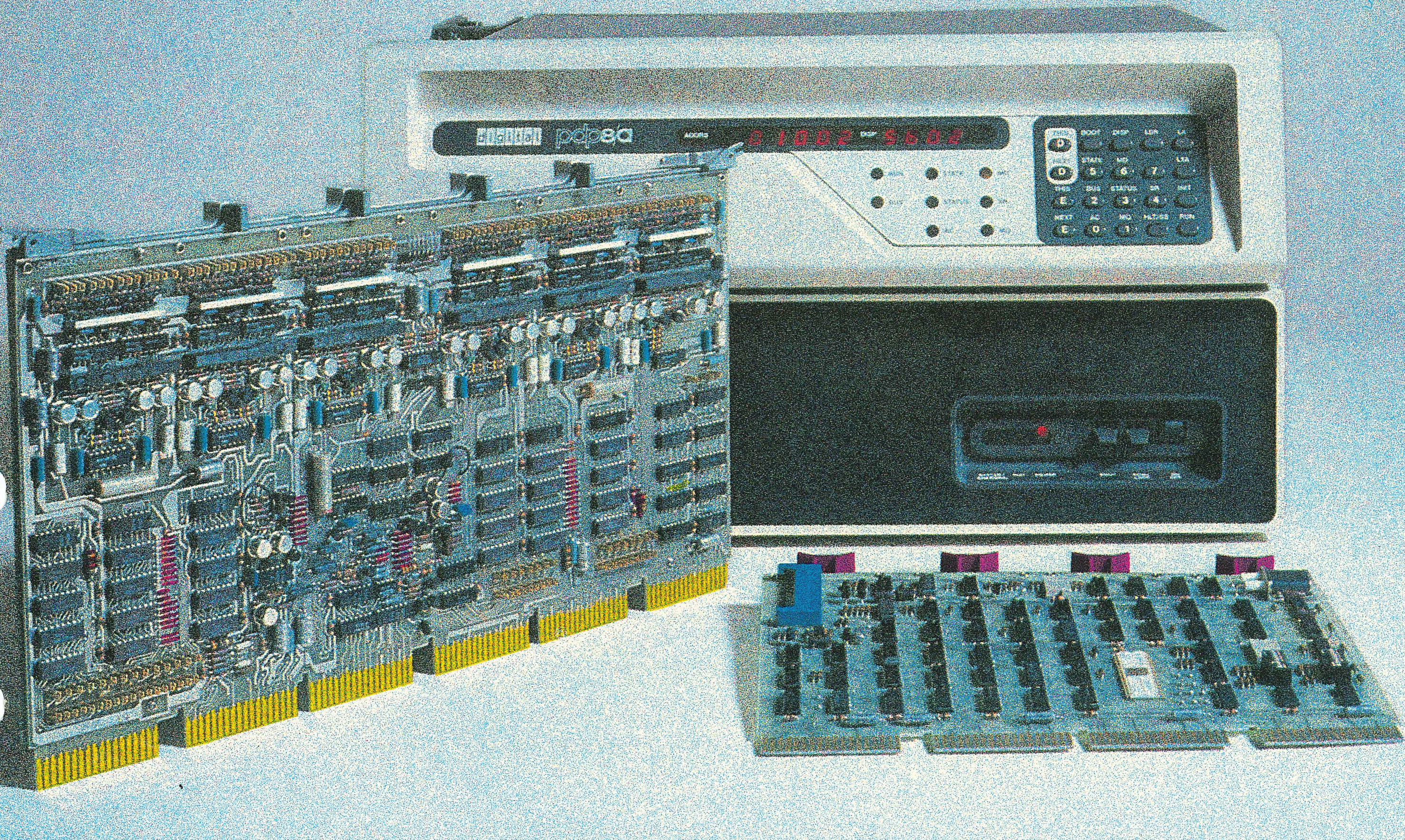
digital

OS/8 and RTS/8

Two good reasons why so many profit-minded customers use our 8/A Series Minicomputers



digital
DEC
CASSETTE
decpack
1100 BPI
decpack
1100 BPI
LINCTAPE
digital
DIGITAL EQUIPMENT CORPORATION
REEL NO.
DATE
DIGITAL EQUIPMENT CORPORATION
DECTAPE
DECTAPE
digital
digital

OS/8 FOR SOFTWARE DEVELOPMENT USE

When you invest in a computer system, you're not just interested in the hardware alone—you're interested in its ability to perform a specific function with minimum effort, the positive ways it can contribute to your overall profitability. Meet DIGITAL's OS/8—a powerful, versatile operating system, offering many of the programming advantages previously available only on much larger and more expensive computer systems. Designed for the PDP-8/A Minicomputer Series, OS/8 is based on a solid foundation of proven performance. Many of its design concepts have evolved from our experience with more than 30,000 hard working members of the PDP-8 family that are installed today.

OS/8 FEATURES

Modular System Design
Through the design modularity of OS/8, the original PDP-8/A installation can expand to handle increasing work loads and new user-developed applications. OS/8 together with an 8/A Series central processor unit supports a wide range of system configurations. Memory can be expanded from 8K to 32K words.

Extensive Peripheral Support
OS/8 supports many different peripherals: floppy and cartridge disks, cassettes, card readers, line printers, papertape devices, as well as data communications terminals.

