DEUTSCHES ELEKTRONEN-SYNCHROTRON DESY

DESY 85-046
May 1985

A DATA PROCESSING SYSTEM BASED ON THE 370/E EMULATOR

by

D. Notz

Deutsches Elektronen-Synchrotron DESY, Hamburg

ISSN 0418-9833

NOTKESTRASSE 85 · 2 HAMBURG 52

DESY 85-046 ISSN 0418-9833
May 1985

A Data Processing System based on the 370/E Emulator

Dieter Notz

Deutsches Elektronen-Synchrotron, DESY, Hamburg, Germany

Abstract

This paper describes the DESY implementation of a 370/E based data processing system.
The 370/E was designed at the Weizmann Institute in Israel by a team led by Hanoch Brafman and emulates an IBM 370/168 mainframe computer. This system can process large megabyte sized programs with a speed approximately 1/4 that of an IBM 3081D mainframe.
Four processors are connected via PADAC interfaces to the IBM, NORD, VAX or TMS9900.

Introduction

There is an increasing demand for computer power in high energy physics. In the era of the forthcoming accelerators a data production rate of 400 tapes per day is estimated. All these data have to be analysed and compared with theoretical predictions.
People designing accelerators need computers to simulate the beam optics. These programs are not I/O intensive and need a lot of number crunching power. In order to support physicists with cheap and IBM compatible computer power the 370/E emulator has been developed at the Weizmann Institute by H. Brafman (Ref.1). Emulation is defined as "the desire to equal or surpass a rival". In this sense the 370/E is a computer which from the user's point of view is indistinguishable from an IBM 370.
In high energy physics the term emulator has become associated with the SLAC 168/E which was designed by P. Kunz (Ref. 2). The 168/E was a successful product and many systems have been built. However, the 168/E with its limited access and separated memory for data and instructions could not run all programs without considerable user involvement. Especially formatted I/O was painful and normally not used.
The advantage of the 370/E is its architectural similarity to the IBM architecture. A combined memory for data and instructions is used and the IBM instructions are emulated directly. Therefore one does not need to translate the programs before running them on the emulator. One only has to link the program together with the 370/E system and the FORTRAN I/O routines and download it. The price of the processor is 45 kDM with a 2 Mbyte memory. Approximately 23 processors are in operation in High Energy Physics (Ref. 3) so far. A new version which is 20% faster has been completed at the Weizmann Institute.

Description

The main components of the 370/E are shown in Fig. 1. The 370/E consists of 14 boards with the dimensions 39.4cm * 23.5cm. The whole processor therefore fits into a box of a typical crate size 45cm * 30cm * 40cm.
The arithmetic and logic unit is divided into five parts . An integer CPU, a dedicated multiplier, two floating point boards and a control unit. The eight memory boards may contain up to 2 Mbytes of memory. If desired the backplane can be easily increased to give space for 4 Mbytes. From the address space point of view the processor can be equipped with 16 Mbytes of memory.

Connections to Host Computers

In our application the 370/E has no I/O devices. It is controlled by a host computer. At DESY an interface has been built for PADAC which allows a connection of the 370/E to a NORD 10, NORD 100, VAX, PDP 11, TMS9900 or NS32016.

The transfer rates without DMA setup time are 1.25 μsec/byte to a NORD100/Emulator, 1.5 μsec/byte to a VAX and 1.57 μsec/byte to a TMS9900. Of the variety of possible connections we describe here only two alternatives, an online and an offline application.

Online Application

Fig. 2 shows a typical online application. At the beginning of a data taking run a prepared load module and the latest constants are transferred from the IBM via the online net to the online computer (NORD, VAX, PDP11) which then loads the program into the 370/E. The constants are stored on local disks. The emulator is then started and gets first the constants and afterwards the experiment's data as they are read out by the online computer. Due to the double buffering in the I/O system the processor can analyse the first event while the second event is read in. From the programmer's point of view one only has to read an event with READ(1,END=4)L,(IEVNT(I),I=1,L) and output it with a WRITE(2)... statement. All error messages and run summaries can be transferred to the online computer via a WRITE(6,...) and printed there. At the end of a run an end-of-file is generated which will close all files and halt the processor.

Offline Application

Fig. 3 shows the offline application. The user sits at the IBM terminal and the 370/E is connected to the IBM via the online net and a TMS9900 microprocessor. To the user, the 370/E looks like an attached processor to the IBM although it is 500m away from the computer center. The TMS9900 acts as a host, checks the connection to the IBM and to the 370/E and looks every 2 minutes at the jobqueue on the IBM to determine whether or not a job has been submitted. The program in the TMS9900 runs for ever and only needs restarting in case of a power failure. The user who wants to submit a job sits in front of an IBM terminal in his known environment. We assume that a big program which has already been developed by several people should run on the 370/E. The following steps are then needed :

a) Prepare a Load Module
The load module is built by the IBM linkage editor. This can be performed in different ways: One can use the LKED procedure under NEWLIB which must load the 370/E system first and afterwards all user's programs

or
one can run a small batch job which links all routines and libraries. Fig. 4 shows an example of such a job.

b) Allocate all Files
All files which should be accessed by the 370/E must reside on disks. In addition one has to create a file LISTFILE which will contain the printout of the 370/E.

c) Prepare Job Control Cards
As in all IBM jobs one has to inform the system of the files which should be opened for each unit. Also the name of the load module and the time limit must be given. The job control file is stored in the user's library. Fig. 5 gives an example.

d) Submit job
In order to submit a job the user must give the submit command S370 or CALL 'TASSO1.LIBRARY(SUBM370E)' and must type the names of the file containing the job control information (i.E. TASSO1.SOURCE(JCL370)). The job is now placed into the jobqueue and will be executed later (Fig. 6).

e) Check Jobstatus
With the command J370 or CALL 'TASSO1.LIBRARY(JOBS370E)' the user gets a list of the last 16 jobs in the 370/E. He can estimate how long he has to wait before the job will be started (Fig. 7). If the job is running he may cancel the job by the CANCEL command or may look at the printout by LIST 'TASSO1.LIST370E.

