IBM®

1620 DATA PROCESSING SYSTEM

MONITOR I PROGRAMMING SYSTEM

Education Guide

Education Development—*San Jose, California*

Copies of this publication can be obtained through IBM Branch Offices.
Address comments concerning the contents of this publication to:
IBM DPD Education Development, Education Center, San Jose, California.

TABLE OF CONTENTS

SECTION I

INTRODUCTION

INTRODUCTION

The 1620 Monitor I programming system should be learned in two phases. The first phase should provide an understanding of the need for and the advantages of any monitor, introduce the job and control card concepts, and identify the components of the system and the way they support each other. The second phase should provide the ability to use Monitor I on an IBM 1620/1311 system to accomplish typical program assemblies and executions and disk storage functions.

Both phases of learning the Monitor can best be realized by using a 1620 disk system, first for demonstration and then for testing problem solutions. If no 1620 is available, the course should still be divided into two phases, and the second phase should still be heavily problem-centered. This requires an instructor who has had firsthand experience with Monitor I. Students cannot learn Monitor I using a detail-centered approach.

The course will be greatly improved if students bring their own SPS and FORTRAN problems to the class.

Monitor I greatly increases throughput on an IBM 1620/1311 system and greatly simplifies disk storage utilization. These advantages must not be obscured by the instructor's attitude or method of presentation.

BASIC TEACHING PRINCIPLES

Here are a few basic points to consider and remember before teaching. They touch only the surface of the task but will save you from gross errors.

How to Employ the Principles of Learning

Motivation

1. Show a need
2. Use students' motives
3. Develop intent to learn
4. Maintain interest
5. Encourage early success
6. Give recognition and credit
7. Avoid adverse feelings and emotional barriers to learning
8. Use competition
9. Use rewards and punishments when needed

Objective

1. Tell the student what he is to learn and what is expected of him
2. Present each unit as a part of a whole

Doing

1. Plan for student activity
2. Devote as much time as possible to student practice
3 Ask questions
4. Use problems

Realism

1. Teach in terms of field conditions
2. Stay on the level of the class
3. Explain subject in terms of how it will be used by the student

Background

1. Build on what the student already knows
2. Use review frequently
3. Draw on class experience for illustrations, stories, and examples

Incidental Learning

1. Teach the student - not just the subject
2. Set a good example
3. Remember you determine his attitude

Tips on Delivery

Contact

1. Secure attention
2. Look at and talk to the class
3. Be conversational
4. Don't talk down to the class
5. Show a genuine interest in your students

Bearing

1. Check appearance
2. Watch your posture
3. Make movements meaningful

Mannerisms

Avoid those things which cause the class to concentrate on you and what you are doing rather than on your communication

Tips on Delivery

Enthusiasm

Present instruction forcefully, enthusiastically, but naturally

Emphasis

Use repetition, gestures, pauses, and variations in rate, pitch and intensity

Volume

Make sure you can be heard

Be Understood

1. Speak the language of your class
2. Be sure your ideas are getting across
3. Enunciate clearly and pronounce correctly

Don't Make Excuses

Don't apologize; You can't sell your subject with a negative attitude

Proper Use of Training Aids

1. KEEP AIDS OUT OF SIGHT WHEN NOT IN USE
2. Select the appropriate aid
3. Prepare for use of the aid
4. Explain aid to the class
5. Show so all can see
6. DO NOT obstruct students' view
7. Use a pointer if necessary
8. Talk to the class - not the aid
9. Use assistants to best advantage
10. Display aids smoothly

Proper Use of Training Films

1. Preview critically and select film for specific instructional purposes
2. Check equipment
3. Introduce film properly; Tell students what it is about and what they should look for
4. Follow-up by practical application, oral discussion, or short quiz

How to Use the Blackboard

1. Check on equipment to be used
2. Check for glare
3. Keep chalkboard clean
4. Don't crowd your work
5. Plan your work in advance
6. Keep material simple and brief
7. Print and draw on large scale
8. Erase unrelated or used material completely
9. Use color for emphasis and variety
10. Prepare complicated illustrations beforehand

Tips on Questions and Quizzes

Ask questions which:

1. Have a specific purpose
2. Are clear and concise
3. Emphasize one point only
4. Require definite answers
5. Are phrased so as to discourage guessing
6. Are related to the how and the why
7. Stress the vital points of the instruction

Tips on Questions and Quizzes

Answering questions:

1. Be sure you understand the question
2. Repeat the question aloud before answering

Quizzes

1. These are really slightly more formal questioning; You should make up one or two brief quizzes to check on student understanding of your material

SECTION II

COURSE DESCRIPTION

IBM

Course Description

1620 DPS MONITOR I PROGRAMMING SYSTEM

Provided for -

People who will be programming and operating the IBM 1620 using the 1620 Monitor I System.

Objectives -

Upon successful completion of the course, the student is able to:

1. Discuss accurately the advantages and functions of the 1620 Monitor.
2. Prepare control cards used in the various jobs that can be run under Monitor control: SPS, SPSX, FOR, FORX, XEQ and DUP.
3. Dump and analyze the system tables and any other disk data.
4. Prepare control cards that will store or call SPS or FORTRAN object programs from disk storage.
5. Write programs using the SPS and FORTRAN disk input/output macros to put or get data.

Prerequisites -

Students enrolled must have successfully completed the 1620 Basic Programming Course, including a study of FORTRAN and the IBM 1311 Disk Storage Unit.

Course Code -

P 2742

Copies of this and other IBM publications can be obtained through IBM Branch Offices. Address comments concerning the contents of this publication to *IBM DPD Education Development, Education Center, San Jose, Calif.*

PRINTED IN U.S.A.

R20-9036-0

Material Required for Conduct of Course -

Student Materials (Normally one copy per student)

Title	Form No.	Abstract Ref.
1620 Monitor I System Specifications	C26-5719	SRL 1620-00
1620 Monitor I System Reference Manual	C26-5739	SRL 1620-00
1620/1443 Monitor I System Specifications	C26-5757	SRL 1620-00
1620/1710 SPS and Absolute Coding Sheets	X26-5627	SRL 1620-00
FORTRAN Coding Forms	X26-7327	SRL 1620-00
Student Handouts		See below

Instructor Materials

1620 Monitor I System Education Guide	R20-8005	See below
1620 Monitor I System (card)	1620-PR-025	

Abstract -

R20-8005 1620 Monitor I Programming System Education Guide

8 1/2 x 11 Looseleaf Instruction Outline - 150 pages
(Brown cover)

This guide contains two alternative course outlines to be used in teaching a five-day programming course (three days for IBM classes). Included in the guide are reproductions of a set of 18 flipcharts used in conjunction with one of the outlines and an extensive set of student handouts to be used with the alternative outline. Flipcharts and handouts are to be reproduced locally.

Major topics included in the guides are: Introduction to the Monitor; JOB Concept; Supervisor; Disk Utility Programs; New SPS Features and Macros; FORTRAN Disk Macros, and FORTRAN Data Storage.

SECTION III

GENERAL COURSE OUTLINE 1

1620 MONITOR I TOPICAL OUTLINE
WITH VISUALS

I Introduction to Monitor
- Monitor Concept
- Features of Monitor
- Advantages

Structure of the Monitor

II SPS Processor
- New Features
- I/O Macros
  - GET
  - PUT
  - SEEK

III Format of Object Program
- Other Macros
  - CALL EXIT
  - CALL LINK
  - CALL LOAD

IV SPS Control Cards
- Notes on Assembly

V FORTRAN
- Data Organization
- New Macros
  - DEFINE DISK
  - FIND
  - FETCH
  - RECORD
- Method of Data Reading
  - Matrix versus Element Input and Output

VI FORTRAN Control Cards
- FORTRAN Processor Notes

VII Disk Utility Programs
- DIM Table
- Equivalence Table
- Sequential Program Table
- DELETE and DEFINE PARAMETERS

VIII JOB Concept
- JOB
- END OF JOB

Other Control Cards
- TYPE
- PAUS

IX Review Disk GET and PUT
- Additional Operands

X SPS II-D Modification Program
- Op Code Definition Cards

XI Addition of SPS Subroutines

SECTION IV

DETAILED COURSE OUTLINE 1

I Introduction

Organization of Presentation.

Start off with a description of the system and how to use it. Program using Monitor. Then go back over the material in more detail and look at the internal workings of the processor - what is generated, how the loader works, etc. Hold off questions in this area until then.

1. Discuss Monitor Concept

To control the operation of several unrelated routines and machine runs so that the computer and computer time are used advantageously.

Something always wanted on 1620 - available on large scale - now 1311 makes it possible!

FLIP CHART 1 - before Monitor
FLIP CHART 2 - with Monitor

2. Features

Somewhat similar to Autotest on 1401- i. e., stack jobs with control cards.

FLIP CHART 3

permits FORTRAN compilation
FORTRAN compilation and run
SPS assembly
SPS assembly and run
SPS run of previously assembled
FORTRAN run of previously assembled

3. Advantages

FLIP CHART 4

What do we have to pay for these advantages? Very little!

Machine requirements:

- only 20K
- at least one 1311
- indirect addressing
- auto divide
- either card or paper tape I/O

Control cards

- must make up some for each operation; quite straightforward

Some core occupied by Monitor at all times, but this is offset by segmenting ability

4. Structure of Monitor

FLIP CHART 5

Four parts: Supervisor
DUP
SPS
FORTRAN

Supervisor - overall control of Monitor
- reads and analyzes control cards
- calls from disk the appropriate program
- always on disk; if it gets lost, can load one card to recall it and start over

Disk Utility Programs - DUP
- set of commonly needed utilities such as dump, load programs to disk, delete programs; discussed later in detail

SPS and FORTRAN

- the normal type processors, but new and more powerful
- convert source statements to machine object programs
- make use of some of DUP for assembly options

II SPS Processor

1. Discuss New Features Unique to the Processor

a. B7 and BB2

- could accomplish this by following with DORG, but this is simpler for programmer

B7 ⟶ B xxxxx
DORG* - 3

BB2 ⟶ BB
DORG* - 9

b. Can have blanks in flag operand

c. New declarative

DVLC - permits programmer to set up a series of adjacent constants; not necessary they be same length

FLIP CHART 6

- a symbolic length may be used and must be previously defined
- constants may be symbolic and need not be previously defined

d. Don't need to, but may type RM after an instruction from typewriter

e. DNB can be up to 99

f. No SEND or ALOW

ALOW is instruction for 1710 which unmasks during loading to permit interrupt.

g. DSAC - same as DAC, but addresses low-order digit.

h. The really significant changes are I/O macros
GET, PUT, SEEK

2. I/O Macros

FLIP CHART 7

GET - must convey to processor
device
mode - alpha or numeric
I/O area

Rather than specify in macro, operand references a "definer" which is a constant conveying the information.

Discuss non-disk first.

Example:

```
          |GET |DEF1
. . . . . |. . | . . . . . .
DEF1      |DCA |, IOAREA
IOAREA    |DAS |80
```

The definer op. code DCA specifies card I/O and alphabetic mode. The operands are:

- the address at which constant is located (normally omitted)
- the address of I/O area

There are other definer op. codes. This generates:

```
DSA  IOAREA
DC   3,IO@
```

Other definers generate different const.

- GET, PUT, SEEK for disk
- simplest type - no relocation
  - nothing on generation

	GET	DEF
DEF	DD(W)	,DCNTL
DCNTL	DDA	
	DC	1,@

For disk use DDA, not DAS or DSS.

Review DDA

FLIP CHART 8

- DDA entry may be used to define disk control field - has 5 operands:

(1) address of constant - must be even; may omit
(2) drive over-ride code - even or zero - no override
(3) 5-digit sector address
(4) number of sectors
(5) I/O storage area - even

For Monitor, must follow DDA with ‡. If reach end of cylinder, with more sectors to go, will automatically increment over to next cylinder and proceed.

PROBLEM - set up macros and any required declaratives to:

- Read a card in numeric in IO1
- Type an alpha message for area IO2
  35 characters
- Position the arm and then read 3 sectors from disk beginning at sector 1 2 3 4 5, w/o WLR into I/O3

SOLUTIONS

(1)

```
          GET   DEF1

DEF1      DCN   ,IO1
IO1       DSS   80
```

(2)

```
          PUT   DEF2
. . . . . . . . . .

DEF2      DTA   ,IO2
IO2       DAS   35
          DAC   1,@
```

(3)

```
          GET   DEF3
. . . . . . . . . .

DEF3      DD    ,DCNTL,,,A
DCNTL     DDA   ,0,12345,3,IO3
          DC    1,@
          DORG  * + 2
IO3       DSS   300
```

III Format of Object Program

- core image
- systems output format (reloadable)
  - absolute
  - relocatable
    - rules for relocation
    - no DORG permitted with absolute address
      DORG at zero

    Why relocatable:
    - permits programs to be in core in different locations at different times
    - no problem to code - handled automatically

Other macros besides GET, PUT, SEEK

CALL EXIT - this is last executed step in users SPS program and generates a return to the supervisor to process the next job. Generates a Branch MONCAL.

CALL LINK, xxxxxx (or NNNN)
- provides the segmenting ability
- pulls in the program named from disk and begins executing
- may, if relocatable program, specify a relocation address
- if not, will stack them in successively higher core, starting at 2402

CALL LOAD, xxxxxx (or NNNN)
- pulls in the program, but does not execute

IV SPS Control Cards

- precede SPS deck and govern manner in which assembly is performed
- no longer controlled by sense switches

Give example: Want to assemble SPS, store back on disk, type symbol table and give program a name so can execute later, and stop and enter new statement on error.

```
* TYPE SYMBOL TABLE
* STORE RELOADABLE
* NAME CRPATH
* ERROR STOP
```

Discuss other control cards except *SYSTEM SYMBOL TABLE
*LIBR
*NO SUBROUTINES
*ID NUMBER

- may have no SPS control cards
- discuss Monitor control cards - which call SPS processor in to analyze SPS control cards

SPS
SPSX
XEQ

- show examples of assemble, assemble and execute, execute

Notes on New Assembly

- 4K symbol table in core - 235 max
- if overflow, shifts this batch to disk and builds 4K more
- fixed length entries (17 digits)
- search whole table for mult-defined - slower if must go to disk
- additional storage - all for symbol table
- intermediate output to disk unless 2 pass
- all final output to disk only; output is by DUP, automatically

V FORTRAN

- get all capabilities of FORTRAN II - i. e. sub-programming
- only change to language specifications is inclusion of disk commands

1. Data Organization on Disk

- can specify ff- and kk- length of fix and float
- can specify how many words to be stored per record
- a record consists of up to 200 characters
- both fixed and floating are stored the same size, whichever is max.
- if this max. size X number of words/record < 100, one record is equivalent to 1 sector
- if max. size X number of words/record > 100 and < 200 (limit), then 1 record is equivalent to 2 sectors

- must organize the data in the order in which it is to read - like a tape record where must pass whole record to pick out any one field
- each data record is numbered, starting with 1 and on up, and data is addressed by this record number (FLIP CHART 9)
- record 1 is 218 sectors from beginning of work cylinder - 0-23 normal

2. New Macros

DEFINE DISK

Form DEFINE DISK (n_1, n_2) where n_1 is number of words per record; n_2 is total number of records to be reserved for data - used for cylinder reservation.