Ease of Use
OS/8 provides the development programmer with a complete, logical interface to program and file structures. All data files or executable programs are stored in one or more system libraries where they may be accessed for loading, modification, or execution by simple keyboard commands.

Dynamic Resource Allocation
OS/8 maintains the current status of system resources to determine the combination of peripherals, input files, output files and run-time options to be used during program execution.

Flexible Language Processing
OS/8 supports a variety of higher-level languages: BASIC, FORTRAN IV and Industrial BASIC. In addition, the highly efficient assembler language PAL-8 can be used when speed or program size is critical.

Complete Device Independence
OS/8 allows programs to be coded in a format without regard for the characteristics of a particular I/O device.

Increased Availability of Main Memory
OS/8 maximizes utilization of available mass storage because the resident portion of the operating system requires only 256 words of memory. Non-resident portions of the system are swapped into memory from the system peripheral device automatically, as required. In addition, the operating system provides the capability of dividing large programs into smaller segments.

Maximized Programming and Run-Time Efficiency
You can run programs written entirely in one language or made up of segments in different languages. For example, code input/output routines can be run in assembly language to reduce handling time, while mathematical algorithms can be written in FORTRAN IV to save programming time.

All these features add up to more performance, more speed, more convenience and more economy in solving your particular problem. Such is the purpose of the OS/8 Operating System running on a PDP-8/A Minicomputer. A combination of comprehensive system software and powerful hardware that can reduce your operating expenses and contribute to your overall profitability.

OS/8 SOFTWARE COMPONENTS

OS/8 encompasses many automatic and simplified programming aids. These aids include the following system components:

Keyboard Monitor
Provides communication between the user and the OS/8 executive routines by accepting commands from the terminal keyboard.

Command Decoder
Allows the user to communicate with the system library program by accepting a command string from the keyboard, indicating input/output devices and files.

Device Handlers
Transfer data to and from peripheral devices. Device handlers are an integral part of the system that allow OS/8 to perform device independent I/O on as many as 15 different peripherals at a time.

User Service Routine
Controls the directory operations for the OS/8 system via standard subroutine calls.

OS/8 SOFTWARE LIBRARY

DIGITAL's PDP-8/A Minicomputer Series has a comprehensive software library fully integrated under OS/8. Within the library are:

- Editing programs
- Assembly programs
- Language compilers
- Debugging programs
- File maintenance routines
- Special purpose programs

Editing Programs

The two PDP-8/A Series editing programs, EDIT and TECO, are a collection of routines used to create and maintain user libraries.

EDIT—EDIT is a line-oriented, file inspection subsystem that allows the user at a terminal or keyboard to enter text into a computer, edit it, store it and retrieve it. The subsystem is especially useful for programs that often need modification. Among EDIT's features are multiple editing commands that permit programs to be typed, modified and checked quickly and simply using simple keyboard commands. A character search command allows part of a line to be changed without changing the entire line. A character string search command provides the facility for locating and examining the contents of a permanent file.

TECO—TECO is a powerful, character-oriented program editor that will handle any form of ASCII text. With only a few simple keyboard commands, the user can store, maintain and retrieve permanent data files. These files may include listings, manuscript, or business data. Easy to learn and simple to use, TECO features:

- Extensive character processing capabilities—Since TECO is character oriented rather than line oriented, there are no extra line numbers. In addition, it is not necessary to replace an entire line in order to change a single character.
- Versatile macro commands—TECO's editing-macro command structure allows the user to make repetitive changes throughout a large program or data file with only a single command.
- Efficient command string management—TECO enables command strings to be saved, a significant feature that saves time on subsequent or repetitive program executions.

Assemblers

Two assemblers, PAL-8 and SABR, are available with all PDP-8/A Minicomputers. They are highly efficient and responsive tools that take maximum advantage of the PDP-8/A Series hardware, while imparting all of the features of a higher-level language.

PAL-8—PAL-8 is a two-pass language assembler that provides the programmer with the convenience of coding directly in machine-oriented symbolic instructions. It features:

- Conditional assembly—Enables the same source program to produce different binaries for different purposes.
- Paginated listings, page headings and page numbering—Improves program documentation.
- Binary-symbol table search—Performs a search for all like binary words and lists the memory address.

SABR—SABR (Symbolic Assembler for Binary Relocatable programs) is an advanced, one-pass assembler producing relocatable binary code with automatically-generated page and field linkages. It supports an extensive list of pseudo-operations which provide, among other facilities, external subroutine calling with argument passing and conditional assembly.

A SABR program may call routines from a large library of subroutines and functions. These are loaded together with the SABR program by the Linking Loader. In an optional second pass, SABR produces an octal/symbolic listing of assembled programs. SABR has the following features:

- Binary relocatable output—Programs can be divided into logical entities. Each division can be written, assembled and loaded independently.
- An extensive library of mathematical, utility and input/output subroutines —Provides optimized code and saves programming, debugging and execution time.
- Generates links to off-page references—Saves time and effort, as programs can be written without concern for page boundaries.
- From 8K to 32K of memory—Programs are easily expandable.
- SABR is fully integrated into the OS/8 operating system—Saves programming, debugging and execution time.