The 370/E Operating System

The operating system of the DESY 370/E is adapted to our needs and environment. Only a single user runs on the processor at one time. We do not support any multitasking. The processor is connected to an IBM and should support all I/O facilities the user normally gets on the mainframe like sequential READ, WRITE, REWIND, direct access READ, WRITE, FIND and full support in case of errors like divide check or negative SQRT. In order to get this service all programs doing input/output must be written in FORTRAN IV or FORTRAN 77 and must be compiled by the IBM compiler.
The layout of the operating system is shown in fig. 8. The first locations are fixed and allocated to program status words (PSW's) and channel address

(CAW) and channel status words (CSW's) as in the IBM 370. The first word contains the PSW for initial program load (IPL) to start the program. The section for unsupported operation code contains routines to simulate some instructions which are not implemented in the hardware like move character long MVCL or CLCL. For REAL*16 operations one can load a simulation package (IEAXPALL) from the system link library (SYS1.LINKLIB) to which control is transferred.
The supervisor call handler (SVC) supports the following IBM supervisor calls:

SVC 3,	EXIT,	to terminate a task
SVC 4,	GETMAIN,	to allocate dynamic memory
SVC 5,	FREEMAIN,	to release dynamic memory
SVC 8,	LOAD,	to load a member declared by IDENTIFY
SVC 9,	DELETE,	to delete a member
SVC 10,	GETMAIN,	to allocate dynamic memory
	FREEMAIN,	to release dynamic memory
SVC 13,	ABEND,	to terminate a task abnormally
SVC 14,	SPIE,	to set or cancel SPIE exit
SVC 35,	WTO,	to write to the operator
	WTOR,	to write to operator and reply
SVC 40,	EXTRACT,	to provide information from task control block
SVC 41,	IDENTIFY,	to add an entry point to a copy of a load module
SVC 60,	STAE,	to set or cancel STAE exit.

All information concerning open files is stored in the IHOUAC table. This table indicates which unit is open for sequential or direct access I/O. Before a load module is downloaded the IBM opens all files and transfers the data control block (DCB) parameters into this table. By this method the 370/E knows which files are accessible and which record length and blocksize should be used.
All other constants from outside like date, time, size of program and jobname are inserted into the load module on fixed locations before downloading.
The rest of the operating system belongs to the FORTRAN input/output package.
The operating system is linked in front of the user's program by an INCLUDE TASSO(SYST370E) statement in the linkage editor. The user's main program and all other subroutines are loaded in the middle. The remaining space is used for dynamic allocation of I/O buffers or histogram routines.

Input/Output for IBM FORTRAN programs

Fig. 9 indicates the user's program on the IBM written in FORTRAN. All I/O requests to files must be transferred in such a way that the user does not know whether his program runs on the IBM or on the emulator. This is done in the following way (Ref. 4): Each FORTRAN program which was generated by the IBM compiler generates a call to IBCOM# for each READ or WRITE. A lot of parameters like addresses and FORMAT statements are exchanged between the program and the FORTRAN I/O package. IBCOM# then does the formatting and transfers buffers to FIOCS#. Here only a few parameters like the unit number, I/O request, buffer address and buffer length are exchanged.

In the case of the 370/E the FORTRAN runtime library has been split into two parts: IBCOM# runs on the 370/E processor and FIOCS# runs on the IBM or host. For direct access I/O the routines DIOCS1 and DIOCS4 are used. The corresponding program on the IBM is DIOCS# and DEFILE. IBMTRA is the actual transfer routine between the 370/E and the host. As the information which is exchanged between the 370/E and the IBM is well known the IBM can easily be replaced by a minicomputer like NORD or VAX as long as the files are delivered in IBM format.
The transfer speed between the 370/E and the IBM is 5 µsec/byte. In order to avoid a slowing down of the processor caused by this transfer rate several levels of pipelining are used: The processor has two buffers for each I/O unit and the IBM also has two buffers. For sequential input and output and for direct access output the processor can continue with its calculation while the transfer takes place. For direct access input (!) the processor must wait until the record is really shipped down from the IBM disk into the 370/E memory. If several consecutive direct access records are read into the user's program the 370/E issues a FIND request to the IBM so that the next record can be transferred while the program is still operating on the previous data.

Status and Performance

Fig. 10 shows an example of a job which was executed on the 370/E. The only indication that the job did not run on an IBM but on the emulator are the addresses in the traceback of the error messages. All addresses are the same as in the linkage editor.
Four 370/Es are running at DESY. One processor has operated for a year and has executed 1116 jobs using 150 000 min CPU (370/E time). The job profile can be seen in fig. 11. Apart from short tests many jobs remain in the 370/E for several hours . The only problems which occur from time to time are breakdowns of the IBM link and the IBM online system. In addition,some jobs need a lot of data from the mass storage device. If these data cannot be delivered within a time limit of 1 minute the job will be cancelled in order to release the link. Normally data are transferred from the mass storage system to disk when the job is submitted by the user. If the waiting time for job execution is not too long data remain on disk and are available when the job is started.
The processor does not introduce any problems to the users once they have learned how to build an IBM load module. From the point of view of the computer center the 370/E looks like one of the 40 online jobs which are running in the mainframe.

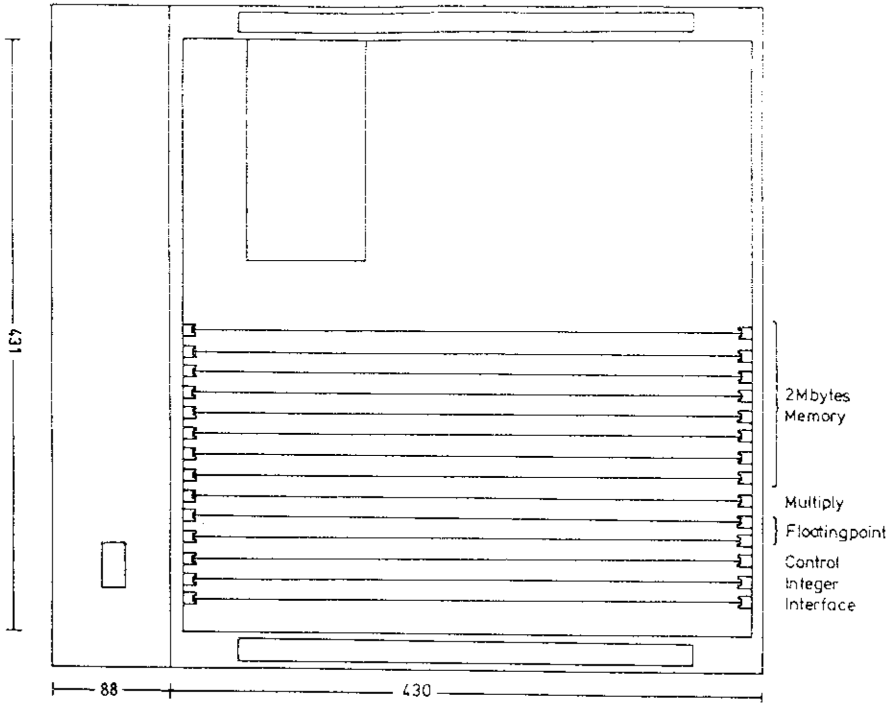
Acknowledgement

The work described in this paper was only possible due to the enormous amount of work carried out by Hanoch Brafman and his team. One of the processors at DESY was built at the Weizmann Institute and installed by Richard Fall and Yaron Gal. The operating system was upgraded by David Botterill (RAL) and implemented at DESY by Rafi Yaari. A lot of the construction of the DESY processor was done by Kay Rehlich, Klaus Nimmer

(DESY) and Bob Hatley (RAL). The memory with 1/4 Mbyte per card was laid out by Chris Bebeck (Cornell). The routines in the TMS9900 were written by Martin Dieckvoß (Hamburg University) and the online link has been made available by Gerd Hochweller and his group.