Note: If make n_2 too large, processor automatically cuts it back to maximum allowable.

Define disk in mainline only, not in sub-programs

I/O Commands : since data is written on disk in constant length, i.e., both fixed and floating are the same size, there is no need for a format statement to be associated with disk reading and writing.

The three I/O commands are:

FIND
FETCH
RECORD

} FLIP CHART 9A

Form -FETCH(I), A, B, C, D, E
- where I has a value from 1 to n_2 and refers to the record number
- I may be fixed pt. variable, subscripted or not

- depending upon the value of n_1 (no. words/record), several records (sectors or pairs of sectors) may be read before the list is satisfied.

Example:

- if $n_1 = 2$, then records I, I+1, and I+2 would be read.

Note: If list is not satisfied at end of one record, the value of I is actually incremented by 1 in core and reading resumes. At the end of the read, when the list is satisfied, I will be incremented to 1 past the last record read.

Problem

Given ff = 12, kk = 4, $n_1 = 10$

a. how many sectors per record?
b. what would be most efficient value for n_1 given f and k?
c. if, for example, I = 6 (FETCH (I), A, B, C, D), what will I be after operation?

Solution

a. 2 length = 12 + 2 = 14 * 10 = 140
b. 7 7 x 14 = 98
c. I = 7 (has read only 1 record; increments by 1 even though 1 record = 2 sectors)

Note: Since FETCH(I) causes value of I to increment, should not use this as index of DO containing FETCH.

Example:

```
      DO 10 I = 1, 10
10    FETCH(I), A(I)
```

} Invalid

Rather:

```
      I = 1
      DO 10 J = 1, 10
10    FETCH(I), A(J)
```

Here A(J) is first word in each record (if n=1, only 1) and I is automatically incremented.

Note: FETCH(I) includes a seek to the proper cylinder, if necessary. If not, reading begins at once. If seek is required, seek time is interlocked.

FIND

To avoid this interlock, may wish to precede FETCH by a seek or FIND(I). Processing can then be carried out while arm is being positioned.

Example:

```
I = 10
FIND(I)

    calculate
    previous
FETCH(I)
```

However:

FIND only initiates a seek if it is needed; this is also true of FETCH and RECORD.

If disk operation occurs between FIND and FETCH (this could be initiated with the assemble inst.), then will seek new location, read or write, then reseek. Unless very sure which cylinders data is on, don't use FIND. It may actually slow down program.

(End of first day in two-day class)

Method of Data Reading

- when reading or writing a list of elements, obviously must set up an I/O area (beginning in even location) and build the record there
- this involves data movement within core as well as core-disk movement
- can avoid this if output entire matrix at a time, as do in READ and PUNCH - i.e., FETCH(I), A where A is a singly, doubly, or triply defined matrix, and where I is record number of beginning
- this is much faster
- in order to read directly into matrix area, core location must be even; this is assured only if ff and kk are both even
- matrix write packs data without regard for n
- may even be faster to set up dummy matrix - e.g., to output 3 values, set up X(3), move values to matrix and RECORD(I), X
- if write with matrix, must read back in same fashion
- cannot mix element and matrix reads in one I/O command
- matrix operation does WLR check thru GM stored on disk in order not to overlay beyond the dimensioned area
- can work only on pack upon which work area is defined to fall

EXIT

- pause will generate as normal; stops, then goes to next step
- STOP, generates halt and then a BRANCH to MONCAL
- CALL EXIT - branches to MONCAL; i.e., back to Monitor with no halt

CALL LINK

- similar to call link in SPS
- used to pull in another mainline core load and branch to the first executable program
- not used to call subprogram; this is done in normal fashion with CALL SUBPROGRAM

Note: No CALL LOAD available in FORTRAN. In order to pass data from one FORTRAN program to another (when the two are not executed in sequence), must use a DCOPY prior to executing the FORTRAN to make the data available in the work cylinders. Must be sure and not place it in work cylinders where it would be destroyed during FORTRAN program loading. Certainly no less than 218th sector of work, since this is always record 1.

VI FORTRAN Control Records FLIP CHART 10

FANDK - vary length, at compilation time, of floating and fixed 2 - 28, 4 - 10; can, at execution, specify a smaller f and k if desired

PSTSN - punch out symbol table and addresses of numbered statements

POBJP - punch object program in system output format

LDISK - store program on disk by name; don't need name on FUNCTION or SUBROUTINE subprogram

DATA - must follow program for execution before data can be entered (not needed SPS), and even if no data

LOCAL - specifies that certain subprograms are to be loaded on call only; many can occupy same storage area one at time

Form: *LOCAL, MAIN, SUB1, SUB2, SUB3

- number of LOCAL cards specified in FORX cd., or XEQS card
- maximum of 50 per mainline
- LOCAL can't call subroutines or subprograms not already in CORE; may desire dummy call in mainline

FORTRAN Processor

- two passes internally
- have option of two subroutine sets for fp., hardware or not, short form occupies up 7500 and long form to 14000
- use long if possible, otherwise short; long is faster - doesn't have to keep going back to disk
- all links in a program will use same subroutine set
- statement numbers now direct branches; no longer branch to branch
- subscripting routines are new
  - literal subscripts handled at compile - faster
  - double and triple subscripting not in line; go to subroutine (same routine, two enters); slower, but shorter

Overflow Storage

If get overflow:

- use short subroutines
- pull subprograms and make LOCAL
- segment mainline - CALL LINK

VII Disk Utility Programs

- a series of handy utilities written and included under control of the supervisor
- relieves programmer of costly, time-consuming burden
- called by Monitor card DUP

Routines - FLIP CHART 11
- performed as part of assembly or
- coded control card calls them

1. Disk Identification Map - DIM

a. used by all phases of the Monitor to find data and programs in disk

b. on cylinder 24, up to 999 20-digit entries; can expand to 5 cylinders - 4995 entries

c. format - FLIP CHART 12

2. Equivalence Table

a. no names in DIM entry

b. have a table giving DIM number equivalent to a named program

c. form 12 for 6 alpha; 4 DIM number 0001

d. cylinder 25 - 80 sectors - 500 names

e. first 51 are used by systems

f. can add to end or delete from middle and squeeze up

g. DIM X 2 + 048000: 5 left = sector; right 1 = entry

3. Sequential Program Table

a. cylinder 99 on each pack - sector 1 - 80

b. lists contents of pack and empty places
(if they ask about format, see Ref. Manual)

Pack Label: sector 0 of cylinder 99

Monitor Core Req. : 00100-02401 - IORT or LOADER

Update DIM

- automatically handled when causes program to be loaded or DUP f'n to be performed
- see chart - pg. 25 of Monitor booklet
- see above (pg. 25) ID number SPS control

4. Examine Two DUP Programs in Detail

DELETE

a. Function
- to delete a program along with its associated DIM equiv. and seq. program entries
- does not fill in space, but will come back and drop in a suitably sized one

b. Control Card

*DELET XXXXXX NNNN

Either one - not both

DEFINE PARAMETERS

a. Function - to define size and layout of areas on disk and establish other stnds

Normal	WORK	00-23
	DIM	24
	EQUIV	25

b. Control Card - - - FLIP CHART 13

VIII JOB Concept

Several Monitor control cards we haven't looked at yet.
Intermix systems:

MONITOR CONTL.
SPS, FORTRAN or DUP CNTL.
SOURCE PROG.
DATA
etc.

1. JOB concept - enables us to tie together several programs.
Example: If PROG 1 calls PROG 2, then can't execute either until both are assembled.

a. want supervisor to treat the two programs as one job
b. use Monitor control card JOB and END of JOB (ǂ ǂ ǂ ǂ)
c. may make each program a separate job

FLIP CHART 14

d. if prog. error in job, does not execute any others in that job, but will continue assembly and do SPS output

2. Other Control Cards

TYPE - calls for control cd. from type
PAUS - halt, permit switch setting, etc.; handy in FORTRAN

Comments - ǂ ǂ ———— Col. 7 - 80 ~~~~
follow by another control

- module # on drive - to flip flop disk
- shouldn't use last card check; use trailer card instead

(Go back to some features we skipped.)

IX Review Disk GET and PUT

- can specify for disk PUT - RBC
- a disk GET or PUT references a DDW or DD
- we have looked at two operands; e. g. -

|DD |, CON1

DD may have additional operands

- relocation address
- reposition RW heads
- actual or relative sector

|DD |, CON1, 16000, R, A

generation:

short form $M_0 M_1$ XXXXX ‡

long form $M_0 M_1$ XXXXX XXXXX ‡

address of DDA — relocation address

Values of M_0, M_1

M_0

relocatable

- 0 - due to no entry A - make sector address in DDA relative to start of work cylinder; this const. stored in 422 - 425
- 1 - update "high storage used" indicator as well as zero

absolute
- 2 - use sector address as specified
- 3 - use sector address and update high
- n̄ - flag over M_0 says reposition to a specified cylinder

M_1

sector mode
- 0 - with WLRC - user puts ⫩ at end of data in core
- 2 - without WLR
- 4 - track mode with WLRC - user puts ⫩ at end of data in core
- 6 - track mode without WLRC

All possible combinations can't be achieved thru macro's - may have to code the linkage directly

Example: track mode

X SPS II-D Modification Program

Functions:

- to modify the SPS processor
- add or delete mnemonic op. codes
- add or delete SYSTEMS SYMBOL TABLE items

To call : stored in Monitor as SPSLIB

Monitor control card: XEQ SPSLIB

SPS Mod. Control: FLIP CHART 15

Op Code Definition Cards - FLIP CHART 16

Example: to set up mnemonics to write on printer

write numeric: 38 PPPPP Q09QQ

use XY3

| PRN | 893 | 3⑧PPPPP Q0⑨QQ |

write alpha:

| PRA | 993 ——→ 3⑨PPPPP Q0⑨QQ |

Note: various sizes of mnemonic op. codes have max. allowable. Example:

~ 16 3 digit
~ 14 4 digit

If get message NO ROOM, try again with different number of digits in mnemonic - then may fit.

Note: DELETE OP CODE - not cricket to delete such codes as TF, etc.

Define System Symbol Table

- permits commonly used symbols to be always defined; not required to be defined in users program
- system already has some, may add others, up to 150

Example:

*DEFINE SYSTEM SYMBOL TABLE

6 - 11	12 - 15	16 ——— 75
XXXXXX	blank	absolute address or symbolic operand already in SST

- if use positive operand - absolute
- if use negative operand - relocatable

Example: | PROD | 00099

- when define new one, must redefine old as well since immediately deletes all user-defined

XI Addition of SPS Subroutine Macros

- modify op. code table to include new mnemonic
- write subroutine in SPS
- assemble relocatable and store on disk

See FLIP CHART 16

- subroutine scheme is changed - PICK time reduced from 9 to 5 ms.
- Control cards to assemble MACRO sub

FLIP CHART 17

a. now 17 subroutines; may add 12 more, #18 to 29 - space reserved in DIM table for these.

b. to calculate DIM no. for ID no. entry, take base no.

130 - fixed divide (subroutine set 01)
100 - fixed length (sub set 02)
70 - variable length (sub set 03)
40 - auto f. pt. hardware (sub set 04)

and add subroutine # (18 - 29) to this

Example: first new variable length = #18
70 = 88

DIM entry for subroutine - FLIP CHART 18

Practice Problem

What are control cards necessary to enter macro XYZ, the second new macro, into OPCODE TABLE ?

Solution

```
‡‡JOB
‡‡XEQ SPSLIB
* DEFINE OP CODE
            |XYZ |- 191
* ENDLIB
‡‡‡‡
```

- if not familiar with writing subroutines, use of PICK, etc., let's look at this later.

SECTION V

FLIPCHARTS FOR OUTLINE 1

HEADER

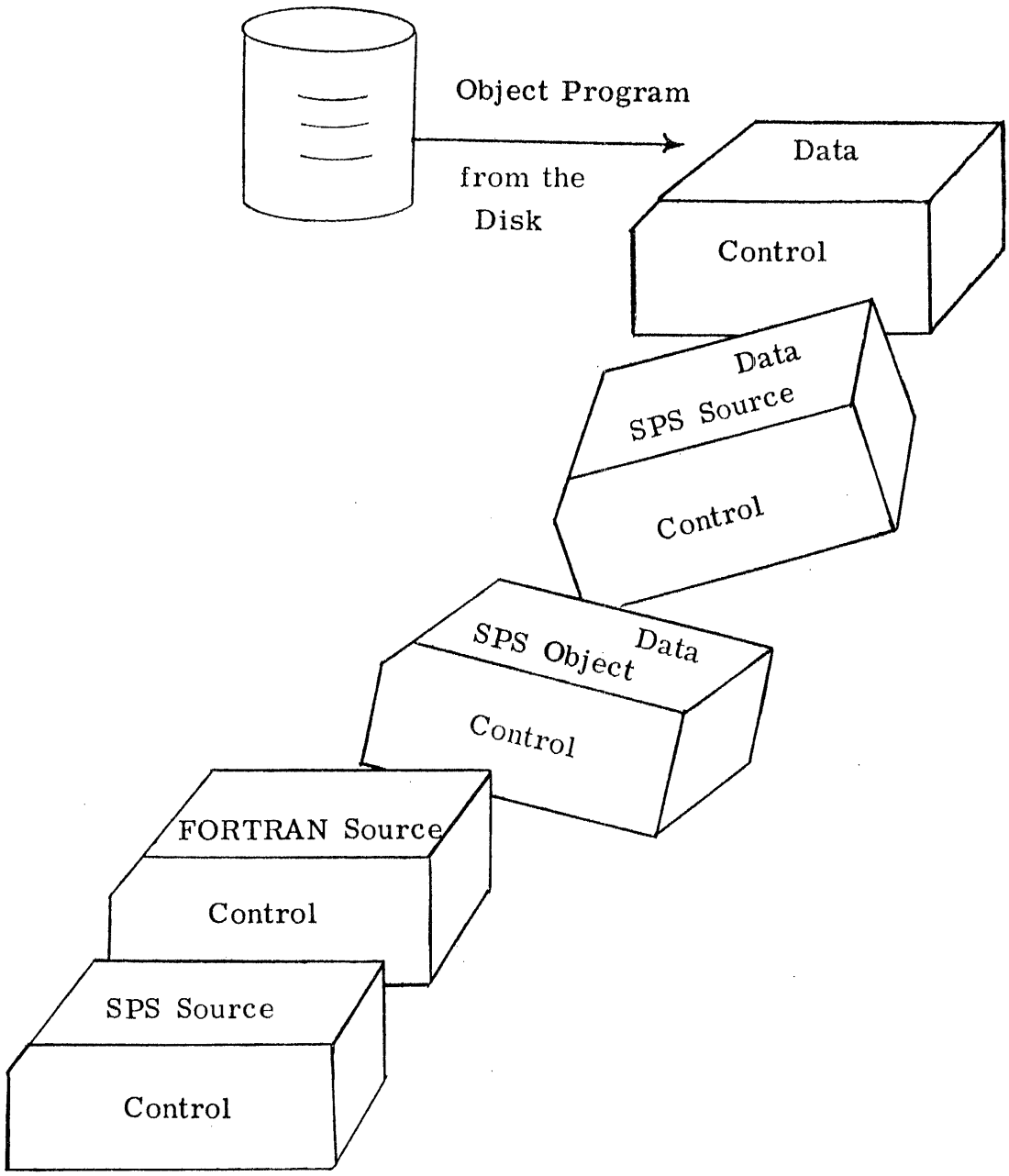
1620 MONITOR

FLIP CHART 1

1620 with many people running around and cards flying
(TURMOIL)