FORTRAN IV

FORTRAN IV has the following features:

- Full, ANSI-standard FORTRAN IV—Existing programs run with little or no modification. New software is not machine-dependent. Requires little learning time.
- Hardware independence—The run-time system can use from 8K to 32K, and automatically configures itself to take advantage of an extended arithmetic element or floating point processor.
- Interrupt-driven run-time system—The system can perform specialized background functions, such as refreshing a scope display, while the processor is executing an I/O transfer.
- Dynamic run-time overlays—Provides 10 times the existing size of memory, so that programs requiring over 300K words of core storage can be structured to run on a 32K PDP-8.
- Direct access I/O—Random blocks in very large files can be accessed and updated independently.
- Generalized array subscripting—Data is more accessible; further processing is easier.
- Every program is fully I/O-device independent—Standard device specifications can be used, or unit specifications can be freely assigned at load-time.
- Error diagnostics at compile and run-time—Makes the job of debugging easier and quicker.
- Mixed-mode arithmetic—Eliminates mixed-mode errors.
- Logical variables and operators—Expedites routine decision processing.
- Text manipulation—Enables FORTRAN IV to accept and store words of text, as well as numeric values, such as tabular data.
- FORTRAN-callable subroutines allow the user real-time access to standard laboratory peripherals such as clocks and A/D converters.

FORTRAN IV Plotter

FORTRAN IV drives an incremental plotter under control of the OS/8 operating system at up to 300 steps per second. The user has complete control over pen state (up/down), angle of plotting and overall size of plot. Diagrams and other drawings can be adapted to the user's specific needs.

BASIC

- Similar to Dartmouth College BASIC—Compatible with and as easily learned and applied as versions of Dartmouth BASIC, OS/8 BASIC offers extended operations and functions for the experienced programmer.
- 8K-32K of memory—Provides in 8K plenty of space for computational code or data storage. If more than 8K is available, it can be used to increase program size and/or data storage capability.
- Program chaining—Program size is virtually unlimited.
- Two-processor configurations—OS/8 BASIC runs with or without the Extended Arithmetic Element (EAE). Programs execute faster with EAE.
- Interactive editing—Provides for instant modification of a program.
- String operations—Expands the applications into such areas as text processing, formatting and command language interpretation.
- Core-image files—Compiled BASIC programs may be stored as save images and loaded into another OS/8 system. Programs created on one OS/8 configuration will run on any other OS/8 system having sufficient memory.
- Dynamic file I/O—There is no limit to the total number of files a program can access, as long as only four remain active simultaneously. Files and devices may be specified during program execution.
- Real time—The Lab-8/A software enables the user to solve a range of real-time problems using a higher level language. Lab-8/A contains a set of 12 functions which enable a user of OS/8 to utilize the A/D converter, display control, real-time programmable clock, and 12-channel buffered digital I/O.

OS/8 Industrial BASIC

This is an extension and modification of OS/8 BASIC which provides additional functions and statements to support real-time industrial acquisition and control.

Extended functions supported include:

- Analog I/O
- Digital I/O
- Loading and reading of counters
- Clock function
- Console commands

...Plus two new features:

- Industrial 14 Programmable Controller—OS/8 Industrial BASIC provides higher-level language monitoring and control of up to four Industrial 14/30 and 14/35 Programmable Controllers. Asynchronous serial line links permit control, testing and reporting operations for the Industrial 14. Programs for the Industrial 14 can be loaded from, read by and verified (via the serial line link) with Industrial BASIC files.
- Core-Only Run-Time System—A run-time system can be generated on the OS/8 host system for use in a memory-based target system. File-oriented operations are excluded from the target machine run-time system. The transportation medium from host to target system is either papertape or cassette.

Debugging Programs

ODT (Octal Debugging Technique)

- Invisibly co-resident with the user program—There's no need to allocate memory for a debugging package during development.
- Breakpoints can be set anywhere in a program—The user can trace the execution of a program step-by-step or even instruction-by-instruction. Execution proceeds normally up to and then after the breakpoints.
- Binary memory search mechanism—Specified areas of memory may be searched.

BITMAP

- Maps the memory requirements of any binary file—Vacant memory areas are easily identified, eliminating time spent on scanning source listings page-by-page to locate free core areas.
- Locates double- or multiple-memory register allocations—If a memory register is accidentally allocated to more than one binary word, BITMAP flags the register and indicates how many times its contents were specified.
- Maps multiple files simultaneously—Overlays may be mapped onto the program that they will load over. Modules may be mapped in sets, reflecting the manner in which they will be loaded. Programs can be compared, register-by-register, on the basis of storage allocations.

CREF (Cross-Reference Utility Program)

- Alphabetical cross-reference table—Produces a numbered assembly listing and then alphabetically lists each symbol and literal in the program and line number of every reference to it.
- Optional two-pass operation—Doubles the number of symbols that can be accommodated in a program.

BATCH

- Batch processing monitor—Lengthy jobs can be run on the computer during off hours and can be executed unattended.