References

(1) H. Brafman et.al., A Fast General Purpose IBM Hardware Emulator, Weizmann Institute, Dept. of Nuclear Physics, Internal Report, January 1983 Review on the Impact of Specialized Processors in Elementary Particle Physics, Padova, March 23-25,1983.
(2) P. Kunz et. al., Experience using the 168/E Microprocessor for Offline Data Analysis, SLAC-PUB-2418, October 1979
(3) Institutions and contact persons using 370/E's (Number of processors in brackets):
Weizmann Institute, H. Brafman (3); Rutherford Lab., J. Barlow (2);
DESY, D. Notz, (4); Aachen, G. Peise (1); Bonn, M. Kokott (1);
Siegen, M. Rost (1); Imperial College, G. Fayers, (1); Birmingham,
H. Shaylor, (1); Tel Aviv, Y. Gnat, (1); Cornell, C. Bebeck, (6);
CERN, P. Schmid, F. Chevrier, DELPHI, OPAL (2).
(4) D. Notz, The Input/Output Software for the 370/E Emulator, DESY, Internal Report F1-82/01, 1982 and TASSO Note No. 251, March 1983 (unpublished).



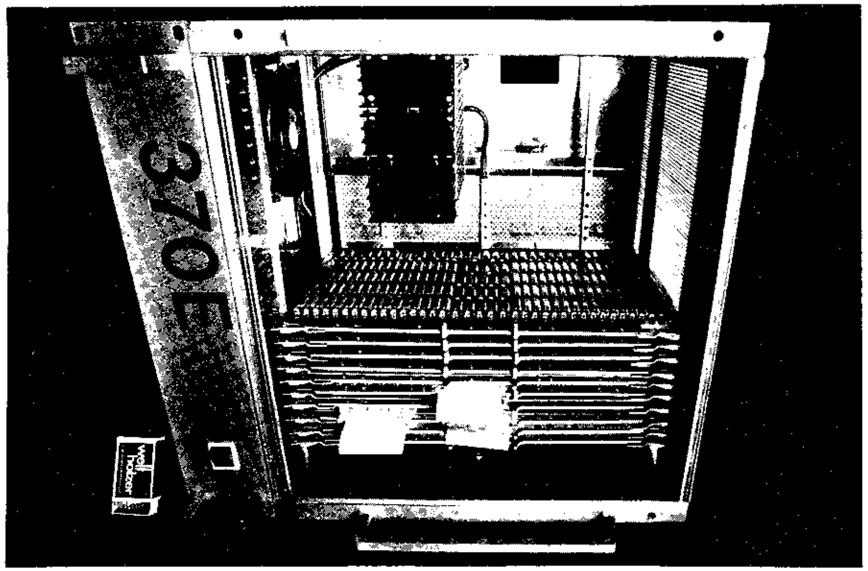


Fig. 1 The 370/E consists of 5 CPU boards, 1 interface board and 4 to 8 memory boards.

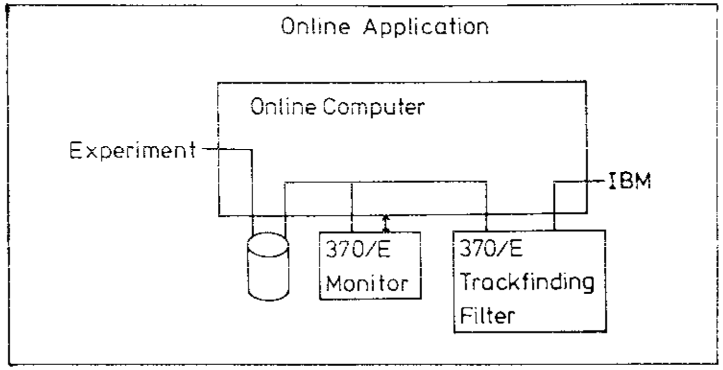


Fig. 2 Data from an experiment are written into a buffer or on a disk. Then they are analysed and filtered by the 370/E and sent to the IBM or on a tape.