FLIP CHART 2

1620 plus 1311 - no people
(ORGANIZED)


Object Program
from the
Disk
Data
Control
Data
SPS Source
Control
SPS Object
Data
Control
FORTRAN Source
Control
SPS Source
Control

FLIP CHART 3

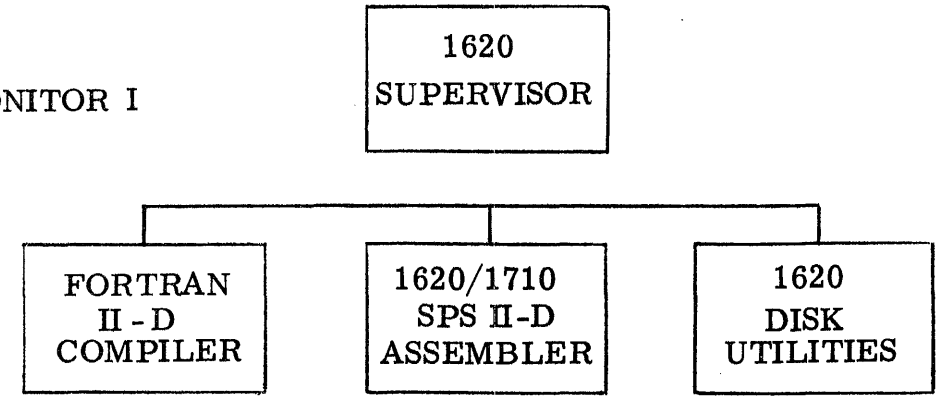
FLIP CHART 4

ADVANTAGES

- Reduce Set-Up Time
- Processor Loading
- Program Organization on the File
- Complete Log of Operations
- FORTRAN II for 20K
- Expanded Error Checking
- Versatile I/O Macros
- Easy Program Segmentation
- Commonly Used Disk Utilities
- Virtual Load / GO - FORTRAN - SPS

FLIP CHART 5

1620 MONITOR I



FLIP CHART 6

DEFINE VARIABLE LENGTH CONSTANT

|DVLC |ADDRESS, LENGTH 1, CON 1, LENGTH 2, CON 2

STRING |DVLC |, 3, 100, 4, 7130, JOE, 612

$\bar{1}00$ $\bar{7}130$ $\bar{6}12$

STRING

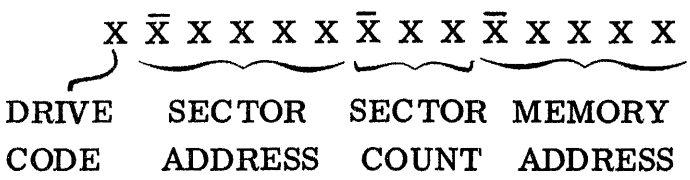
FLIP CHART 7

I/O MACROS

SEEK - POSITION DISK ARM
GET - READ AN INPUT RECORD FROM CARD, PAPER TAPE, TYPEWRITER OR DISK
PUT - OUTPUT A DATA RECORD

FLIP CHART 8

DISK CONTROL FIELD



FLIP CHART 9

IF $W^* n_1 \leq 100$

SECTOR RECORD	SECTOR RECORD	SECTOR RECORD	SECTOR RECORD	SECTOR RECORD

IF $100 < W^* n_1 \leq 200$

SECTOR	SECTOR	SECTOR	SECTOR	SECTOR	SECTOR	SECTOR	SECTOR
RECORD		RECORD		RECORD		RECORD	

FLIP CHART 9A

FIND (I)

FETCH (I) LIST

RECORD (I) LIST

FIND (I)

FETCH (I) A, B, X, Y

RECORD (I) A, B, X, Y

FLIP CHART 10

FORTRAN CONTROL

*FANDK	ff	kk
*PSTSN	n	
*POBJP	n	
*LDISK	nnnnnn	nnnn

FLIP CHART 11

DISK UTILITY PROGRAMS - DUP

WRITE ADDRESSES

ALTER SECTOR

DISK TO OUTPUT

LOAD PROGRAMS

REPLACE PROGRAMS

DISK TO DISK

DELETE PROGRAMS

DEFINE PARAMETERS

DEFINE DISK PACK LABEL

DEFINE FORTRAN LIBRARY SUBPROGRAM NAME

FLIP CHART 12

DISK IDENTIFICATION MAP - DIM

D D D D D D S S S C C C C C E E E E E L

D D D D D D - DISK SECTOR ADDRESS
S S S - SECTOR COUNT
C C C C C - CORE ADDRESS (ALL 9's RELOCATABLE)
(UNITS FLAG RELOADABLE)
E E E E E - ENTRY ADDRESS

L - LAST CHARACTER	FILE PROTECTED	PERMANENTLY ASSIGNED
ǂ	YES	NO
$\bar{ǂ}$	YES	YES
ǂ	NO	NO
$\bar{ǂ}$	NO	YES

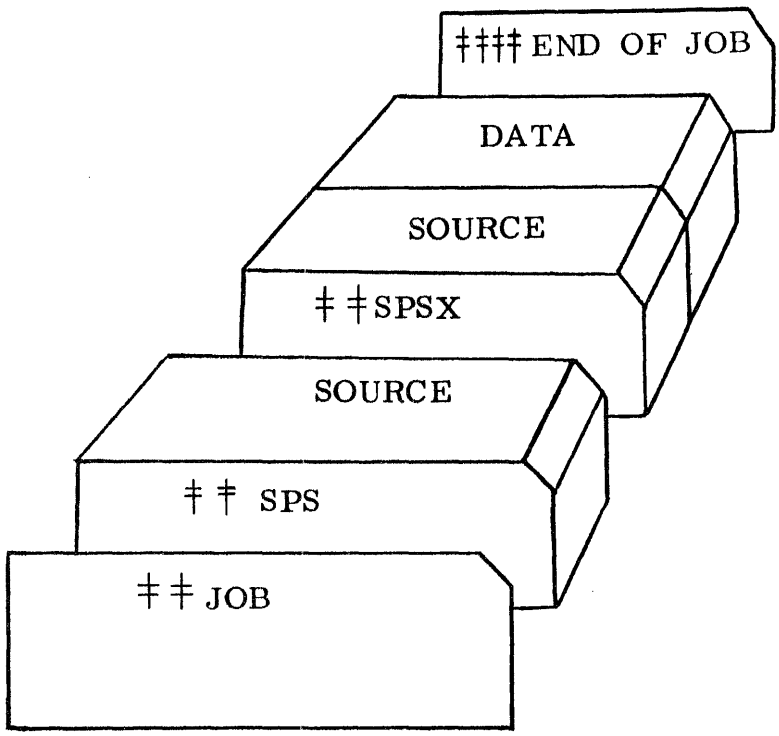
FLIP CHART 13

DEFINE PARAMETERS CONTROL CARDS

SPECIFIES STANDARD FOR:

- SECTOR ADDRESS OF WORK CYLINDERS
- NO. OF WORK CYLINDERS
- NO. OF DISK DRIVES
- NO. OF SECTORS IN DIM TABLE
- NO. OF SECTORS IN EQUIVALENCE TABLE
- NO. OF SECTORS FOR SEQUENTIAL PROGRAM TABLE
- FLOATING POINT MANTISSA LENGTH - SPS
- SPS SUBROUTINE SET
- NOISE DIGIT
- LENGTH OF MANTISSA FOR FORTRAN
- LENGTH OF FIXED POINT FOR FORTRAN
- SOURCE OF INPUT FOR FORTRAN SUBPROGRAMS
- CORE CAPACITY OF OBJECT MACHINE
- CORE CAPACITY OF SOURCE MACHINE
- FORTRAN SUBROUTINE SET

FLIP CHART 14



FLIP CHART 15

SPS MODIFICATION CONTROL CARDS

```
DEFINE  OP        CODE
DELETE  OP        CODE
LIST    OP CODE TABLE

DEFINE SYSTEM SYMBOL TABLE
END  LIB
```

FLIP CHART 16

OP CODE DEFINITION CARDS

New Mnemonic Code	Type of Instruction Code	Assembled Instruction			Type of Instruction
Cols. 12-15	Cols. 16-18				
	XY0	X Y	PPPPP	QQQQQ	ANY
	XY2	3 X	PPPPP	Q 0 7Q Y	DISK
	XY3	3 X	PPPPP	Q 0YQQ	I/O
	XY4	3 4	PPPPP	Q 0XQY	CONTROL
	XY6	4 6	PPPPP	QXYQQ	BI
	XY7	4 7	PPPPP	QXYQQ	BNI
	-XY4	4 X	PPPPP	QQQQY	MASK
	-XY2	8 X	PPPPP	YQQQQ	SIOC
	-XY1	MACRO SUBROUTINE NUMBER XY			

FLIP CHART 17

CONTROL CARDS TO ASSEMBLE

SPS MACRO SUBROUTINES

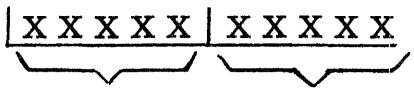
ASSEMBLE RELOCATABLE

LIBR

ID NUMBER	ERROR STOP	OPTIONAL
STORE RELOADABLE	LIST CARD	

FLIP CHART 18

DIM ENTRY FOR SUBROUTINE



LENGTH - AUTOMATICALLY PLACED HERE

SUBROUTINE IDENTIFICATION NO. SPECIFIED AS OPERAND OF DEND

XX	XX	X
SUBROUTINE SET #	SUBROUTINE NO. 18 - 29	NO. OF ENTRY POINTS

SECTION VI

GENERAL COURSE OUTLINE 2

IBM 1620/1311 MONITOR I PROGRAMMING SYSTEM

TEACHING OUTLINE 2

This outline attempts to be complete except where reference is made to supporting documents. Most of the needed detail information is available on the handout sheets, which are intended for use as take-home flipcharts.

R refers to the 1620 Monitor I System Reference Manual, C26-5739-1.

V refers to handout sheets included in Section VIII of this guide.

SECTION VII

DETAILED COURSE OUTLINE 2

1620/1311 Monitor I Preliminary Teaching Outline

July 28, 1963

I. Formalities

A. Name, location, telephone — Produce Visual
B. Position — Produce Visual
C. Class Title — Produce Visual
D. Class Purpose

1. You get latest Monitor info
2. I want your reaction, good or bad
3. I want your recommendations

E. Class Prerequisites

1. 1620 System
2. 1311 Mod 3
3. 1620/1710 SPS
4. 1620 FORTRAN II
5. 1620 Monitor I C26-5739

F. Class Duration - hours
G. Class Style

1. Informal - questions any time
2. Attendance
3. Left to student to separate between external specs, operation, internal organization, and internal operation
4. Question of release to customers
5. Exchange of ideas

H. Message Center
I. Class Sign-up Sheet Containing: — Route around, duplicate and pass out

1. Name
2. Title
3. Location
4. 1620/1311 Accounts

II. Monitor I - Introduction

A. Why? Philosophy of repetitive functions — R-4
B. Functions

1. File maintenance R-4
2. I/O coordination "
3. Job to job continuity "
4. Control over system programs "

C. Precedents

1. File maintenance - many program systems and monitors
2. I/O coordination - many program systems and monitors
3. Job to job continuity - all monitors
4. Control over system programs - all monitors

D. History of Specs

1. Announcement in planned program
2. Preliminary manual
3. Preliminary release to region
4. Preliminary release to field
5.