LINCTAPE
DECTAPE
digital
DIGITAL EQUIPMENT CORPORATION
COPYRIGHT © 1969, 1972, 1974
DIGITAL EQUIPMENT CORPORATION

- Option spooling of output files—Execution is faster, less paper is consumed, and output can be dumped selectively from the spool files to any hard-copy device at a later time.
- Full-monitor command set—With BATCH, only a single command is needed to carry out a complicated procedure ordinarily requiring several monitor calls, extensive I/O specifications and numerous commands. Operator intervention is not required.

  All the commands for Keyboard Monitor, Concise Command Language and Command Decoder are available in BATCH.

OS/8 APPLICATION SOFTWARE

Lab Application Packages

The following minimum components are required to operate the Lab Application Programs.

- PDP-8/E or PDP-8/A Series central processor with 8K of memory and an ASCII-compatible terminal.
- One auxiliary storage system:
  Dual-drive RX8 Floppy Disk.
  Single-drive removable-cartridge RK8 DECpack with a dual-drive cassette or high-speed papertape reader/punch.
- A/D converter and multiplexer.
- Programmable real-time clock with Schmitt triggers.
- Point-plot display controller.
- OS/8 binary license.

Supported options:

- Extended arithmetic element.
- Additional DECtape or disk drivers.
- XY analog plotter.

Advanced Signal Averager

The Advanced Signal Averager is distinguished by its ability to back (presync) average, sort or edit averages contingent on some external event, sample at two different rates, and calculate the raw statistics needed for standard deviation confidence limits and trends. It provides statistical confidence limits around the average, standard deviation for each data point in the average, and provides first-order trend read out. It features 1024 points averaging in double precision.

Histogram Application Program

This program provides a powerful tool for the neuro-physiologist (and others) to investigate both spontaneous and stimulated spike train activity through the generation of three histogram types: time-interval, post-stimulus, and latency.

Auto- and Cross-Correlation Application Program

The Auto- and Cross-Correlation Package is designed to correlate data at sampling rates ranging from 0.13 to 204.7 milliseconds, on-line, with the user controlling all parameters from the terminal. It displays and scales while computing and provides output that can be used with other DIGITAL programs.

DAQUAN

DAQUAN is used for data acquisition in the time domain by boxcar, multisweep signal averaging, and for general-purpose data reduction. After the data is acquired, a wide variety of processing techniques are used to analyze the data interactively. A special feature of DAQUAN is its ability to determine peaks in a complex spectrum. Once the peaks have been defined, a report may be printed containing individual peak information consisting of peak minima, peak maxima, maxima as a percentage of the largest peak and the percent area.

Fast Fourier Transform (FFT) processing using DAQUAN requires EAE. DAFFT (Data Acquisition and FFT) and PAFFT (Power Averaged by FFT) have most of the functional capabilities of DAQUAN.

DAFFT has provisions for:

- Signal averaging of up to 1,024 double-precision points in the time domain.
- FFT into frequency domain; up to 1,024 complex coefficients.
- Power versus frequency spectrum as a sum of the squares of the coefficients.

PAFFT does signal averaging in the frequency domain with up to 1,024 double-precision points. The dead time between sweeps is the computation time for resetting to zero mean, performing the FFT, Hanning filtering, and computing and adding power to the double-precision average.

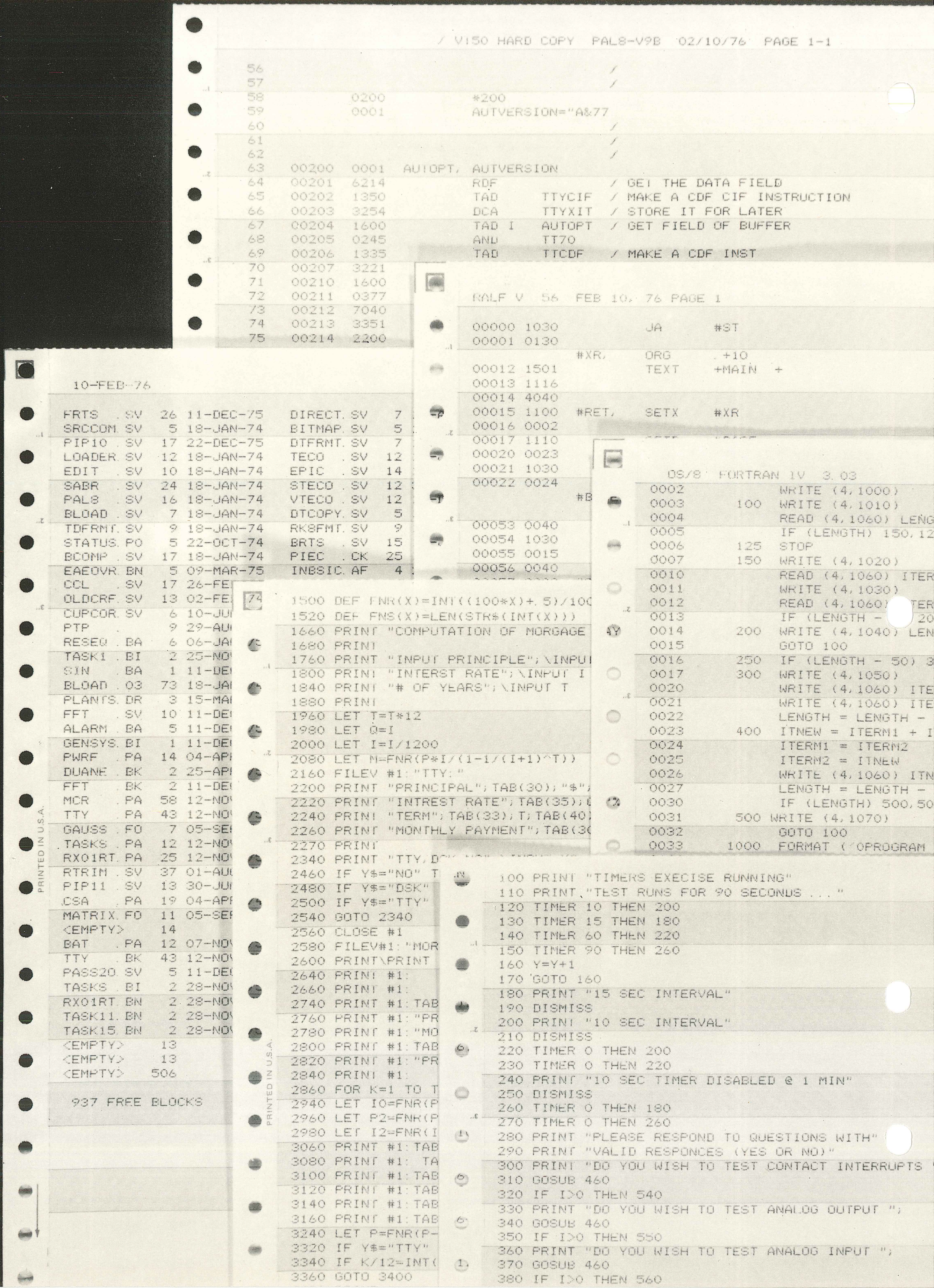
DAFFT and PAFFT Features:

- Perform complex FFT and inverse.
- Compute scaled-power spectrum.
- Optional 3- or 11-point digital filtering in time domain.
- Optional Hanning filtering in frequency domain.

Basic Signal Averager Application Package

The Basic Signal Averaging software allows an effective high-speed technique for improving the signal-to-noise ratio of a repetitive analog signal. Data is acquired via an analog-to-digital converter (ADC). The raw data is stored in a current buffer after the sweep and is added to the sum buffer after the sweep is over. During the sweep, control may be exercised to display the average or the last sweep.

Start-up of the program is conversational, allowing specification of the number of sweeps, sampling rate, and delay from the starting sync pulse.



```
                    / V150 HARD COPY  PAL8-V9B  02/10/76  PAGE 1-1

56                                              /
57                                              /
58           0200          *200
59           0001          AUTVERSION="A&77
60                                              /
61                                              /
62                                              /
63   00200   0001  AUTOPT, AUTVERSION
64   00201   6214          RDF             / GET THE DATA FIELD
65   00202   1350          TAD     TTYCIF  / MAKE A CDF CIF INSTRUCTION
66   00203   3254          DCA     TTYXIT  / STORE IT FOR LATER
67   00204   1600          TAD I   AUTOPT  / GET FIELD OF BUFFER
68   00205   0245          AND     TT70
69   00206   1335          TAD     TTCDF   / MAKE A CDF INST
70   00207   3221
71   00210   1600
72   00211   0377
73   00212   7040
74   00213   3351
75   00214   2200
```

```
RALF V  56  FEB 10, 76 PAGE 1

00000 1030           JA      #ST
00001 0130
             #XR,    ORG     . +10
00012 1501           TEXT    +MAIN  +
00013 1116
00014 4040
00015 1100   #RET,   SETX    #XR
00016 0002
00017 1110
00020 0023
00021 1030
00022 0024
             #B
00053 0040
00054 1030
00055 0015
00056 0040
```

```
 10-FEB-76

FRTS  .SV  26 11-DEC-75    DIRECT.SV   7
SRCCOM.SV   5 18-JAN-74    BITMAP.SV   5
PIP10 .SV  17 22-DEC-75    DTFRMT.SV   7
LOADER.SV  12 18-JAN-74    TECO  .SV  12
EDIT  .SV  10 18-JAN-74    EPIC  .SV  14
SABR  .SV  24 18-JAN-74    STECO .SV  12
PAL8  .SV  16 18-JAN-74    VTECO .SV  12
BLOAD .SV   7 18-JAN-74    DTCOPY.SV   5
TDFRMT.SV   9 18-JAN-74    RK8FMT.SV   9
STATUS.PO   5 22-OCT-74    BRTS  .SV  15
BCOMP .SV  17 18-JAN-74    PIEC  .CK  25
EAEOVR.BN   5 09-MAR-75    INBSIC.AF   4
CCL   .SV  17 26-FE
OLDCRF.SV  13 02-FE
CUPCOR.SV   6 10-JU
PTP         9 29-AU
RESEQ .BA   6 06-JA
TASK1 .BI   2 25-NO
SIN   .BA   1 11-DE
BLOAD .03  73 18-JA
PLANTS.DR   3 15-MA
FFT   .SV  10 11-DE
ALARM .BA   5 11-DE
GENSYS.BI   1 11-DE
PWRF  .PA  14 04-AP
DUANE .BK   2 25-AP
FFT   .BK   2 11-DE
MCR   .PA  58 12-NO
TTY   .PA  43 12-NO
GAUSS .FO   7 05-SE
TASKS .PA  12 12-NO
RX01RT.PA  25 12-NO
RTRIM .SV  37 01-AU
PIP11 .SV  13 30-JU
CSA   .PA  19 04-AP
MATRIX.FO  11 05-SE
<EMPTY>    14
BAT   .PA  12 07-NO
TTY   .BK  43 12-NO
PASS20.SV   5 11-DE
TASKS .BI   2 28-NO
RX01RT.BN   2 28-NO
TASK11.BN   2 28-NO
TASK15.BN   2 28-NO
<EMPTY>    13
<EMPTY>    13
<EMPTY>   506

 937 FREE BLOCKS
```