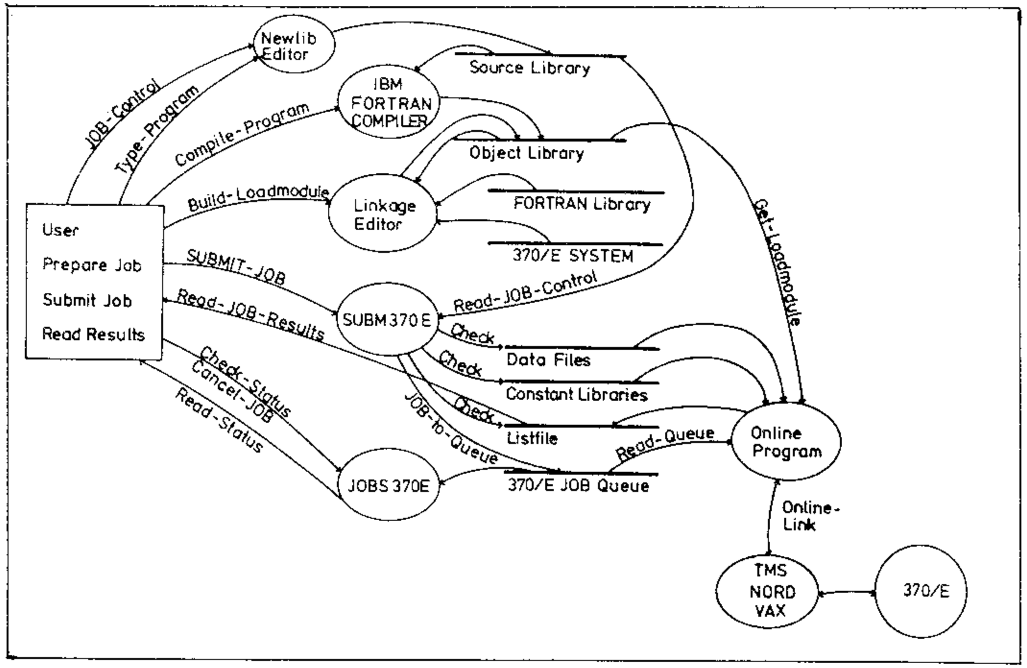


Fig. 3 Offline application. The user sits at the IBM terminal, writes a program (NEWLIB) and compiles it. Afterwards the program is linked (LINKAGE EDITOR) together with the 370/E operating system. The job is then submitted and has access to all IBM files. Results can be read from LISTFILE.

```
//            JOB '10601601,GEP-USER',HOTZ,CLASS=E,TIME=(,5)
//*MAIN ORG=EXT,RELPRI=HIG
// EXEC FORTHCL,PARM.LKED='MAP,LIST'
C
C      FORTRAN PROGRAM
C
       STOP
       END
//LKED.SYSLIN DD DDNAME=SYSIN
//             DD DSN=&&LKSET,DISP=(OLD,DELETE)
//             DD DUMMY
//LKED.SYSLIB DD
//             DD
//             DD DSN=H02BAS.GPHINI,DISP=SHR
//             DD DSN=H02BAS.GEPL9,DISP=SHR
//LKED.SYSLMOD DD DSN=F1BNOT.TSOLIBL(GEP470E),DISP=(SHR,KEEP)
//LKED.TASSO DD DSN=TASSO1.LIBRARY,DISP=SHR
//LKED.TSO    DD DSN=F1BNOT.TSOLIBL,DISP=SHR
//LKED.SYSIN DD *
 INCLUDE TASSO(SYST370E)
//
```

Fig. 4 Example of a job to prepare a load module (compile and link). One can connect all libraries to the linkage editor. The 370/E system is loaded by an INCLUDE TASSO(SYST370E) statement.

```
//F1BNOT00 JOB TIME=5
//STEP00 EXEC PGM=F1BNOT.TSOLIBL(GEP470E)
//LISTFILE DD DSN=F1BNOT.LIST371E
//FT46F001 DD DSN=F1BNOT.GEP46
//FT48F001 DD DSN=F1BNOT.GEP48
```

Fig. 5 Job control language for a 5 minutes job and 2 output files for graphic information.

```
TYPE IN NAME OF FILE CONTAINING JOB CONTROL CARDS
EXAMPLE: TASSO1.SOURCE(JCL370)
f1bnot.tsolib(jclgep)
//F1BNOT00 JOB TIME=5
GEP470E)
ALLOCATE FILE 88
//GO.FT88F001 DD DSN=F1BNOT.TSOLIBL(GEP470E)

LENGTH OF PROGRAM 211832 00033B78
---------------------------------
ALLOCATE FILE 86
//GO.FT88F001 DD DSN=F1BNOT.LIST371E
ALLOCATE FILE 46
//GO.FT88F001 DD DSN=F1BNOT.GEP46
ALLOCATE FILE 48
//GO.FT88F001 DD DSN=F1BNOT.GEP48
JOB NO  1871 F1BNOT00 TIME=   5 SUBMITTED TO 370/E
PROCESSOR HAS CHECKED QUEUE AT  07/02/85 09.40.45 LAST JOB: F1BNOT00
```

Fig. 6 Submit a job. The user gives the file which contains the job control information (JCL). The length of the load module is checked and all files are allocated.

```
 2860 JOBS DONE,  0 JOBS WAITING. PROCESSOR WAS ACTIVE AT 29/04/85 11.31.01
2845+1 F35FRS00  800 F35FRS.B1.L(T01)            LIST372E   18/04/ 14.06 15.27
2846+1 F35FRS00  800 F35FRS.B1.L(T01)            LIST372E   18/04/ 19.13 19.30
2847+1 F35FRS00  800 F35FRS.B1.L(T01)            LIST372E   18/04/ 19.42 19.55
2848+1 F35FRS001200 F35FRS.B1.L(T01)             LIST372E   18/04/ 19.57 12.31
2849+1 F35FRS00  200 F35FRS.B1.L(T01)            LIST372E   19/04/ 16.33 17.04
2850+1 F35FRS00  200 F35FRS.B1.L(T01)            LIST372E   19/04/ 18.51 18.58
2851+1 F35FRS001200 F35FRS.B1.L(T01)             LIST372E   19/04/ 19.00 11.35
2852+1 F35FRS001200 F35FRS.B1.L(T01)             LIST370E   20/04/ 18.19 10.54
2853+1 TASSO100    1 TASSO1.LIBRARY(E370TEMP     LIST370E   22/04/ 08.13 08.14
2854+1 F35FRS001200 F35FRS.B1.L(T01)             LIST370E   22/04/ 18.37 11.12
2855+1 F35FRS001200 F35FRS.B1.L(T01)             LIST370E   25/04/ 16.53 20.11
2856+1 F1BNOT04    6 F1BNOT.TSOLIBL(E470TEST     LIST371E   26/04/ 13.52 14.03
2857+1 F35FRS001200 F35FRS.B1.L(T01)             LIST370E   26/04/ 19.38 19.46
2858+1 F35FRS001200 F35FRS.B1.L(T01)             LIST370E   26/04/ 19.48 19.49
2859+1 F35FRS001200 F35FRS.B1.L(T01)             LIST370E   26/04/ 19.54 23.10
2860+1 F35FRS001200 F35FRS.B1.L(T01)             LIST372E   26/04/ 23.13 15.48
TYPE: CANCEL   OR   JCLJOB   OR   STOP   OR   EXIT
```

Fig. 7 The job status shows which job has finished.

```
PROGRAM STATUS WORDS
CHANNEL ADDRESS AND CHANNEL STATUS WORDS
------------------------------------------------
FIXED LOCATIONS FOR PROGRAM AND PROCESSOR SIZE
------------------------------------------------
INTERRUPT HANDLER
------------------------------------------------
SIMULATE UNSUPPORTED OPERATION CODES
------------------------------------------------
OPERATION SYSTEM FOR PROCESSOR WITH OLD INTERFACE
------------------------------------------------
FIXED LOCATIONS FOR INTERRUPT, DATE,TIME, SIZE
------------------------------------------------
IHOUAC TABLE FOR FORTRAN UNITS AND DCB'S
------------------------------------------------
INPUT/OUTPUT ROUTINES, BUFFER HANDLER
------------------------------------------------
TRACE BACK ROUTINES, ERROR HANDLING
------------------------------------------------
DIRECT ACCESS INPUT/OUTPUT HANDLING
------------------------------------------------
LAST ADDRESS: 8710 (HEX)
```

Fig. 8 The operating system contains the supervisor call handler, the program interrupt handler and part of the FORTRAN input/output routines.