E. Status of System

1. Early release to PID
2. Scheduled release of final version

F. Major Sections

1. Supervisor
2. Disk Utility Package
3. SPS 2-D
4. FORTRAN 2-D

G. Machine Requirements R-8

1. 20K
2. IA
3. Card or P.T. I/O
4. 1311 Mod 3
5. Hardware Divide for

    a. FORTRAN object programs
    b. SPS object programs using certain subroutines

III. Supervisor - Super

A. Overall Coordination

1. Between jobs (job concept defined later)

a. Determine required function
b. Locate and pull down working program

2. Within job

a. Prevent system shut-down
b. Supply I/O routines as required

B. Maintains Communications Area expand later
i.e. records and updates system status

IV. Disk Utility Programs - DUP
(under control of Super)
Generalized Routines for

A. Specific Functions (see list) V - 1
B. Use by Other Parts of System
C. Note Function as Disk Storage Clerk

V. SPS 2-D

A. Incorporates

1. 1620/1710 SPS
2. 1710 SPS II

B. Additions

1. Disk I/O capabilities
2. I/O Macros
3. System options

C. Will Discuss

1. New capabilities
2. Interesting, obscure portions from 1710 SPS II

VI. FORTRAN 2-D

A. Incorporates FORTRAN II
B. Additions

1. Disk statements
2. System options

C. Will Discuss

1. New capabilities

VII. Disk Storage Organization - 3 tables

A. Equivalence Table R-9

1. Content

a. Name
(1) Concept
(2) Definition
b. Number

2. Function (reduction of alternatives)
3. Format V - 2
4. As delivered R-9

a. Position on disk (always follows DIM table)
cyl.25,sec00-79
b. Capacity (500); max 6240 (999 sectors)
c. Reserved entries

(1) F. lib subr names (50)
(2) SPS 2-D modification prog.
(3) If unused - all nines

5. Type - packed to the "left"
6. Use - determine DIM number when given name

B. Disk Identification Map (DIM) Table R-9

1. Entry content - explain general function

a. Disk sector address
b. Sector count
c. Core address
d. Entry address
e. Status type

2. Function of entry - description of disk area

a. Defines limits of area
b. Defines type and other info for retrieving program
c. Defines status for file maintenance programs

3. Format V-3
Note similarity with disk control field

4. Table Type - Sequentially ordered on DIM no.
Explain interaction with Equivalence table

5. As delivered

a. Location

(1) Cylinder 24
(2) Cannot be changed

b. Capacity - 999 entries; max - 49995 entries (5 cyl) V-4
c. System entries

6. Use - locate storage information given DIM no.

C. Sequential Program Table R-10

1. Content

a. Cylinder numbers
b. DIM numbers
c. Sector counts

2. Function - Image of disk usage, <u>each pack</u>
3. Format V-5
4. Type - pack to the "left", i.e. left-justified

5. As delivered

    a. Location - cyl 99, sec 2-81
    b. Capacity - 2000; max - 2500 (100 sec)

6. Use (principle) to locate available storage

D. Working Storage

1. Definition - DIM entry 0001
2. Content - Scratch area
3. Function - To provide work area for all programs
4. As delivered - cyl 00-23; max 99 (with 2 drives)
5. Use

    a. Supervisor - between programs only
    b. DUP - program loaders and control info as needed
    c. SPS - temp storage of output partially decomposed programs and tables
    d. FORTRAN - temp storage of output partially decomposed programs and tables
    e. Object program - data and subprograms as needed

E. Labels - Each Pack

1. 1400 - 1620 common label

    a. Location - cyl 99, sec 181 - 199
    b. Use - interchange with 1400 series equipment
    c. Defined - DIM 0151 and 0155

2. Monitor

    a. Location - cyl 99, sec 1, pos 00-04
    b. Use - pack ID (system = 00001)

VIII. System Start-up V-6

A. Disk Integrity Lost

B. Core Lost, Disk OK (cold start) R-8
C. Supervisor Lost Control, Core OK - You Think!
(lukewarm start)

IX. Supervisor (in detail)

A. Monitor Control Records R-12

1. Function - op codes to the Super
2. Ident - dbl RM, pos 1 & 2
3. Types V-7
Define job - the records between a job and end-of-job inclusive and the implications of these records to the Supervisor
4. Formats (quiz on examples) R-14
Note V-8,9

a. Disk pack drive override (use analogy to dial-up of tape unit)
b. Disk pack label check (refer to logical drive)
c. SPS, SPSX, XEQS (defer lengthy explanation of SPS subr sets at this point)
d. FOR, FORX, XEQS (also defer discussion of FORTRAN subroutines)
e. XEQ, XEQS - name takes precedence over DIM no., but - DIM no. faster access
f. XEQ, XEQS - defer explanation of program formats
g. Remainder of cards are for comments, RM to stop type.

5. Summarize how Super uses tables to pull down programs R-10
6. Review concept of stacked input R-17

a. Minimum requirements for a job
b. Go over possibilities again
c. Control record specifying execution implies end-of-job
d. Examples V-10,11
R-18

B. Monitor Control Record Analyzer - functions R-18

1. Call for record via type (if entry is from tape)
2. Read Mon Cutl Rec
3. Analyze and set parameters from Cutrl Rec
4. Take action as directed by pseudo op-codes
5. Error detection and institute corrective procedures V12-14 R-19

C. Input-Output Routine - IORT R-21

1. Functions

   a. Perform I-O tasks for

      (1) Super
      (2) Compilers
      (3) DUP
      (4) Object programs

   b. Check I-O operations
   c. Correction of I-O errors where possible
   d. Institute error correction procedures if applicable
   e. Maintain records of disk arms status

2. Use by programmer (general discussion)

   a. Entered via I/O macro codes
   b. System errors produce type messages
   c. Restart procedures given for all error halts
   d. Error counts kept

3. I-O error routine V15, R-24

   a. Procedure
   b. Response codes V16, R-24
   c. Error types V17, R-25, 26
   d. Recall that IORT is used by one and all, hence messages may occur any time

4. Error count retrieval routine V18, R-27

D. System Communication Area

1. Core — V19, R-29
2. Disk sector 19663 — V20, R-29

E. Object Program Loader - functions — R-27

1. Load user prog for cds, tp, or dsk
2. Recog SOF
3. Seq ck if fr cds (col 76-80 (w) flag)
4. Recog SPS patch cds (col 76-80 00000 no flag)
5. Error detection, reporting, correction — V21

F. System Output Format - SOF — V22

X DISK UTILITY PACKAGE

A. Review R-31

1. Trigger - monitor control record, DUP
2. Use of Eq., DIM, and Seq. Prog. tables
3. Note that one of the functions of DUP may be to alter these tables

B. DUP Control Records R-32

1. If typewriter input - ENTER DUP CNTRL REC typed
2. Error in format - mess., halt, return to Super for next monitor control record (hence correction of card is possible)
3. Recall that since DUP uses IORT error messages out of IORT may occur
4. Write address routine R-32

   a. Function

      (1) Write sec addr on dsk pk
      (2) Write read-only flgs - optional
      (3) Clear sectors (to zeros) - optional

   b. Control record format V-1
   c. Operation V-1

5. Alter Sector Routine R-33

   a. Control record format V-2
   b. Operation V-2
   c. Error Procedures R-33, -34
   d. Note alpha mode entry and special character equivalents R-34

6. Disk to Output routine R-34

   a. functions V-3
   b. Recall SOF
   c. control record format V-4
   d. output formats

      (1) cards - 3 sec per 4 cds @ seq. no. and trailer card
      (2) p.t. - same as cds sans seq no
      (3) type V-5

7. Load Programs Routine

   a. functions and options — V-6
   b. review and explain

      (1) working vs permanent storage
      (2) read only sectors
      (3) movable vs. immovable info
      (4) SOF relocatable, SOF absolute, core image

   c. Control record format — R-27 , V-7
   d. Recall messages if read-only flgs
   e. Explain mess given by any DUP prog which loads or moves a program

8. Replace Prog. Routine — R-38

   a. features

      (1) Update prog in permanent dsk stor fr

         (a) cds
         (b) p.t.
         (c) dsk perm stor
         (d) dsk work area

      (2) Assign multiple names (entries to prog.
      (3) Add read only flgs to existing prog.

   b. Control record format — V-8

9. Disk to Disk Routine — R-38

   a. Features and restrictions

      (1) dsk to dsk only
      (2) copy into available area only (including work cyl)
      (3) specifically, not allowed to overlay (or move) another data area
      (4) write read only flgs on copy X work area
      (4) encounter of read only flg during copy stops copying

   b. Control record format — V-9

10. Delete Programs Routine — R-39

   a. Features

XI SPS 2-D

A. New Language Specifications

1. Important inclusions from 1710 SPS II

a. Declarative

(1) DSAC (spec alpha const)	R-59
(2) DVLC (variable length constant)	R-58
(3) DDA (disk address)	R-60
(4) DGM (group mark)	R-61

b. Imperative

(1) Disk I/O mneumonics	R-147, 148
(2) Product area macros	R-67
(3) BNG (branch no GM)	R-143

2. New I/O Mneumonics

a. Declarative R-84

(1) Explain
(2) Types
(3) Formats
(4) Generated coding V1-4

b. Imperative R-83

(1) Function
(2) Types
(3) Formats
(4) Generated coding V1-4

3. Coding form-examples of usage V5, 6, 7
4. Rules of Relocatability R-92

B. Input Sequence - Assembly

1. JOB
2. SPS
3. SPS Control Records R-87

a. General Format

(1) Spelling
(2) Blanks
(3) Remarks

b. Note

(1) Apr 1 sec on dsk per source statement required for intermediate string (5 cyl max)
(2) Error mess if attempt 1 pass mode @ too large prog.
(3) For check purpose object mach assumed same as source if not specified
(4) Hierarchy of parameter designation

(a) Standard (Subr. Set, Mantissa, Noise)
(b) Assemble time
(c) execute time

(5) Order of punched output

(a) Reseq source deck (type corrections not included)
(b) List deck
(c) Symbol Table
(d) SOF for object program (list deck not reloadable)

(6) Review of SOF abs, SOF reloc, CIF
(7) Rules for program replacement R-44

4. Source Statements

a. Language Errors

(1) Message V9, 10
(2) Cause
(3) Procedure - processor disposition of error V11, 12
(4) Procedure - on-line correction V13

b. Organizational Errors and Procedure V14

5. End of Job
6. Assembler Output and Formats V15
7. End of Assembly information R-90

C. Input Sequence - Execution

1. JOB
2. XEQ, XEQS, or SPS X

a. Subroutine

(1) Set ID V16
(2) Macro generated coding V17

b. functions

(1) Obtains data
(2) Calculates return address
(3) Resets error indicator (loc 401)
(4) Provides common work storage
(5) Transfers control to specific subr
(6) Returns result to calling prog
(7) Exits to calling prog

c. Parameter locations V-24

2. Twelve per sets 1, 2, 3 allowable
3. Operands - 2 to 9 allowed
4. Procedure - Programming include

a. Entry point identification V-25
b. Identification no. in DEND V-25
c. If PICK is used, codes for reference to PICK V-26, 27

5. Procedure - Operation

a. Add mneumonic to op-code table (use modification prog
b. Assemble Subr - include

(1) Assemble relocatable
(2) DIM no. (see base no. for SPS subr) V-28
(3) LIBR card

6. Example V-29, 32

XII

FORTRAN II-D

A. New Language Specs

1. CALL EXIT (STOP compilation)
2. Disk I/O

   a. DEFINE DISK (n1, n2)

      (1) n1, no. of items per logical record
      (2) selection of physical rec. size by processor
      (3) n2, max no of logical records

   b. RECORD (i) list

      (1) nature of i
      (2) function of i
      (3) lists for disk I/O

         (a) element type-sector usage
         (b) matrix type
            i. all matrix elements
            ii. f and k both even
            iii. sector usage

      (4) note: no reference to FORMAT

   c. FETCH (i) list

      (1) restrictions on matrix lists
      (2) use of dummy items (nX not allowed)

   d. FIND (i)

      (1) no unnecessary seeks - ref to sys. com. area
      (2) RECORD, FETCH have implied seeks
      (3) System restoration after intervention
          (LOCAL subpr. calls - not discussed yet)

B. Compilation (discussion in deck set-up sequence

1. JOB
2. FOR or FORX (defer discussion of LOCAL specs)
3. Subroutines

   a. Library i.e., relocatable V(1)

      (1) loaded with calling program
      (2) loader receives call via program header record
          (to be discussed.

(3) versions

(a) w/AFP
(b) wout/ AFP
(c) only one version loaded to disk with system

b. Arith and I/O V (2)

(1) always available to object program during exec
(2) versions - (besides AFP and no AFP)

(a) "long" form - 14,000 core
(b) "short" form -
(i) 7500 core
(ii) blocked
(iii) load on call

c. Type identification (standard as del.) V (3)
d. new error conditions V (4)

4. Fortran control records (except LOCAL and DATA)

a. basic format V (5)
b. error procedure V (5)
c. method of replacement w/LDISK
d. method of supplying name if FUNCTION or SUBROUTINE
e. review or explain SOF V (6)

5. Switch use-compilation V (7)
6. Source statements

a. New type I errors V (8)

(1) revert to "edit only"
(2) ignore

(a) POBJP
(b) PSTSN
(c) LDISK
(d) execution (return to super)

b. New type II error V (8)

(1) continues compile
(2) ignore

(a) POBJP
(b) PSTSN
(c) execution
(d) (note load-to-disk)?

7. Phase-to-Phase Operation of Compiler

   a. Intermediate results held on disk (work storage)
   b. All output after successful compilation

      Order
      (1)
      (2)
      (3)

   c. Overlap check (comp. continues)
   d. End of compile type-out
   e. review or explain load-to-disk typeout

C. Execution V(9, 10, 11)

1. JOB-XEQS pair or original FORX
2. LOCAL (load on call) subprograms

   a. concept, philosophy, and usage
   b. restrictions

      (1) cannot call another LOCAL
      (2) cannot call a new lib. subr.

   c. control record format
   d. control record format
   e. recall local card count in XEQS or FORX

3. Switch use during execution V(7)
4. DATA control record
5. Data - action by super re unprocessed data
6. End-of-Job

D. Object Loading

1. location of subpr
2. location of links
3. procedure for LOCAL subpr
4. error procedures V(12, 13)

E. Object Time Core Map V (14)
F. Object Time Disk (work storage) Map V (14)
G. Addition of Lib. Subr. V (15)
H. Writing Subpr. in SPS 2-D V (16-21)
I. Compilation improvement techniques

1. Statement no. comp.
2. Undefined variable check-stop
3. Constant subscript wash-out
4. Subscript calculation wash-out
5. 2, 3-dim subsc cal out of line
6. Subpr argument pick-up, loop coding

SECTION VIII

STUDENT HANDOUTS FOR OUTLINE 2

SUPERVISOR

TABLE OF CONTENTS

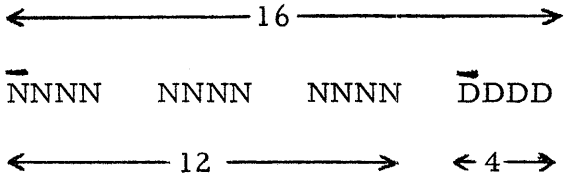
DISK UTILITY PACKAGE

List of Programs

1. Write Sector Addresses
2. Alter Sector (through typewriter)
3. Disk to Output (cards, tape, or type) (data or programs)
4. Load Programs (to disk storage) (from cards, tape, disk work)
5. Replace Programs (on the disk from cards or tape or work or DIM)
6. Disk to Disk (data or programs) (copy)
7. Delete Programs(actually, data or programs) i.e., DIM entry and object
8. Define (system) Parameters
9. Define Disk Pack Label
10. Define FORTRAN Library Subroutine Name

EQUIVALENCE TABLE ENTRY

Equivalence Table Entry - 16 Digits



6 character name 4 digit DIM number

Example:

$\overline{6}245676800000\overline{3}63$

DIM TABLE ENTRY

DIM Table Entry - 20 Digits

$\overline{D}DDDDD\ \overline{S}SS\ \overline{C}CCCC\ \overline{E}EEEEF$

$D\overline{D}DDDD$	Disk sector address of the program or data
$\overline{S}SS$	Sector count
$\overline{C}CCC\overline{C}$	Core address if the program or data is in absolute or core image format, all nines if in relocatable format. If units position is flagged, the Subroutine Supervisor will be used to load the program.

Note the similarity between the first 14 digits and a disk control field.

$\overline{E}EEEE$	Program entry address. This address is relative if the program is relocatable.

F

if	File Protected	Permanently Assigned
‡	yes	no
$\overline{‡}$	yes	yes
≠	no	no
$\overline{≠}$	no	yes

Example: $0\overline{0}55\overline{6}6\overline{0}25\overline{9}999\overline{9}\overline{0}0000\ \overline{≠}$

ie. Sector 05566;
25 sectors in length
relocatable; use Subroutine Supervisor to load;
stored with read-only flags and is also im-movable.