PRINTED IN U.S.A.

```
   OS/8  FORTRAN IV  3.03
0002              WRITE (4,1000)
0003        100   WRITE (4,1010)
0004              READ (4,1060) LENG
0005              IF (LENGTH) 150,12
0006        125   STOP
0007        150   WRITE (4,1020)
0010              READ (4,1060) ITER
0011              WRITE (4,1030)
0012              READ (4,1060)  TER
0013              IF (LENGTH -   20
0014        200   WRITE (4,1040) LEN
0015              GOTO 100
0016        250   IF (LENGTH - 50) 3
0017        300   WRITE (4,1050)
0020              WRITE (4,1060) ITE
0021              WRITE (4,1060) ITE
0022              LENGTH = LENGTH -
0023        400   ITNEW = ITERM1 + I
0024              ITERM1 = ITERM2
0025              ITERM2 = ITNEW
0026              WRITE (4,1060) ITN
0027              LENGTH = LENGTH -
0030              IF (LENGTH) 500,50
0031        500   WRITE (4,1070)
0032              GOTO 100
0033       1000   FORMAT ('0PROGRAM
```

```
1500 DEF FNR(X)=INT((100*X)+.5)/10
1520 DEF FNS(X)=LEN(STR$(INT(X)))
1660 PRINT "COMPUTATION OF MORGAGE
1680 PRINT
1760 PRINT "INPUT PRINCIPLE";\INPUT
1800 PRINT "INTERST RATE";\INPUT I
1840 PRINT "# OF YEARS";\INPUT T
1880 PRINT
1960 LET T=T*12
1980 LET Q=I
2000 LET I=I/1200
2080 LET M=FNR(P*I/(1-1/(I+1)^T))
2160 FILEV #1:"TTY:"
2200 PRINT "PRINCIPAL";TAB(30);"$";
2220 PRINT "INTREST RATE";TAB(35);
2240 PRINT "TERM";TAB(33);T;TAB(40)
2260 PRINT "MONTHLY PAYMENT";TAB(3
2270 PRINT
2340 PRINT "TTY,D
2460 IF Y$="NO" T
2480 IF Y$="DSK"
2500 IF Y$="TTY"
2540 GOTO 2340
2560 CLOSE #1
2580 FILEV#1:"MOR
2600 PRINT\PRINT
2640 PRINT #1:
2660 PRINT #1:
2740 PRINT #1:TAB
2760 PRINT #1:"PR
2780 PRINT #1:"MO
2800 PRINT #1:TAB
2820 PRINT #1:"PR
2840 PRINT #1:
2860 FOR K=1 TO T
2940 LET I0=FNR(P
2960 LET P2=FNR(P
2980 LET I2=FNR(I
3060 PRINT #1:TAB
3080 PRINT #1: TA
3100 PRINT #1:TAB
3120 PRINT #1:TAB
3140 PRINT #1:TAB
3160 PRINT #1:TAB
3240 LET P=FNR(P-
3320 IF Y$="TTY"
3340 IF K/12=INT(
3360 GOTO 3400
```

PRINTED IN U.S.A.

```
100 PRINT "TIMERS EXECISE RUNNING"
110 PRINT "TEST RUNS FOR 90 SECONDS ..."
120 TIMER 10 THEN 200
130 TIMER 15 THEN 180
140 TIMER 60 THEN 220
150 TIMER 90 THEN 260
160 Y=Y+1
170 GOTO 160
180 PRINT "15 SEC INTERVAL"
190 DISMISS
200 PRINT "10 SEC INTERVAL"
210 DISMISS
220 TIMER 0 THEN 200
230 TIMER 0 THEN 220
240 PRINT "10 SEC TIMER DISABLED @ 1 MIN"
250 DISMISS
260 TIMER 0 THEN 180
270 TIMER 0 THEN 260
280 PRINT "PLEASE RESPOND TO QUESTIONS WITH"
290 PRINT "VALID RESPONCES (YES OR NO)"
300 PRINT "DO YOU WISH TO TEST CONTACT INTERRUPTS
310 GOSUB 460
320 IF I>0 THEN 540
330 PRINT "DO YOU WISH TO TEST ANALOG OUTPUT ";
340 GOSUB 460
350 IF I>0 THEN 550
360 PRINT "DO YOU WISH TO TEST ANALOG INPUT ";
370 GOSUB 460
380 IF I>0 THEN 560
```

RTS/8 FOR MULTI-TASK, REAL-TIME APPLICATIONS

DIGITAL's RTS/8 Operating System has been designed as the solution to a typical problem: Your financial resources preclude the purchase of a large computer and yet your work load requires something more than a single-task system.