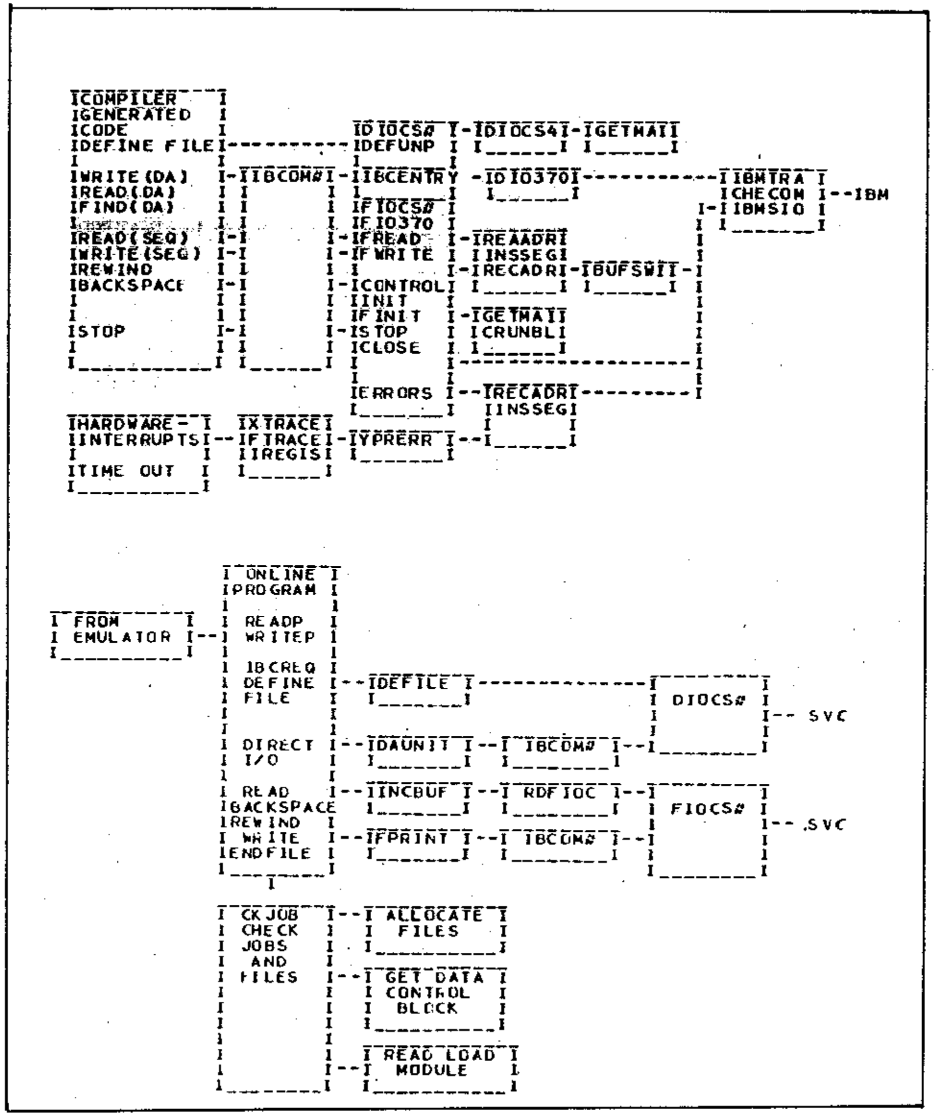


Fig. 9 Interface routines between user's program and IBM. The FORTRAN routines are written in such a way that the user does not need to change his program.

```
                   C    22/04/85 504291124  MEMBER NAME  MAIN370   (TSOLIB)
    ISN 0002                CALL QUAD
    ISN 0003                CALL DACCES
    ISN 0004                CALL SEQUEN
    ISN 0005                CALL STA371
    ISN 0006                STOP
    ISN 0007                END
*OPTIONS IN EFFECT*NAME(MAIN) OPTIMIZE(2) LINECOUNT(60) SIZE(MAX) AUTODBL(NO
                   C    12/02/85 504220948  MEMBER NAME  QUAD      (TSOLIB)
    ISN 0002                SUBROUTINE QUAD
                   C-------------------          EXAMPLE FOR REAL * 16
    ISN 0003                REAL*16 QA,QB,QC,QD
    ISN 0004                QA=2.
    ISN 0005                QB=4.
    ISN 0006                QC=QA*QB
    ISN 0007                QD=4Q0*QATAN(1Q0)
    ISN 0008                WRITE(6,2)QA,QB,QC,QD
    ISN 0009       2        FORMAT(///1X,'REAL * 16:',4F20.10)
    ISN 0010                RETURN
    ISN 0011                END
*OPTIONS IN EFFECT*NAME(MAIN) OPTIMIZE(2) LINECOUNT(60) SIZE(MAX) AUTODBL(NO
                   C    02/07/82 502140951  MEMBER NAME  STA371    (TSOLIB)
    ISN 0002                SUBROUTINE STA371
                   C---------------------      EXAMPLES FOR SOME ERRORS -------
    ISN 0003                DIMENSION ADR(1),UTIME(4)
                   C                            PRINT SIN  AND  COS
    ISN 0004                IUNIT=6
    ISN 0005                WRITE(IUNIT,1)
    ISN 0006         1      FORMAT(1X,'SIN COS TABLE')
    ISN 0007                PI=3.141592
    ISN 0008                DO 2 I=20,180,20
    ISN 0009                RADPI=I*PI
    ISN 0010                RAD=RADPI/180.
    ISN 0011                SN=SIN(RAD)
    ISN 0012                CS=COS(RAD)
    ISN 0013                WRITE(IUNIT,4)I,SN,CS,RAD
    ISN 0014       4        FORMAT(1X,I4,3F10.4)
    ISN 0015       2        CONTINUE
                   C                           FLOAT DIVIDE
    ISN 0016                A=3.
    ISN 0017                B=0
    ISN 0018                C=A/B
    ISN 0019                WRITE(IUNIT,6)A,B,C
    ISN 0020         6      FORMAT(1X,'AFTER DIVIDE CHECK',3F10.5)
                   C                           FIXED DIVIDE
    ISN 0021                I=5
    ISN 0022                J=0
    ISN 0023                K=I/J
    ISN 0024                WRITE(IUNIT,8)I,J,K
    ISN 0025        8       FORMAT(1X,'AFTER FIXED DIVIDE',3I6)
                   C                           OVERFLOW
    ISN 0026                EOV=1.E60
    ISN 0027                EOVL=EOV*EOV
    ISN 0028                WRITE(IUNIT,10)EOV,EOVL
    ISN 0029        10      FORMAT(1X,'OVERFL',2E15.7)
                   C                           UNDERFLOW
    ISN 0030                EUN=1.E-60
    ISN 0031                EUNV=EUN*EUN
                   C                          NEGATIVE SQRT
    ISN 0032                AA=SQRT(-1.)
    ISN 0033                WRITE(IUNIT,14)AA
    ISN 0034        14      FORMAT(1X,'AFTER NEGATIVE SQRT',1F15.5)
                   C                          ADDRESS EXCEPTION
    ISN 0035                WRITE(IUNIT,16)
    ISN 0036        16      FORMAT(1X,'BEFORE ADDRESS VIOLATION')
    ISN 0037                II=2 300 000
    ISN 0038                AB=ADR(II)
    ISN 0039       99       WRITE(IUNIT,20)
    ISN 0040        20      FORMAT(1X,'FINISH')
                   C                          WRONG UNIT
                   C        WRITE(0,12)
    ISN 0041                WRITE(IUNIT,12)
    ISN 0042       12       FORMAT(1X,'AFTER WRONG UNIT')
    ISN 0043                RETURN
    ISN 0044                END
*OPTIONS IN EFFECT*NAME(MAIN) OPTIMIZE(2) LINECOUNT(60) SIZE(MAX) AUTODBL(NO

    ISN 0002                SUBROUTINE DEFINE
    ISN 0003                COMMON/DEFILC/IU,IV
    ISN 0004                DEFINE FILE 12(20,25,U,IV),13(22,100,L,IU)
    ISN 0005                RETURN
    ISN 0006                END
*OPTIONS IN EFFECT*NAME(MAIN) OPTIMIZE(2) LINECOUNT(60) SIZE(MAX) AUTODBL(NO
```