SYSTEM MODULE AS DELIVERED

Program/Table	Cylinders	DIM Numbers
Working Storage	00-23	1
DIM table	24	3
Equivalence table	25	2
FORTRAN Subprogram Loader	80	138, 147, 149, 150 152, 157
FORTRAN Library Subroutines	81	* 10-39
SPS Library Subroutines	82-83	**40-130
FORTRAN I/O and Arithmetic Subroutines	84-97	144-146
FORTRAN Compiler	86-90	136-137, 153, 156
Disk Utility Program	85, 91-92	139-143, 154, 155
SPS Assembler	93-96	8, 9, 131, 132
SPS Subroutine Supervisor	85	133
Supervisor Program with I/O Routine	98	134, 148
Loader Routine	98	135
System Area	99	151
1440, 1401, 1410, Systems Header Label Area	99	166-169
Mutual Disk Pack Identification Label	99	162-165
Sequential Program table	99	4, 5, 6, 7

* only DIM entries 10-25 are in use when the system is delivered

** only DIM entries 40-57, 69-86, 99-117, and 130 are in use when the system is delivered.

SEQUENTIAL PROGRAM TABLE ENTRIES

Sequential Program Table Entries - 4 Digits

Three types:

1.	Cylinder begin point	70 XX	where XX is the cylinder no. (00 - 99)
2.	DIM no.	DDDD	where DDDD is the DIM no. of a program on the cylinder specified.
3.	Available Area	9YYY	where YYY is a sector count of the unused area between adjacent entries.

Example: 7048 0434 0435 7049 0436 9010 0437
7050 0437 7051 0437 7052 0437 ....

Data 434 and 435 on all of cylinder 48. Data No. 436, 10 available sectors, and start of 437 on cylinder 49. Data No. 437 continues onto cylinders 50, 51 and 52.

SYSTEM START-UP

1. Load Monitor to Disk

2. Cold Start (Load Supervisor to core from disk)

Enter and Execute

```
00000        34  00032  00701
00012        36  00032  00702
00024        49  02402
00031        X
00032        Y  19636  113  00102 ≠
```

Where X = 1 for Monitor Control Records from type
3 for Monitor Control Records from tape
5 for Monitor Control Records from card

Y = Drive code of Monitor System

3 Luke Warm Start (Supervisor in core)

```
49 00796      (B MONCAL)
```

LIST OF MONITOR CONTROL RECORDS

JOB	Identifies beginning of job and specifies job parameters.
≠ ≠	Double RM. Marks end-of-job.
SPS	Calls down SPS Processor.
SPSX	Calls down SPS Processor and sets execute switch on.
FOR	Calls down FORTRAN Processor.
FORX	Calls down FORTRAN Processor and sets execute switch on
XEQ	Calls down or loads named SPS object program not requiring SPS subroutines and executes.
XEQS	Calls down or loads named object program and executes.
DUP	Calls down Disk Utility Processor.
TYPE	Calls on typewriter for entry of all further Control Records for this job.
PAUS	Causes System halt. Resume by pressing Start.
other	Treated as comments cards to be typed.

MONITOR CONTROL RECORD FORMATS

All Types:

1, 2 ≠ ≠

All Types except comments, i.e.
JOB, end-of-job (≠ ≠) SPS, SPSX, FOR, FORX, XEQ, XEQS, TYPE, PAUS, DUP:

3-6 Type name, left-justified

Comments type is recognized as any other text in positions 3-6

All types except XEQ, XEQS, ≠ ≠, Comments

	7	source of next input	1=type 3=tape 5=cards

JOB	8-11	Disk Module change numbers	
	12-16	Disk Pack ID drive 0	
	17-21	Disk Pack ID drive 1	
	22-26	Disk Pack ID drive 2	
	27-31	Disk Pack ID drive 3	

SPSX	8-9	SPS Subroutine Set ID number	if non-standard
	10	Noise Digit	if non-standard
	11-12	Mantissa length	if non-standard

FORX	8	FORTRAN Subroutine Set ID number
	9-10	LOCAL control card count

XEQ and XEQS

	7-12	Name of program to be called	
	13-16	DIM entry number (name takes precedence if both are given)	
	17-21	Program loading address if prog is not core image If unspecified 2402 is assumed.	
	22-26	Program execute entry point if not core image. Relative if relocatable.	
	27	source of input	blank = disk 3 = tape 5 = cards

XEQS	28	FORTRAN Subroutines Set ID	
	29-30	LOCAL control card count (FORTRAN)	
	31-32	SPS Subroutine Set Id	if non-standard
	33	SPS Noise digit	if non-standard
	34-35	SPS mantissa length	if non-standard

All types

Characters beyond those specified for any type are at the discretion of the user and may be used for comments.

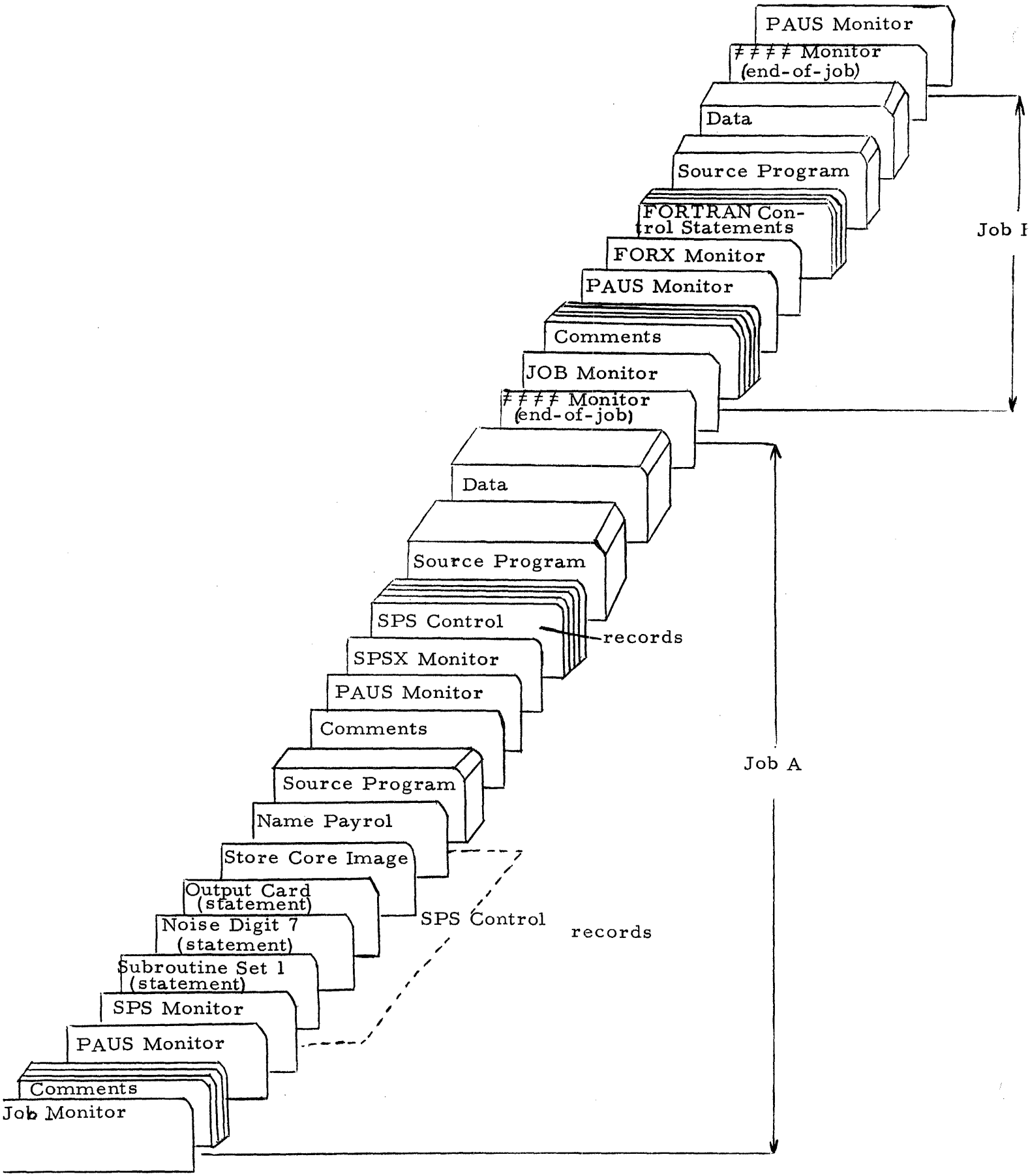
Examples

≠≠JOBb501230000100151001520015 3bbVIBbANALb

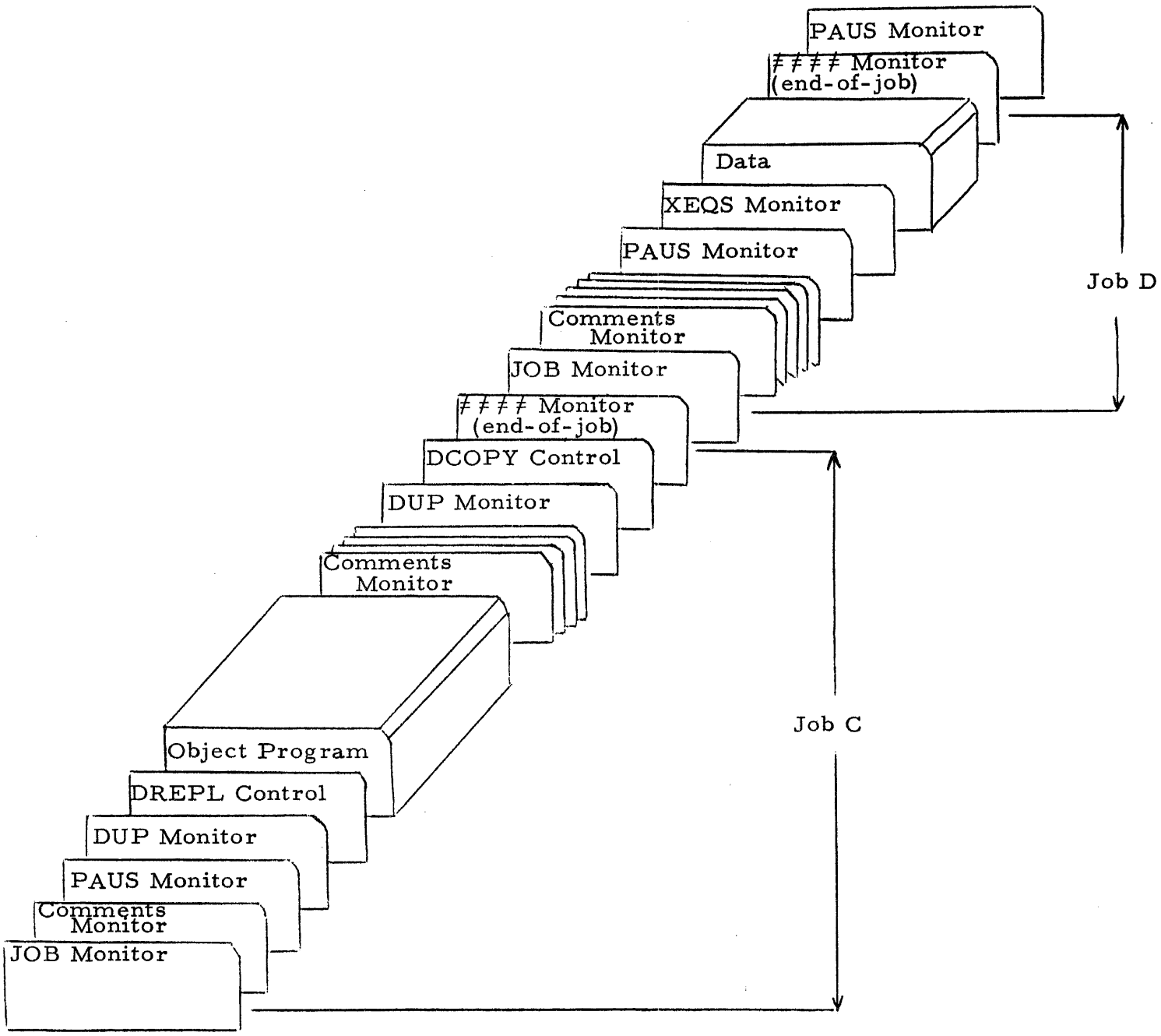
≠≠DUPb5

≠≠bTHISbPROGbWASbNAMEDbSNARK.

≠≠FORX3300


PAUS Monitor
≠≠≠≠ Monitor (end-of-job)
Data
Source Program
FORTRAN Control Statements
FORX Monitor
PAUS Monitor
Comments
JOB Monitor
≠≠≠≠ Monitor (end-of-job)
Job B
Data
Source Program
SPS Control
records
SPSX Monitor
PAUS Monitor
Comments
Source Program
Name Payrol
Store Core Image
Output Card (statement)
Noise Digit 7 (statement)
Subroutine Set 1 (statement)
SPS Monitor
PAUS Monitor
Comments
Job Monitor
Job A
SPS Control
records

STACKED INPUT, DUP and XEQS JOBS



MONITOR CONTROL RECORD ANALYZER ROUTINE ERROR CONDITIONS

Message ERROR IN FIELD AT COLUMN XX. PHASE TERMINATED.

Cause A Monitor control record contains an invalid code in the field starting at column XX.

Action The phase is skipped and the supervisor will pass records from the control record source until it encounters the next monitor control record.

Message EXECUTION IS INHIBITED.

Cause An error has occurred within a job which may prevent susscssful execution of the user's object program.

Action None required. No user object program can be executed until the next JOB monitor control record is encountered.

Message OBJECT DIM ERROR,PHASE TERMINATED.

Cause The supervisor has not been able to find the DIM entry which was called for in a XEQ or XEQS control record.

Action The phase in which the error occurred is terminated and processing continues.

Message OBJECT NAME ERROR,PHASE TERMINATED.

Cause The supervisor has not been able to find an entry in the equivalence table corresponding to the name supplied in cols. 7-12 of XEQ or XEQS control record.

Action The phase is terminated.