RTS/8 fits your intermediate position. It does this because it can do many tasks. For example, RTS/8 can run your batch processing jobs and handle your real-time applications concurrently. RTS/8 makes maximum use of the PDP-8/A Minicomputer because it was designed from the beginning as a complete, self-contained, independent, multi-task operating system. The RTS executive requires as little as 700 words of main memory.

In development applications, the RTS/8 system can help shorten product development cycles and therefore hold down costs. While you're running and debugging your software programs, you can use the same system to simultaneously perform real-time tasks on-line. It will allow up to 63 tasks to run concurrently, competing for system resources on a fixed-priority basis.

RTS/8 FEATURES

Multi-Tasking

RTS/8 provides its multi-tasking capabilities by making use of the otherwise idle processor cycles that occur periodically during a program's execution. RTS/8 maintains a list of jobs ready to run. Whenever an executing job is interrupted, the operating system selects a ready-to-run job from the list and processes the job.

Priority Allocation

RTS/8 maintains a user-defined priority list that allows "hot jobs" to be processed first. As a result, programs in execution can, if necessary, be temporarily suspended and removed from memory (swapped out), to make room for a higher-priority job.

Resource Management

RTS/8 maintains the current status of system resources to determine the combination of peripherals, input files, and run-time options to be used during the current execution of a program.

Foreground/Background Processing

To take advantage of the PDP-8/A efficiency, regardless of the level of real-time activity, a background task can be introduced into the system. What this means is that users with as little as 12K words of memory can develop programs in the background using OS/8 while running real-time tasks in the foreground under RTS/8.

RTS/8 is a two-for-one system written in assembly language and designed to reduce software development cost by reducing the time, capital investment, and effort required to develop a software system.

Included in the RTS/8 system are system modules (tasks) which control standard I/O devices, and a task which allows interactive system control from the console terminal. You can expand memory and add peripherals and interfaces to meet your specific needs. The non-resident task capability means that the amount of memory (core) required for a particular application can be far smaller than the sum of the various routines needed at various times to support the system. Foreground tasks are written in assembly language.

RTS/8 TASKS

The controlling program in an RTS/8 system is the RTS/8 Executive. The Executive enhances the PDP-8 architecture by running with task switching inhibited. That is, interrupts are on but higher-priority tasks may not request the CPU until the Executive has finished its request. When the Executive finishes processing the request, it checks whether any higher-priority tasks have become runnable while task switching was inhibited. If so, it runs the highest-priority task.

Each task in an RTS/8 system has an associated unique task number which serves the following purposes:

1. The Task Number is used by the RTS/8 Executive as an index to various system tables where information about tasks is kept.
2. The Task Number is used by other tasks in the system for reference in Executive Requests.
3. The Task Number determines its priority; the lower the Task Number, the higher the priority of the task.

By using RTS/8, the programmer is able to take advantage of a set of software modules that will interface with his hardware, thereby freeing him to concentrate on his own programs and greatly reduce development time.

RTS/8 MODULES

RTS/8 Executive (monitor)

The RTS/8 Executive controls the overall execution and interaction of task schedules, startup, and suspension, as well as the passing of information between tasks.

Monitor Console Routine (MCR) Module

The monitor console routine provides the programmer functions (which he can request from the console terminal) to control, inspect, and debug an experimental system.

RK8-E Module

This module controls the passing of information between a task and an RK8-E cartridge disk. Data is read and written in the standard RTS/8 block format.

RX8-E Floppy Disk Module

This handler enables interrupts from the RX01; it accepts and processes messages sent to it.

TC08 Module

This module controls the passing of information between a task and a DECtape unit. Data is read and written in the standard RTS/8 block format.

OS/8 Files Module

This module enables the user to look up, create, and delete files in OS/8 directories from a foreground task. When used in conjunction with one or more of the previously mentioned mass storage modules, it gives programmers the capability to read or write OS/8 files on mass storage devices.

OS/8 Background Module

The combination of the previously mentioned device drivers and the OS/8 Background module allows the execution of any of the OS/8 operating system utilities (i.e., PAL-8, BASIC, EDITOR, TECO, BATCH) to run under the RTS/8 executive. An LA36, VT50, or ASR33 terminal must be dedicated to OS/8 system execution. Included in the module is OS/8 terminal support and all related background functions.

Clock Module

The clock module accepts requests (in the form of RTS/8 message) to perform actions after a specified time has elapsed.

Console and Non-console Terminal Modules

These drivers each handle a single terminal in either line or character mode. Input in line mode is terminated by a carriage return or an ALTMODE character, and may be edited with a RUBOUT or CTRL/U character. In character mode, input is not echoed and is terminated by overflow of a specified character count.

Line Printer Module

The RTS/8 line printer driver supports an LA180 or LE8 line printer. Its structure is also identical to line mode in the terminal module.

Cassette Module

The RTS/8 cassette driver is used with the TA8-A/TU60 DECcassette drives to allow the user to read or write data on a cassette.