Fig. 10 This program gives an example how input/output and errors are handled by the 370/E.

```
        C   24/01/83 305060830  MEMBER NAME  COPSU2    (TSOLIB)        FORTRAN
                SUBROUTINE DACCES
        C---------------------------------------------------------------------
        C
                COMMON/DEFILC/IU,IV
                DIMENSION IAR(25)
                WRITE(6,20)
         20     FORMAT(//1X,'START DIRECT ACCESS TEST')
                CALL DEFINE
                WRITE(6,10 )
        10      FORMAT(1X,'WRITE TO 12')
         32     FORMAT(1X,'NEXT RECORD IS:IV=',I4)
                DO 4 I=1,5
                DO 2 K=1,25
         2      IAR(K)=I+K-1
                WRITE(12'I)IAR
                WRITE(6,32)IV
            4   CONTINUE
                WRITE(6,12)
        12      FORMAT(1X,'WRITE 4 RECORDS TO 13 STARTING RECORD 2')
                WRITE(13'2)IAR,IAR,IAR,IAR
                WRITE(6,33)IU
        33      FORMAT(1X,' NEXT FREE RECORD IU',I4)
                WRITE(6,36)
        36      FORMAT(1X,' WRITE 5 RECORDS TO UNIT 13')
                DO 5 I=1,5
                DO 22 K=1,25
         22     IAR(K)=I+K-1
                WRITE(13'I)IAR
                WRITE(6,33)IU
            5   CONTINUE
                WRITE(6,38)
        38      FORMAT(1X,' WRITE RECORD 3 ON UNIT 12 AND RECORD 1+2 ON UNIT 13')
                WRITE(12'3)IAR
                WRITE(6,32)IV
                WRITE(13'1)IAR,IAR
                WRITE(6,33)IU
                WRITE(6,14)
        14      FORMAT(1X,'READ FROM 12')
                DO 6 I=1,5
                READ  (12'I)IAR
                WRITE(6,32)IV
            6   CONTINUE
                WRITE(6,12)
                READ(13'2)IAR,IAR,IAR,IAR
                WRITE(6,33)IU
                WRITE(6,16)
        16      FORMAT(1X,'READ FROM 13')
                DO 7 I=1,5
                READ  (13'I)IAR
                WRITE(6,33)IU
            7   CONTINUE
                WRITE(6,40)
          40    FORMAT(1X,' FIND RECORD ON UNIT 13')
                FIND(13'4)
ON 1.3.0 (01 MAY 80)      DACCES     SYSTEM/370 FORTRAN H EXTENDED (ENHANCED)     DAT
                WRITE(6,33)IU
                RETURN
                END
                SUBROUTINE SEQUEN
        C--------------------           SEQUENTIAL INPUT/OUTPUT
                DIMENSION  ZAHL(100)
                INTEGER ITXT(34)/' 123',33*'4567'/
                LOGICAL * 1 LTXT(133)
                EQUIVALENCE(LTXT(1),ITXT(1))
                WRITE(6,11)
        11      FORMAT(///1X,' SEQUENTIAL INPUT/OUTPUT TEST')
                DO 10 I = 1,100
                ZAHL(I)=I
            10  CONTINUE
                WRITE(6,12)ZAHL
           12   FORMAT(1X,10F7.2)
         6       FORMAT(133A1)
                WRITE(9,20)
        20      FORMAT(1X,'TEXT FOR UNIT 9 BEFORE REWIND')
                REWIND 9
                WRITE(9,22)
        22      FORMAT(1X,' TEXT FOR UNIT 9 BEFORE BACKSPACE')
                BACKSPACE 9
                WRITE(9,26)
        26      FORMAT(1X,' TEXT FOR UNIT 9 BEFORE ENDFILE')
                ENDFILE 9
                DO 14 I = 1,11
                READ(8,6,END=24) (LTXT(K),K=1,I)
                WRITE(6,6) (LTXT(K),K=1,I)
            14  CONTINUE
         24     CONTINUE
                WRITE(6,16)
          16    FORMAT(1X,'FINISH SEQUENTIAL')
                RETURN
                END
```

```
  F64-LEVEL LINKAGE EDITOR OI
              DEFAULT OPTION(S)
IEW0000         INCLUDE TASSO(SI
IEW0000         INCLUDE LINK(IEA


   CONTROL SECTION
     NAME       ORIGIN    LENGTH
   LOWC370E        00        D00

   OPCT370E       D00         28
   SIMU370E       D28        288
   SVCT370E       FB0         30
   TEMP370E       FE0         20
   MAIN370       1000         78
   CPLIST        1078         CC
   IHOUAC        1148        648
   INTSRV        1790         AC
   IHCBUF        1840         18
   CTMCOM        1858         30
   IHCBF2        1888          8
 $PRIVATE        1890         6C


   MAIN2         1900         C6
   GETMAI        19C8         38
   SVCALL        1A00         40
   EXT370        1A40          2
   READP         1A48          2
   CTRACE        1A50         40
   SUBR370E      1A90        224

   MVCOM         1CB8         56

   MACHSIZE      1D10        10B
   IHOFIOS2      1E20          4
   IHOEFIOS      1E28        254

   IFYVSIOS      2080        1EC

   IFYVCMSS      2270          2

   IFYVDIOS      2278          2

   IFYCVIOS      2280          2

   IFYVSTAE      2288          C
   GETMNW        2296         44

     NAME       ORIGIN    LENGTH
   IADDR         22E0          A
   IDSTL         22F0         14
   IREGIS        2308          A
   TIOINT        2318        1AC

   X2EBCD        24C8        134
   XLATE         2600        283
   FIOINI        2888         F8
   GETMNO        2980        290

   GETCOR        2C10        218

   IBMSIO        2E26       3718
   BFFR1B        6540        226
```

```
   BUFSWI         6768        29E
   CHECOM         6A08        198
   CHUNBL         6BA0        2C6
   FIO370         6E68        C56

   WTOZ           7AC0        240
   IBMTRA         7D00        3A8
   REAADR         80A8        376
   RECADR         8420        2E6
   FTRACE         8708        976
   XTRACE         9080        48C
   YPRERR         9510        240
   DCBSET         9750        1BE
   INSSEG         9910        1AC
   DIOCS#         9AC0         2C
   DEFUNP         9AF0        250
   DIOCS4         9D40        5BA
   DIO370         A300        D50

   FIOSTP         B050        23E
   FIOCHA         B290        110
   VFILL          B3A0        122
   VZERO          B4C8        122
   VBLANK         B5F0        11A
   UCOPY1         B710         5C
   UCOPIV         B770        1C4


   IEAXPALL       B938        304
   IEAXKALL       BC40        7C0
   MAIN           C400         F0
   QUAD           C4F0        1DA
   STA371         C6D0        402
   DACCES         CAD8        62E
   DEFINE         D108         E0
   SEQUEN         D1E8        516
   IHOSCOS *      D700        234


     NAME       ORIGIN    LENGTH
   IHOFQTEN*      D938        110

   IAND    *      DA48         3C

   IHOECOMH*      DA88        E30

   FIOAP#  *      E888        61C
   IHOCOMH2*      EED8        9A5

   IHOUATBL*      F880        638
   IHOQATN2*      FEB8        4B4

   IHOSSIN *     10370        244

   IHOSSQRT*     105B6        174

   IHQFCVTH*     10730        CFA


   IHOEFNTH*     11430        800

   IHOERRM *     11C30        624

   IHOQCONI*     12258          4
   IHOQCONO*     12260          4
   IHOUOPT *     12268        538
   IHOFCONI*     127A0        416


   IHOFCONO*     12BB8        8B8

   IHOETRCH*     13470        2AE

   IHOFTEN *     13720        220

   CSWTAB        13940        400
   CIBWAT        13D40          4
   TSTCOM        13D48          4
   DEFILC        13D50          8

  ENTRY ADDRESS      C400
  TOTAL LENGTH      13D58
****E370MAIN  NOW REPLACED IN DATA SET
AUTHORIZATION CODE IS         0.
```