Message	CANNOT RESTORE COMMON -- RESET AND START TO RETRY
Cause	Common area does not read into core storage from disk storage correctly
Action	Depress reset and start keys to retry the read operation.
Message	EXECUTION
Cause	Loading and execution of users object program has started
Action	None required
Message	JOB CARD GROUP ONLY
Cause	Control record analyzer routine is expecting a JOB, TYPE, OR PAUS Monitor control record.
Action	Enter JOB, PAUS, or TYPE, Monitor control record from typewriter and depress the release and start keys.
Message	ERROR IN FIELD AT COL. XX. SET SW4 TO IGNORE, OFF TO RE-ENTER CARD.
Cause	An illegal character has been detected in a JOB CONTROL record data field.
Action	To ignore the error, turn program switch 4 on and depress the start key. The message "CONDITION IGNORED" is typed and processing continues. To correct the error turn program switch 4 off, and press start. The monitor control record source will be changed to the typewriter and the operator may then re-enter the control record Note that col. 7 must have a source digit if it is desired that the supervisor reads succeeding records from the original source.
Message	PACK NUMBER ERROR ON MODULE X. SET SW4 ON TO IGNORE OFF TO RECOMPARE.
Cause	Disk pack identification numbers compare "unequal".

Action — To ignore the error, turn program switch 4 on and depress the start key. The message "CONDITION IGNORED" is typed and processing is resumed. To correct the error place the correct disk packs on the disk drives and depress the start key. The disk pack identification n umbers will again be checked by the program.

Message — END OF JOB

Cause — The end of a job has been reached

Action — none required

Message — ENTER MONITOR CONTROL RECORD.

Cause — A "≠≠TYPE" control record has been entered.

Action — The operator must enter a Monitor control record from the typewriter. Control record source is now the typewriter.

Message — SYSTEM DIM ERROR, PHASE TERMINATED.

Cause — Supervisor is unable to find DIM entry for SPS, FORTRAN, OR DUP.

Action — The phase is terminated.

THE I/O ERROR ROUTINE

The I/O error routine has the facility to handle any of the following errors.

Any Check

Typewriter write

Typewriter read

Paper tape read

Card write

Card read

Cylinder overflow

Write error count

Illegal DIM entry

System

Unavailable disk drive

Control record trap

IORT ERROR CORRECTION OPTIONS

Error Correction Code	Option
00	Ignore the error. When this option is used, the I/O routine will finish processing the I/O operation as though the error had not occured.
05	Re-execute the I/O operation. If an error reoccurs during the next execution an error message is again typed, the computer stops, and the operator can exercise the same option or another option.
10	Skip this phase of the job. If at System time (SPS FORTRAN) DUP, or Supervisor), control is returned to the Monitor control record analyzer routine which will read records at the control card source until a Monitor control record is read.
15	Discontinue execution and return control to the Monitor control record analyzer routine and pass records to the next "JOB" Monitor control record.
20	Continue processing by branching to a specified core storage address. When this option is exercised the operator enters 5 digit branch address from the typewriter. No further processing of the I/O request (reposition, execute, etc.) will be done when this option is exercised

IORT ERROR SITUATIONS

	MESSAGE	CAUSE	CORRECTIVE ACTION
1.	ENT ERROR 0607161736738	Any check (19)	A flag will appear over the units position of the indicator causing the error. Exercise operator option
2.	none	Incorrect typewriter write (07)	Invalid character is struck through with a horizontal line, exercise option
3.	TYPE ERR	Typewriter read error (06)	exercise option.
4.	PTR ERR	Paper tape read (06)	backspace tape. Option 05 only.
5.	CDP ERR	Card write (07)	One retry has already been attempted. Exercise option
6.	CDR ERR	Card read (06)	Exercise option
7.	DSK OFL	Disk overflow (38)	Option 05 will act like option 00
8.	DSK ERR XXXXX 0607161736738	Disk read or write error other than legitimate overflow (37 or 38)	XXXXX is object prog return point. Flag will appear as in ENT ERROR above. Exercise option. If 05 and error persist 1. Disk, parity, I/O check switches on STOP 2. Option 05 again 3. Examine console lights
9.	BAD DISK WRITE	Error while writing error count	1. Clear select-lock 2. Reset-start
10.	RESET START MAP ERR XXXXX IIII	User supplied illegal DIM no. in a CALL statement	XXXXX is object prog return. IIII is illegal DIM No. 1. Option 00 2. RS 3. Computer halts 4. type correct DIM 5. RS
11.	IMP ERR	Error routine cannot determine error cause	No intervention allowed
12.	Stop at 00467	Error while IORT reading own subroutines	to retry 1. Clear select-lock 2. Reset-start
13.	MOD ERR XXXXX	Incorrect drive code specified	1. Option 00 only 2. RS 3. Halt 4. Type correct 1-digit drive code 5. RS
14.	TRP ERR	IORT read ≠≠ card when looking for data	Supervisor will process if read in alpha mode
15.	MUST RELOAD	≠≠ card read in numeric mode	1. Reload control record 2. Start

ERROR COUNT RETRIEVAL ROUTINE

1. Enter
```
34 00032 00701
36 00032 X0702
49 00070 0
119754 001 00046≠
```

Where X is the drive code for Monitor I

Error counts for:

06 Read Check
07 Write Check
16 MBR-E
17 MBR-O
36 Address
37 WLR-RBC
38 Cylinder Overflow

will be typed as follows:

$\overline{X}X\ \overline{X}X\ \overline{X}X\ \overline{X}X\ \overline{X}X\ \overline{X}X\ \overline{X}X$

CORE STORAGE COMMUNICATIONS AREA

402-421	Twenty digit DIM entry or fourteen digit disk control field being used by the I/O routine, DUP, or other program.
422-425	Four left-most digits of starting work cylinder address (1 $\overline{0}$00 as delivered)
426	Source of SPS or FORTRAN source program input: 1 = type; 3 = paper tape; 5 = card
427	If flagged, loading resumes at 00000 after a TCD-TRA. If unflagged see 435-39.
428	Source of object program being loaded. 3=tape; 5= cards; 7=disk. If flagged DEND entry triggers program execution. If unflagged then TCD triggers execution.
429	Source of Monitor control input. 1 = type; 3 = tape; 5 = cards. Flagged if subroutines are to be called with SPS or FORTRAN object programs.
430-34	High indicator, i.e., address plus one of highest core location loaded.
435-39	Address where loading is resumed following SPS TRA instruction. Will be $\overline{0}$0000, $\overline{0}$0075, $\overline{0}$0150, or $\overline{0}$0225

DISK COMMUNICATIONS AREA

Disk Sector 1 19663

Positions

00-19	DIM entry used by IORT and Super
20-21	no used
22	0 prog to be loaded to disk is in core image format 1 prog to be loaded to disk is in relocatable format
23	0 = paper tape output; 1 = card output
24-35	6 alpha char: name of prog to be loaded after assembly
36-39	DIM no of prog to be loaded after assembly
40-41	Mantissa length for SPS subr
42-43	SPS Subr ID
44	SPS Noise digit
45-46	FORTRAN Mantissa length
47-48	FORTRAN fixed point length
49	No. of drives available to Monitor
50-64	Super prog-indicators
65-83	Reserved
84-88	First core address of a relocatable object prog
89-93	Computed relocation address of object prog
94-98	Card Sequence no.
99	RM

OBJECT LOADER ERROR SITUATIONS

Message	Cause	Action
XXXXX LD1	Card Sequence Error	XXXXX is the last card in correct sequence. Place cards to follow XXXXX in hopper. Start.
LD2	Card Read Error	Re-read by check-reset and start on the 1622
LD3	Disk Read Error	Start to retry
LD4	Disk Read Error while reading loader	Start to retry

SYSTEM OUTPUT FORMAT - SOF

(All records are card image except for sequence no.) Characters 1-75 contain addresses, data and indicator codes. If cards, 76-80 contain a sequence number or 00000.

I.	1-5	Address of first data character
II.	6	Indicator code
III.	7-8	Length of data
IV.		Data starts in position 7 or 9 since III may not be applicable.

II, III, IV are repeated as many times as can be accommodated on one record. I. does not repeat until the next record unless there is a break in loading sequence.

Indicator Codes

$\neq$	Break in loading sequence. Next 5-digits are new loading address.
$\overline{\neq}$	Used to terminate data on a record prior to position 75
0	Indicate a TRA-TCD compilation in a relocatable program, - next 5 digits are a branch address
$\overline{0}$	Same as 0 except prog is absolute.
1	data is instructions
$\underline{2}$	constants to be relocated
$\overline{2}$	constants not to be relocated
$\underline{3}$	relative addresses to be relocated
$\overline{3}$	relative addresses not to be relocated
$\underline{4}$	Supplies numeric blanks to relocatable prog
$\overline{4}$	Supplies numeric blanks to non-relocatable prog
$\underline{6}$	End of relocatable prog. In SPS next 5-digits indicate prog length.
$\overline{6}$	End of absolute program.

There will be a trailer card with 99999 in 1-5 following the last program card in SPS or FORTRAN output.

DISK UTILITY PACKAGE

ACKNOWLEDGEMENTS

This material on the IBM 1620 Monitor I programming system is the result of two independent education efforts. The first teaching outline, with flipchart sketches, was prepared by W. E. Winn and Mrs. Wendy Gaunt of the Cincinnati and Minneapolis Education Centers. The second teaching outline, with student handouts, was prepared by C. E. Berry, from Western Region Programming Systems. Both outlines represent a considerable amount of careful planning. They will be helpful to anyone teaching a three to five day course on the 1620 Monitor. Suggestions concerning this material should be directed to Education Development, San Jose, California.

TABLE OF CONTENTS

DUP WRITE ADDRESS ROUTINE CONTROL RECORD FORMAT

1	* (asterisk)
2-6	DWRAD
7-12	Disk sector address where writing starts
17	P if read only-flags are to be written
18	Z if data is to be zeroes; S if not to be zeroed
21-26	Address to be written at starting sector
27-32	Last address to be written

DUP WRITE ADDRESS ROUTINE PROCEDURE

1. * DUP TURN ON WRITE ADDRESS KEY, START

2. Type-out: WRITE AND SAVE or WRITE AND ZERO
   SEEK START STOP
   $X\bar{X}XXXX$ $X\bar{X}XXXX$ $X\bar{X}XXXX$

3. User verifies intentions. Start

   If unable to find start addr (7-12)
   ER SK $XX\bar{X}XXX$ where $XX\bar{X}XXX$ is the seek address.
   Also, 20 sector addresses from the track will be typed. STOP.
   START returns control to SUPERVISOR.

4. *DUP TURN OFF WRITE ADDRESS KEY, START.

DUP ALTER SECTOR ROUTINE CONTROL RECORD FORMAT

1	*
2-6	DALTR

DUP ALTER SECTOR ROUTINE PROCEDURE

1. SECTOR typed
2. Type 6 digit sector to be altered, RS
   (If more than 6 digits are typed
   SECTOR NUMBER ILLEGAL, START TO REENTER DALTR)
3. Sector type out (margins 70 or more apart)

1st. HALF xxxxxxxxxx xxxxxxxxxx xxxxxxxxxx xxxxxxxxxx xxxxxxxxxx ORIGINAL
2nd HALF xxxxxxxxxx xxxxxxxxxx xxxxxxxxxx xxxxxxxxxx xxxxxxxxxx ORIGINAL

4. Each 10 digits is given a section number, 01-10.
   SECTION is typed out
5. Type 2-digit section number of first section to be changed, RS
   (If illegal: SECTION NUMBER ILLEGAL, START TO REENTER DALTR)
6. xxxxxxxxxx TYPE CHANGE. Selected section is typed out.
7. Type change in same format. X may be typed for unchanged digits. If more than one section is to be changed a blank must separate them.
   (If type-in proceeds beyond section 10: TYPE-IN EXCEEDS SECTOR LENGTH typed. Start allows operator to begin again)
8. Original and Corrected Sector are typed out
9. SECTION typed. Operator may change other sections.
10. If operator types ≠ the updated sector is written on disk. DISK SECTOR $\overline{\text{D}}\overline{\text{D}}$DDDDD CORRECTED typed where $\overline{\text{D}}\overline{\text{D}}$DDDDD is the sector being altered.
11. SECTOR typed. User may change another sector.
12. If user types ≠, RS, control returns to Super.

Note that data entry is in alpha mode. Flagged RM and flagged GM are entered through W and G keys respectively; J thru R for flagged digits; - for flagged zero.

DUP DISK-TO-OUTPUT ROUTINE FUNCTIONS

Output to cards, tape, or typewriter

1. Prog or data by name
2. Prog or data by DIM number
3. Data between sector limits
4. DIM table
5. Equivalence table
6. Availability List
7. Sequential Program table

DUP DISK-TO-OUTPUT CONTROL RECORD FORMAT

1	*
2-6	DDUMP
7-12	Name
13-16	DIM number (Name takes precedence if both are given)
17	Output device - C, P, T
18	Output Identification

I = DIM table
E = Eq. table
A = Availability entries from module specified
S = Sequential Program table
M = Program: see 7-12 or 13-16
L = Sector Limits: see 21-32

19	Module number (if A or S in 18)
21-26	Beginning sector of output (if L in 18)
27-32	Last sector of output (if L in 18)

DUP DISK-TO-OUTPUT TYPEWRITER FORMATS

1. Availability Lists

(NOT CARD OUTPUT)

AAAAA
BBBBBB CCCCCC
BBBBBB CCCCCC
etc.

AAAAA = Disk pack ID
BBBBBB = start disk addr of unused area
CCCCCC = end disk addr of unused area

2. Equivalence Table

NNNNNN IIII (5 per line)

where NNNNNN is the name
IIII is the DIM no.

3. All others: 100 digit sectors, 75 on the 1st line, 25 on the second.

DUP LOAD PROGRAMS ROUTINE FEATURES

The routine provides the following program loading options:

1. A Name may be assigned to the program in the equivalence table.

2. A DIM entry may be created by assigning a DIM number to the program.

3. The disk storage location can be permanently assigned (fixed).

4. An entry address (execution address) can be assigned in the DIM entry to the program.

5. Read-only flags can be written in the sector addresses.

6. The disk storage location can be selected by cylinder (S) for the program without permanent assignment. Thus, several associated programs can be assigned the same cylinder or group of cylinders by the user without acutally specifying sector addresses.

7. Programs in either core-image or system output formats can be loaded; and programs in system output format can be converted to core image while loading.