Cassette Files Module

This driver allows the user to look up, enter and delete files from a DECcassette. When used with the cassette driver, the user can read or write standard CAPS-8 format data files on DECcassette.

Power Fail/Auto-Restart Module

This driver provides the mechanism by which the system can recover from a power failure, if a low power condition occurs.

Industrial Control Subsystem (ICS-8)

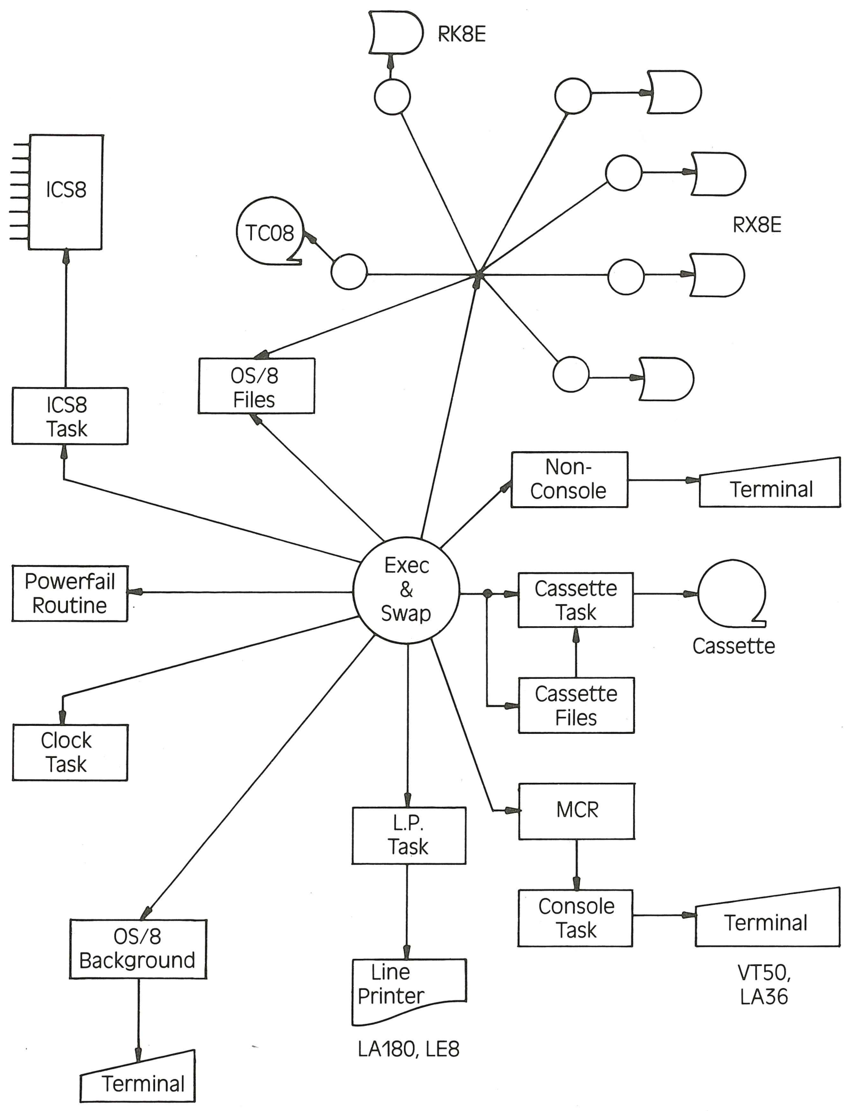
This driver enables the user to control all the various types of ICS modules. It performs two types of actions—immediate and deferred. Immediate actions include reading and sending analog and digital values to appropriate ICS modules. Deferred actions may be linked to specified events within the ICS (counters overflowing, switches being thrown).

Swap Module

This task actually swaps other tasks into or out of memory. It determines if a task is already in memory or whether it must be first written onto mass storage before another task may reside in its partition (area of memory) or whether a new task can be swapped into memory without regard to what was in the partition previously.

MEMORY REQUIREMENTS

The memory requirements for an RTS/8 system vary, depending upon the size of user-generated tasks and the particular mix of RTS/8 modules used. To determine the size of the DIGITAL-supplied portion of the system, simply start with the monitor and add to it the particular modules and drivers which are needed.



RTS/8 System Configuration

RTS/8 MEMORY LAYOUT

Modules	Size
Executive	640 + 7 per Task
Clock	384
Swap	256
Console Terminal	384
Non-console Terminal	384
RK08	128
RK8/E	256
RF08/DF32	128
OS/8 Files	768
OS/8	768 + 8K + Drivers
OS/8 with BATCH	768 + 12K + Drivers
PF/Auto Restart	128
ICS-8	896—1152
Cassette Files	384
Line Printer	128
TC08	256
Monitor Console Routine	896—1408
RX8E Handler	256

MINIMUM SYSTEM CONFIGURATION

The minimum RTS/8 configuration for a run-time system is a PDP-8/A family processor with 4K of memory.

Minimum RTS/8 development configuration is OS/8 Version III operating system which requires a PDP-8 with 8K of memory, 64K words of mass storage and a terminal (VT50, LA36 or Teletype).

digital

DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS 01754

PRINTED IN U.S.A. EA 05604 76 06A/19 K 04 20 (06428)