Fig. 10 cont.

LINKAGE EDITOR

```
JOB         :F1BNOTCO
TIME        :    3 MIN
START TIME  :29/04/85 11.27.24

MODULE NAME:F1BNOT.TSOLIBL(E370MAIN)
LIST FILE   :F1BNOT.LIST370E
BLOCKSIZE=   963 LRECL=   137 IRECFM=  84
//GO.FT09F001 DD DSN=F1BNOT.LIST373E
BLOCKSIZE=  3500 LRECL=   137 IRECFM=  84
//GO.FT08F001 DD DSN=F1BNOT.COMF8
BLOCKSIZE=   400 LRECL=    80 IRECFM= 144
//GO.FT12F001 DD DSN=F1BNOT.DATEST1
BLOCKSIZE=   100 LRECL=   100 IRECFM= 128
//GO.FT13F001 DD DSN=F1BNOT.DATEST2
BLOCKSIZE=   100 LRECL=   100 IRECFM= 128
//GO.FT08F001 DD DSN=F1BNOT.TSOLIBL(E370MAIN)
BLOCKSIZE=  6233 LRECL=     0 IRECFM= 192
LENGTH OF MODULE     00013D58 (HEX)
ENTRY POINT          0000C400 (HEX)
FT 8F001 DD DCB=(LRECL=   84,BLKSIZE= 3952)
FT12F001 DD DCB=(LRECL=  104,BLKSIZE= 3956)
FT13F001 DD DCB=(LRECL=  104,BLKSIZE= 3956)
FT 6F001 DD DCB=(LRECL=  137,BLKSIZE= 4000)
FT 9F001 DD DCB=(LRECL=  137,BLKSIZE= 4000)


REAL * 16:          2.0000000000          4.0000000000          8.0000000000
                                                                 3.1415926536

START DIRECT ACCESS TEST
WRITE TO 12
NEXT RECORD IS:IV=    2
NEXT RECORD IS:IV=    3
NEXT RECORD IS:IV=    4
NEXT RECORD IS:IV=    5
NEXT RECORD IS:IV=    6
WRITE 4 RECORDS TO 13 STARTING RECORD 2
 NEXT FREE RECORD ID    6
 WRITE 5 RECORDS TO UNIT 13
 NEXT FREE RECORD ID    2
 NEXT FREE RECORD ID    3
 NEXT FREE RECORD ID    4
 NEXT FREE RECORD ID    5
 NEXT FREE RECORD ID    6
 WRITE RECORD 3 ON UNIT 12 AND RECORD 1+2 ON UNIT 13
NEXT RECORD IS:IV=    4
 NEXT FREE RECORD ID    3
READ FROM 12
NEXT RECORD IS:IV=    2
NEXT RECORD IS:IV=    3
NEXT RECORD IS:IV=    4
NEXT RECORD IS:IV=    5
NEXT RECORD IS:IV=    6
WRITE 4 RECORDS TO 13 STARTING RECORD 2
 NEXT FREE RECORD ID    6
READ FROM 13
 NEXT FREE RECORD ID    2
 NEXT FREE RECORD ID    3
 NEXT FREE RECORD ID    4
 NEXT FREE RECORD ID    5
 NEXT FREE RECORD ID    6
 FIND RECORD ON UNIT 13
 NEXT FREE RECORD ID    4


 SEQUENTIAL INPUT/OUTPUT TEST
    1.00    2.00    3.00    4.00    5.00    6.00    7.00    8.00    9.00   10.00
   11.00   12.00   13.00   14.00   15.00   16.00   17.00   18.00   19.00   20.00
   21.00   22.00   23.00   24.00   25.00   26.00   27.00   28.00   29.00   30.00
   31.00   32.00   33.00   34.00   35.00   36.00   37.00   38.00   39.00   40.00
   41.00   42.00   43.00   44.00   45.00   46.00   47.00   48.00   49.00   50.00
   51.00   52.00   53.00   54.00   55.00   56.00   57.00   58.00   59.00   60.00
   61.00   62.00   63.00   64.00   65.00   66.00   67.00   68.00   69.00   70.00
   71.00   72.00   73.00   74.00   75.00   76.00   77.00   78.00   79.00   80.00
   81.00   82.00   83.00   84.00   85.00   86.00   87.00   88.00   89.00   90.00
   91.00   92.00   93.00   94.00   95.00   96.00   97.00   98.00   99.00  100.00

1
12
123
1234
12345
123456
1234567
12345674
123456745
1234567456
FINISH SEQUENTIAL
```

Fig. 10 cont.