8. Info between work Sector limits

DUP CONTROL RECORD FORMAT FOR LOAD PROGRAMS ROUTINE

1	*
2-6	DLOAD
7-12	Name of Program
17-20	DIM no. of Program
21-26	Start disk address of prog in work cylinders to be stored
27-32	End disk address of program in work cylinders
33-38	Disk address where program is to be loaded to. If included, the program will be permanently assigned to this address.
39-43	Core address of non-relocatable program. This goes into CCCCC of DIM entry.
44-48	Entry address. This is placed in EEEEE portion of DIM entry and must be given for program in core image format.
49	Input device C, P, D
50	I = prog is in core image format (CIF) S = prog is in SOF M = prog is in SOF to be converted to CIF
51	P = write read-only flags
52-54	Start cylinder where prog can be loaded (format: XYY, X = 0, 1, 2, 3; YY = 00-99) where X is module, YY is cylinder no.
55-57	End cylinder where prog may be loaded (same format as preceding)

DUP CONTROL RECORD FORMAT FOR REPLACE PROGRAM ROUTINE

1	*
2-6	DREPL
7-12	Name of program (see note)
13-16	DIM no. if replacing program comes from another disk area
17-20	DIM no. of replaced program (see note)
21-26	Start disk address of prog in wk cyl
27-32	End " " " " " " "
39-43	Core address of non-relocatable info, goes into $\overline{C}$CCCC of DIM entry.
44-48	Entry address. Goes into $\overline{E}$EEEE portion of DIM entry and must be given for programs in core image format.
49	Input device. C, P, D
50	I = info is in core image format (CIF) S = prog is in SOF M = prog in SOF to be stored in CIF
51	P = write read-only flags

NOTE: If contents of 13-16 equal 17-20 and name in 7-12 does not correspond to DIM no. in 17-20, then DUP assumes that 7-12 is a synonym name and is so entered into the equivalence table.

DUP DISK-TO-DISK ROUTINE CONTROL RECORD FORMAT

1	*
2-6	DCOPY
7-12	Name
13-16	DIM no. (name takes precedence)
21-26	Start sector of program to be copied
27-32	End sector of program to be copied.
33-38	Start sector address of new copy
51	P if read only flags are to be written.

DUP DELETE PROGRAM ROUTINE

1	*
2-6	DELET
7-12	Name
13-16	DIM no.

DUP DEFINE PARAMETERS ROUTINE FEATURES

1. Alter assignment of work cylinders location and length
2. Alter assignment of DIM table length
3. Alter assignment of Eq. table length
4. Alter assignment of Seq. Prog. table length
5. Indicate no. of drives on the system
6. Define standard for SPS
   a. floating point mantissa length
   b. Subroutine Set ID no.
   c. Noise digit
7. Define standard for FORTRAN
   a. Mantissa length
   b. Fixed point length
   c. Subprogram source, alternate
   d. Subroutine set ID
8. Indicate machine size for
   a. Source program compilation
   b. Object program execution

DUP DEFINE PARAMETERS ROUTINE CONTROL RECORD FORMAT

1	*
2-6	DFINE
7-12	Beginning disk sector address of work cylinders (000000)
14-16	No. of work cylinder, max 99 (24)
18	No. of drives on system (1)
20-22	Sectors in DIM table, 999 max (200)
24-26	Sectors in Equi valence table, 999 max (80)
28-30	Sectors in seq Prog Table, 100 max (80)
37-38	Standard SPS Mantissa length (08)
40-41	Standard SPS Subr. Set ID (02)
43	Stardard SPS Noise digit (0)
45-46	Standard FORTRAN Mantissa length
48-49	Standard FORTRAN Fixed length (04)
51	Source of FORTRAN Subprogram (other than disk) (5) = cds, 3=pt
53	Core capacity of object machine $\overline{1}$ == 20K; $\overline{2}$ = 40K; $\overline{3}$ = 60K; ($\overline{1}$)
55	Core capacity of source machine, ($\overline{1}$)
57	FORTRAN Arith and I/O Subr. ID, (1)

DEFINE DISK PACK LABEL ROUTINE CONTROL RECORD FORMAT

1	*
2-6	DLABL
7-11	ID number to be assigned
12	Drive number of disk to be labeled

DEFINE FORTRAN LIBRARY SUBROUTINE NAME

1	*
2-6	DFLIB
7-12	Name (left-justified)
14-15	DIM number of subroutine being named.

DISK UTILITY PACKAGE ERROR CONDITIONS

Error Messages	Cause
DUP * ERROR 01	Field missing from control card.
DUP * ERROR 02	"TO" DIM entry number specified in DREPL control card is not in use in DIM table.
DUP * ERROR 03	"TO" DIM entry number, specified in a DREPL control card refers to an immovable program.
DUP * ERROR 04	"FROM" DIM entry number specified in a DDUMP, DREPL, or DCOPY control card is not in use in the DIM table.
DUP * ERROR 05	Work cylinders illegally specified for program storage by DLOAD or DREPL control card entry.
DUP * ERROR 06	DIM entry number specified in a DDUMP, DLOAD, DREPL, DCOPY, or DLETE control card is out of range of DIM table entry capacity.
DUP * ERROR 07	"FROM" DIM entry number in a DREPL control card refers to an immovable program.

Error Messages	Cause
DUP * ERROR 08	Insufficient available storage space at location specified by a DLOAD, DREPL, DCOPY, DFINE control card.
DUP * ERROR 09	DIM table is full.
DUP * ERROR 10	Field in DFLIB, or DLABL control card contains invalid data.
DUP * ERROR 11	Number of modules specified in DFINE control card is greater than 4.
DUP * ERROR 12	Beginning disk sector address of work cylinder in DFINE control card is not first address in cylinder.
DUP * ERROR 13	Insufficient available storage for specified work cylinder (DFINE control card).
DUP * ERROR 14	Number of sectors specified by DFINE control card for sequential program list exceeds 140 sectors.
DUP * ERROR 15	Sector address is non-numeric for DCOPY control record.

Error Messages	Cause
DUP * ERROR 16	Storage location specified by DCOPY control card would cause program storage to overlap work cylinders if allowed.
DUP * ERROR 17	Starting sector address is greater than ending address for DWRAD or DCOPY control card.
DUP * ERROR 18	Sequential Program table is defined as less than required by the present contents of that table (DFINE control card).
DUP * ERROR 19	CORE address of a program to be placed in disk in core image format is less than 2302.
DUP * ERROR 20	Name specified by DDUMP, DCOPY, or DELET control card is not used in Equivalence table.
DUP * ERROR 21	Immovable program prevents DCOPY routine from being executed.
DUP * ERROR 22	DIM number specified by DELET control card is not in use in Equivalence table.

Error Messages	Cause
DUP * ERROR 51 AAAAAA	Name specified is a DLOAD, DREPL, or DFLIB control card or a FORTRAN or SPS control card has been rejected because a duplicate name exists in the equivalence table. AAAAAA is the rejected name.
DUP * ERROR 52 DIM NO IN USE, TYPE REC.MK. TO FIND UNUSED DIM NO.	"TO" DIM entry number specified in DLOAD control card is in use in DIM table.
DUP * ERROR 53 AAAAAA	Name specified in a DLOAD or DREPL control card or a FORTRAN or SPS control card has been rejected because the Equivalence table (with the exception of the first 50 entries) is full. AAAAAA is the rejected name.
DUP * ERROR 54 AAAAAA	Name specified in a DLOAD, DREPL, or DFLIB control card or a FORTRAN or SPS control card has been rejected because the first 50 entries of the Equivalence table are full. AAAAAA is the rejected name.

Error Messages	Cause
DUP * ERROR 55 CARD SEQUENCE NNNNN	Sequence error has been found while reading a program to be loaded to disk storage. NNNNN is the sequence number of the card that is out of sequence. Only cards with an eleven punch over the leftmost position of the sequence number (column 76) are sequence checked; therefore, patch cards are excluded from the check.
DUP * ERROR 56	DIM number supplied in FORTRAN or SPS control card is in use in DIM table and Name specified in the same card has a different DIM number in the Equivalence table.
DUP * ERROR 57	DIM number supplied in FORTRAN or SPS control card is in use in DIM table and Name specified in the same card has no matching name in the Equivalence table.
DUP * ERROR 58	"TO" DIM entry number specified in FORTRAN or SPS control card refers to an immovable program storage area.

Error Messages	Cause
DUP * ERROR 59	DIM entry number specified in a FORTRAN or SPS control card is out of range of DIM table entry capacity.
DUP * ERROR 60	Insufficient available storage space at location specified by FORTRAN or SPS control card.
DUP * ERROR 61	DIM table full.

DISK UTILITY PACKAGE ERROR CONDITION OPERATOR ACTION

Messages 1-22. After the message is typed, the computer halts. The operator may then enter a corrected control record. Depressing the start key returns control to the Monitor Control Record Analyzer routine to read the next monitor control record.

Message 51. No operator action required. The routine continues and loads the program without placing the name in the Equivalence table.

Message 52. After the message is typed, the computer halts. If the operator enters a record mark from the typewriter and depresses the R-S key, the disk utility program will assign a DIM entry and load the program. If the R-S key is depressed without entering a record mark, control is returned to the Monitor Control Record Analyzer routine to read another monitor control record.

Messages 53, 54. No operator action required. The routine continues and loads the program without placing a name in the Equivalence table.

Message 55. After the message is typed the computer halts. To restart:

1. Remove the cards from the hopper.
2. Depress the Non-process Runout key.
3. Remove the last two cards from the stacker.
4. Rearrange cards from steps 1 and 3 in corrected sequence and place them in the hopper.
5. Depress the reader Start key. Note that paper tape contains no sequence number, therefore, it can never generate this type of error.

Message 56. No operator action required. The routine continues and loads the program, generating a DIM entry, without placing the name in the Equivalence table.

Message 57. No operator action required. The routine continues and loads the program, generating a DIM entry, and places the name in the Equivalence table.

Messages 58,59. No operator action required. The routine continues, generating a DIM entry, and loading the program to disk storage.

Messages 60,61. After the message types, the computer halts without loading the program providing no other output is requested. (If a FORTRAN or SPS control record indicates that the compiled or assembled object program is to be punched, the program is outputed without halting the computer.)

To correct the error, a DLOAD, DREPL, or DDUMP monitor control record can then be entered in the stacked input to load the program to a different location from the work cylinders. Depressing the start key returns control to the Monitor Control record analyzer routine to read the next Monitor Control record.

Routine	Error Message Number																																
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	51	52	53	54	55	56	57	58	59	60	61
Write Address (DWRAD)	X																X																
Alter Sector (DALTR)																																	
Disk-to-Output (DDUMP)	X			X		X														X													
Load Programs (DLOAD)	X				X	X		X	X														X	X	X	X	X						
Replace Programs (DREPL)	X	X	X	X	X	X	X	X																	X	X	X						
Disk-to-Disk (DCOPY)	X			X		X		X							X	X	X			X	X												
Delete Programs (DELET)	X					X														X		X											
Define Parameters (DFINE)	X							X			X	X	X	X				X															
Define Disk Pack Label (DLABL)	X									X																							
Define FORTRAN Library Subroutine Name (DFLIB)	X									X													X			X							
FORTRAN or SPS Output																							X		X	X		X	X	X	X	X	X

DISK UTILITY PROGRAM ERROR CONDITION TABLE

SYMBOLIC PROGRAMMING SYSTEMS 2-D

Table of Contents

I/O Mneumonics Generated Coding

1. GET, PUT, SEEK

```
TFM IORTR, *+23
B   ENTRY, DEF, 7
```

where IORTR is core 00565
ENTRY is one of the following:

IORBC	00520	Write disk, RBC
IOPT	00532	Write to output device
IOSK	00554	Seek a disk record
IOGT	00566	Read from input device

DEF is the label of an I/O declarative constant

2. CALL LINK or CALL LOAD

```
TFM  IORTR, *+19
B7   IOCAL
DC   3,320
DC   4,IIII
[DSA  LLLLL]   only if a relocation address operand is given
DSC  1,@
```

where IIII is the DIM of the called program
LLLLL is the core load address of the called program
IOCAL is 00716

3. CALL EXIT

```
B7  MONCAL                    (00796)
```

4. I/O Declarative Constants (except DD and DDW)

```
    DSA  IOAREA
DEF DS   ,*-4
    DC   2,MM
    DSC  1,@
```

where IOAREA is the label of the associated IO area
MM is one of the following:

DTN	$\bar{0}0$
DTA	$\bar{0}6$
DCN	$\bar{0}4$
DCA	$\bar{1}0$
DPTN	$\bar{0}2$
DPTA	$\bar{0}8$

5. DDW and DD

```
DEF   DSC  1, M0
      DSC  1, M1
      DSA  DCTRL
     [DSA  RELOC]   only if third operand is specified
      DC   1,@
```

where DCTRL is the label of the disk control field and M_0 and M_1 are as follows:

M_0 (code)	Option
0	Add the starting address of the work cylinders from the communications area (LOC 422-425) to the sector address in the disk control field. (Used with constant types 1 and 2 only.)
1	Same as option zero except, also update the "high" indicator in the communication area for disk read operation only. This indicator is merely a field which contains the core storage address of the highest position to be loaded plus one.
2	Use the sector address in the disk control field for the disk operation (SEEK, READ, or WRITE.)
3	Use the sector address in the disk control field for the disk operation. Also, update the "High" indicator in the communication area for read operation only.
$\bar{n}$	A flag over the code ($\bar{n}$, n=0, 1, 2 or 3) causes the read/write heads to be repositioned to an assigned cylinder (specified in the communications area) after any disk I/O operation, except seek.