OUTPUT on unit 6

```
SIN COS TABLE
  20    0.3420     0.9397    0.3491
  40    0.6428     0.7660    0.6981
  60    0.8660     0.5000    1.0472
  80    0.9848     0.1736    1.3963
 100    0.9848    -0.1736    1.7453
 120    0.8660    -0.5000    2.0944
 140    0.6428    -0.7660    2.4435
 160    0.3420    -0.9397    2.7925
 180    0.0000    -1.0000    3.1416

IHO209I IBCOM - PROGRAM INTERRUPT (P) - DIVIDE CHECK OLD PSW IS  8000000FA200C998

TRACEBACK   ROUTINE   CALLED FROM ISN   REG. 14    REG. 15    REG.  0    REG.  1

            STA371             0005     4200C4C0   0000C6D0   0000000C   00000000

            MAIN                        60001B4C   0000C400   00000010   000FF210

ENTRY POINT=  0000C400

STANDARD FIXUP TAKEN , EXECUTION CONTINUING
AFTER DIVIDE CHECK    3.00000    0.0      **********

IHO209I IBCOM - PROGRAM INTERRUPT (P) - DIVIDE CHECK OLD PSW IS  80000009A200C9D4

TRACEBACK   ROUTINE   CALLED FROM ISN   REG. 14    REG. 15    REG.  0    REG.  1

            STA371             0005     4200C4C0   0000C6D0   0000000C   00000000

            MAIN                        60001B4C   0000C400   00000010   000FF210

ENTRY POINT=  0000C400

STANDARD FIXUP TAKEN , EXECUTION CONTINUING
AFTER FIXED DIVIDE       5      0      5

IHO207I IBCOM - PROGRAM INTERRUPT (P) -   OVERFLOW   OLD PSW IS  8000000C4200CA0A

TRACEBACK   ROUTINE   CALLED FROM ISN   REG. 14    REG. 15    REG.  0    REG.  1

            STA371             0005     4200C4C0   0000C6D0   0000000C   00000000

            MAIN                        60001B4C   0000C400   00000010   000FF210

ENTRY POINT=  0000C400

STANDARD FIXUP TAKEN , EXECUTION CONTINUING
OVERFL  0.10000C0E+61  0.7237005E+76

IHO208I IBCOM - PROGRAM INTERRUPT (P) -  UNDERFLOW   OLD PSW IS  8000000D4200CA3A

TRACEBACK   ROUTINE   CALLED FROM ISN   REG. 14    REG. 15    REG.  0    REG.  1

            STA371             0005     4200C4C0   0000C6D0   0000000C   00000000

            MAIN                        60001B4C   0000C400   00000010   000FF210

ENTRY POINT=  0000C400

STANDARD FIXUP TAKEN , EXECUTION CONTINUING

IHO251I SQRT ARG=-0.1000000E+01, LT ZERO

TRACEBACK   ROUTINE   CALLED FROM ISN   REG. 14    REG. 15    REG.  0    REG.  1

            SQRT               0032     4200CA48   00010720   00000060   88000000

            STA371             0005     4200C4C0   0000C6D0   0000000C   00000000

            MAIN                        60001B4C   0000C400   00000010   000FF210

ENTRY POINT=  0000C400

STANDARD FIXUP TAKEN , EXECUTION CONTINUING
AFTER NEGATIVE SQRT         1.00000
BEFORE ADDRESS VIOLATION
XTRACE WAS CALLED VIA STAE370E WITH FLAG=        00000000

ADDRESSING             0C5

UNKNOWN INTERRUPT
PSW=   8200CA84  IL+CC= 80000005
TRACEBACK   ROUTINE   CALLED FROM ISN   REG. 14    REG. 15    REG.  0    REG.  1
            YPRERR                      00119F6E   00009510   00000000   00008700
            FTRACE                      4200944E   00008708   00000019   00000000
            XTRACE                      62001C2A   00009080   00000000   00001C84
EXECUTION TERMINATED
 END OF JOB 29/04/85  11.29.36

 TEXT FOR UNIT 9 BEFORE ENDFILE
```

Fig.10 cont. UNIT 6 and UNIT 9 output

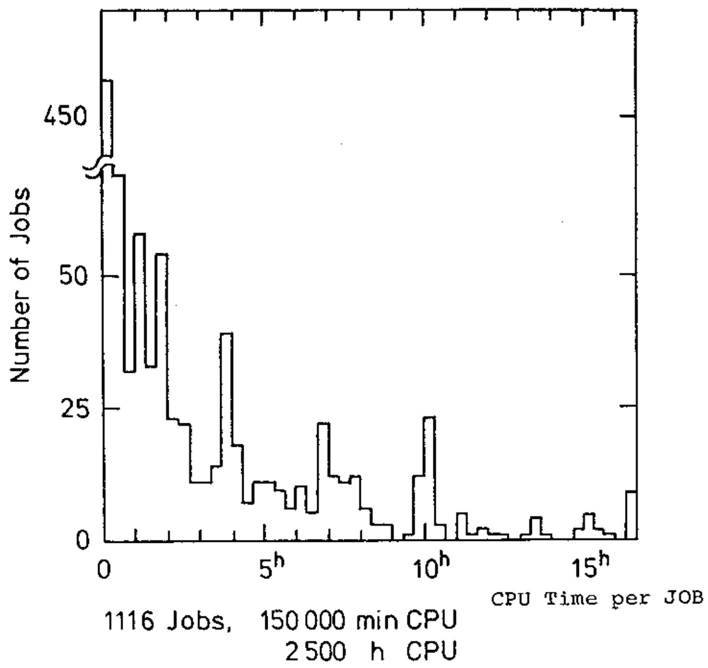


Fig. 11. Job profile of 1116 jobs executed during 1984.