M_1 (code)	Option
0	Disk read or write in sector mode with WLRC. The user must place a group mark (‡) in the core storage loaction following the last character position of the last sector of the record.
2	Disk read or write in sector mode without WLRC.
4	Disk read or write in track mode with WLRC. The user must place a group mark (‡) in the core storage location following the last character position of the last sector of the record.
6	Disk read or write in track mode without WLRC.
$\bar{n}$	A flag over code ($\bar{n}$ n=0, 2, 4, or 6) causes the I/O Routine to branch to a given address after a disk read operation. The given address will be the "execution address" if an extended disk control field is used. Otherwise, it will be the "core address" of the disk control field. If code n is unflagged, the I/O Routine will branch to the first instruction following the disk operation calling linkage in the object program

IBM

1620/1710 Symbolic Programming System
Coding Sheet

Program: _____ Date: _____ Page No. 1 2 of _____

Routine: _____ Programmer: _____

Line	Label	Operation	Operands & Remarks
3 5	6 11	12 15	16 20 25 30 35 40 45 50 55 60 65 70 75
0,1,0			
0,2,0	*	DEFINE	SPECIAL ALPHA CØNSTANT
0,3,0	LDSAC	DSAC	24,LDSAC = LØW-ØRDER DIGIT.,,, CØMMENTS
0,4,0	*	DEFINE	VARIABLE LENGTH CØNSTANT
0,5,0	LDVLC	DVLC	,L1,C1,L2,C2,L3,C3
0,6,0	*	NØ REMA	RKS IN DVLC, TØTAL MAX LENGTH = 50, LDVLC LABELS C1-LØ DIGIT
0,7,0			
0,8,0	*	DEFINE	DISK ADDRESS
0,9,0	LDDA	DDA	,DC,SECADR,SECØNT,CØRADR
1,0,0	*	DC = DR	IVE CØDE, SECADR = SECTØR ADDRESS, SECØNT = SECTØR CØUNT,
1,1,0	*	CØRADR	= CØRE ADDRESS, LDDA REFERENCES DC
1,2,0	*	NØTE TH	AT IØRT REQUIRES A TERMINATING RM
1,3,0			
1,4,0	*	DEFINE	GRØUP MARK
1,5,0		DGM	LØC
1,6,0			
1,7,0	*	PRØDUC	T AREA MNEMØNICS - SAVE
1,8,0		SAVE	LØCA,PADR
1,9,0	*	LØCA I	S HØLD AREA, PADR IS PRØDUCT AREA ADDRESS IF PADR IS
2,0,0	*	ABSENT	80 IS ASSUMED. GENERATED CØDING FØLLØWS
	*	TDM	100,0
	*	DC	1,@,*
	*	TR	LØCA,PADR
	*	TDM	100,9

IBM

1620/1710 Symbolic Programming System
Coding Sheet

Program: _____ Date: _____ Page No. 1 2 of _____

Routine: _____ Programmer: _____

Line	Label	Operation	Operands & Remarks
0,1,0			
0,2,0	* PRØDUCT AREA MNEMØNIC - RSTR		
0,3,0		RSTR	PADR,LØCA
0,4,0	* PADR,LØCA AS BEFØRE, GENERATED CØDING FØLLØWS		
0,5,0	*	TR	PADR,LØCA
0,6,0	*	TDM	100,0
0,7,0			
0,8,0	* BRANCH		NØ GRØUP MARK
0,9,0		BNG	BRADR,GMLØC
1,0,0			
1,1,0	* NØN-DISK I/Ø DECLARATIVE CØNSTANTS - EXAMPLES		
1,2,0	DEF	DCN	,IØAREA,REMARKS
1,3,0	* IØAREA		IS THE LEFT PØSITIØN ØF THE I/Ø AREA IN QUESTIØN ,E.G.,
1,4,0	IØAREA	DSS	80
1,5,0	* DISK		I/Ø DECLARATIVE CØNSTANTS - EXAMPLE
1,6,0	DEF	DDW	,LDDA,RELØC,R,A,REMARKS
1,7,0	* RELØC		IS RELØCATIØN ADDRESS IF REQUIRED; IF PRESENT, R CAUSES
1,8,0	* ACCESS		ARM REPØSITIØNING; IF PRESENT,A SPECIFIES THE SECTØR
1,9,0	* ADDRESS		ØF THE DDA TØ BE ABSØLUTE, ØTHERWISE IT IS ASSUMED TØ
2,0,0	* BE		RELATIVE TØ THE FIRST WØRKING STØRAGE SECTØR
	* LDDA		REFERENCES APPRØPRIATE DDA

IBM

1620/1710 Symbolic Programming System
Coding Sheet

Program: _____ Date: _____ Page No. 1 2 of _____

Routine: _____ Programmer: _____

Line	Label	Operation	Operands & Remarks
010	* 1/Ø	MACRØ	MNEMØNICS EXAMPLES
020		GET	DEF
030			
040		PUT	DEF
050			
060		PUT	DEF,RBC
070			
080		SEEK	DEF
090			
100		CALL	LINK,416
110		CALL	LINK,DIMNØ ,,DIMNØ IS PRØGRAM NAME. IT MUST
120		CALL	LINK,DIMNØ,2402 , BE IN EQUIVALENCE TABLE.
130		CALL	LINK,DIMNØ,RELØC
140		CALL	LINK,NAME,RELØC
150		CALL	LØAD,427
160		CALL	LØAD,DIMNØ
170		CALL	LØAD,428,2402
180		CALL	LØAD,DIMNØ,RELØC
190		CALL	LØAD,NAME
200		CALL	LØAD,NAME,15000
		CALL	EXIT

List of SPS Control Record Types

	Standard
TWO PASS MODE	ONE PASS
OBJECT CORE n where n = 2, 4, 6	2
SUBROUTINE SET nn where nn = 00, 01, 02, 03	02
MANTISSA LENGTH n or nn where nn = 02-45	08
NOISE DIGIT n where n = 0-9	0
ERROR STOP	No error stop
ASSEMBLE RELOCATABLE	Absolute
BEGIN CARD INPUT	Card input
BEGIN PAPER TAPE INPUT	
BEGIN TYPEWRITER INPUT	
TYPE SYMBOL TABLE	No sym tab type
PUNCH SYMBOL TABLE	No pch sym tab
LIST TYPEWRITER	No list type
LIST CARD	No list card
OUTPUT CARD	No output card
OUTPUT PAPER TAPE	No output pt
STORE CORE IMAGE } either neither but not both	
STORE RELOADABLE } either neither but not both	
SYSTEM SYMBOL TABLE	No sys sym tab
NO SUBROUTINES	With subroutines
ID NUMBER dddd where dddd is DIM no. to be assigned or replaced	
NAME aaaaaa where aaaaaa is name to be assigned or replaced	
LIBR	
PUNCH RESEQUENCED SOURCE DECK	No reseq deck

SPS Compiler Description of Error Codes

ER1 The capacity of the machine on which the object program is to be executed has been exceeded. The processor does not take subroutines into account when determining this error.

ER2 Invalid label or record mark is in a label field.

ER3 Invalid OP code or record mark is in an OP code field.

ER4 A label is defined more than once.

ER5
a. a symbolic address contains more than six characters.
b. an actual address contains more than five digits.
c. an undefined symbolic address is used in an operand.
d. a HEAD character ($) is improperly specified.

ER6 A DSA statement has more than ten operands.

ER7 A DSB statement has the second operand missing.

ER8
a. a DC, DSC, or DAC has a specified length greater than 50.
b. a DVLC has a length greater than 50.
c. a DMES has a length greater than 100.
d. a DNB has a length greater than 99.

ER9 A DS, DSC, DAC, DVLC, or DMES statement has no constant specified.

ER10
a. a DC or DSC statement has a specified length which is less than the number of digits in the constant itself.
b. a DAC statement has a specified length which is less than or greater than the number of digits in the constant itself.

ER11 An invalid character is used as a HEAD character in a HEAD statement.

ER12 A HEAD operand contains more than one character.

ER13 A DMES statement contains an invalid starting mode character.

ER14 a. a DMES statement contains a control character which is incorrectly specified.

b. a DMES statement has an invalid format; i.e., stray parenthesis, etc.

ER15 A DMES statement contains an alpha character in a numerical field.

ER16 A DMES statement contains an invalid mode change.

ER17 a. a relocatable assembly contains a relocation error (see rules of relocatability).

b. A relocatable assembly contains a DORG with an absolute operand.

ER18 A symbolic name used in a CALL LINK or CAIL LOAD statement is not in the Equivalence Table.

ER19 The storage area allotted for the symbol table has been exceeded.

ER20 Intermediate output has exceeded disk storage work area (program requires two passes).

ER21 Object output has exceeded disk storage work area.

ER22 Improper "select" operand is in a CALL statement; i.e., neither LINK, LOAD or EXIT are specified.

SPS Compiler

Disposition of Errors When No ERROR STOP Statement Is Used

NOTE: Processing continues in all cases listed below:

ER1	No disposition.
ER2	The label is ignored.
ER3	A NOP is assembled.
ER4	The second definition of the label is ignored; the first definition of the label is used in the assembly.
ER5	The operand is assembled as an absolute 00000.
ER6	The first ten operands are assembled; any remaining operands are ignored.
ER7	The number of elements is set to 1.
ER8	a - Length is set to 50. b - Length is set to 50. c - Length is set to 100 d - Length is set to 99.
ER9	A field of zeros is generated, equal to the size of the length operand for the DC, DSC, DAC, or DVLC constant. In the case of a DMES, an end of message (≠≠) is assembled and the address counter is increased by 100.
ER10	a - The length of the constant is used as the length operand. b - The specified length is used and the programmer-assigned address, if present, is ignored.
ER11	The HEAD character is set to blank.
ER12	The first character of the operand is used as the HEAD character.

ER13 The starting mode is assembled as the alphabetic mode.

ER14 An end of message (≠≠) is inserted into the constant.

ER15 An end of message (≠≠) is inserted into the constant.

ER16 A $\bar{0}$ is placed in the next available location following the mode change.

ER17 a - The operand is assembled as an absolute 00000.

b - The DORG is ignored.

ER18 A DIM number of 0000 is assembled

ER19 Processing continues but no more labels are stored. After completion of the intermediate phase, processing stops, the following message is typed, and control returns to the Supervisor program.

DISK AREA TOO SMALL. ASSEMBLY DELETED

ER20 Processing continues, but no more intermediate data is sent to disk storage. After completion of the intermediate phase, processing stops, the following message is typed, and control returns to the Supervisor program.

DISK AREA TOO SMALL. ASSEMBLY DELETED

ER21 Processing stops immediately and control is returned to the Supervisor program.

ER22 The statement is ignored.

Use of On-line Error Correction in SPS

1. Include SPS Control Record ERROR STOP in deck.

2. Error Message is typed out.

3. RE-ENTER STATEMENT or RE-ENTER OPERANDS typed out.

4. Erroneous statement typed out.

5. Operator types in correction following format of 4.
   Type in according to instructions in 3. (Typewriter will position to correct starting point.)

In addition in the two-pass mode:

During the second pass the processor may type ERXX
RE-ENTER STMT

The operator should re-enter the statement exactly as it was corrected during pass I.

SPS Error Messages After Assembly

1. EXCEEDED SPECIFIED CAPACITY BY XXXXX
2. SUBROUTINES OTHER THAN DIV USED
3. NO DIM ENTRY FOR SUBROUTINE
4. MORE THAN 5 CYLINDERS OF RELOADABLE OUTPUT
   SSW4 ON TO DUMP OUTPUT, OFF TO CONTINUE, NO OUTPUT

Corrective action

1. Not considered a catastrophe error since the user may specify a different subroutine set at execution.

2. Same as 1.

3. Not catastrophic. User may load subroutine in question to disk prior to execution.

4. Self explanatory. Note that a DIM entry cannot specify more than 999 sectors, hence the program will not be loaded to disk in any case.

SPS Output - Listing Formats

1. Symbol Table
   A. Typewriter
      LLLLLLbbbAAAAA?
      The ? is blank if the quantity is positive
      a - if the quantity is negative
      a R if the quantity is relocatable.
   B. Cards
      LLLLLLbAAAAA?bbb Five fields per card.

2. Program List Deck
   Cards

1-5	Page and Line no.
6	blank
7-12	Label
13	blank
14-17	op code
18	blank
19-78	operands

If operands do not extend beyond col 59,the following appears on the same card, otherwise on the next card.

61-65	Address of assembled instruction or constant
66	blank

Imperative statements only

67-68	Machine language op code
69	blank
70-74	P-operand in machine language
75	blank
76-80	Q-operand in machine language

Non-Imperative Statements

67-71	Length of assembled data
72	blank
73-80	If punched, actual assembled data

SPS Subroutine Set ID

Subroutine Set	Number
Divide subroutine for machines not equipped with the Automatic Divide feature. Does not call out or utilize the PICK subroutine.	00
Fixed length subroutines for machines equipped with the Automatic Divide feature.	01
Variable length subroutines for machines equipped with the Automatic Divide feature.	02
Variable length subroutines for machines equipped with the automatic floating point feature (the Automatic Divide feature is a prerequisite).	03

SPS Subroutine Macro Generated Linkage

LINE	LABEL	OPER-ATION	OPERANDS & REMARKS
010		TFM	PCK+10,*+19
020		B7	SUBR,,6
030		DSA	A,B
040		DC	1,@

SPS Subroutine Exponent Over and Underflow Control

Indicator in location 00401

$\neq$	no error
$\bar{1}$	exponent overflow
1	exponent underflow
$\bar{0}$	value cannot be calculated
0	FSIN or FCOS loss of accuracy or FSQR or FLN negative argument

SPS Subroutine Error Resultants

DESCRIPTION OF ERROR	CONTENTS OF 00401	RESULTANT FIELD
FA or FS, exponent overflow	$\bar{1}$	$\bar{9}9...9\bar{9}9$
FA or FS, exponent underflow	1	$00...0\bar{9}\bar{9}$
FM, exponent overflow	$\bar{1}$	$\bar{9}9...9\bar{9}9$
FM, exponent underflow	1	$\bar{0}0...0\bar{9}\bar{9}$
FD, exponent overflow	$\bar{1}$	$\bar{9}9...9\bar{9}9$
FD, exponent underflow	1	$\bar{0}0...0\bar{9}\bar{9}$
FD, attempt to divide by zero	0	If $0 \overline{)\, n}$: mantissa of dividend unchanged, exponent of dividend = E + 99 If $0 \overline{)\, 0}$: mantissa of dividend unchanged, exponent of dividend = -99
FSQR, input argument is negative	0	$\sqrt{\lvert x \rvert}$
FSIN or FCOS, input argument has exponent value greater than the mantissa length	$\bar{0}$	$\bar{0}0...0\bar{9}\bar{9}$
FSIN or FCOS, input argument has exponent value (X) such that 03 X L, where L is the mantissa length	0	SIN (X) or COS (X)
FEX or FEXT, exponent overflow	$\bar{1}$	$\bar{9}9...9\bar{9}9$
FEX or FEXT, exponent underflow	1	$\bar{0}0...0\bar{9}\bar{9}$
FLN or FLOG, attempt to take log of zero	$\bar{0}$	$\bar{9}9...\bar{9}\bar{9}9$
FLN or FLOG, input argument is negative	0	LN ($\lvert x \rvert$) or LOG ($\lvert x \rvert$)

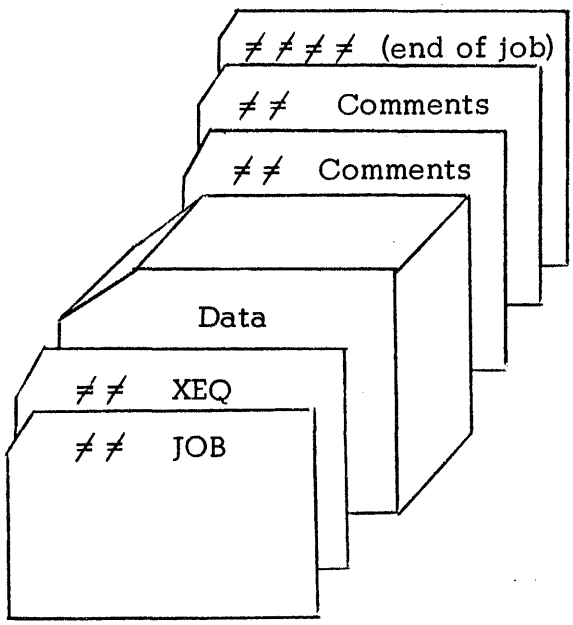
SPS Errors at Object Load Time

CORE CAPACITY EXCEEDED BY XXXXX LOCATIONS
PROGRAM IS TERMINATED

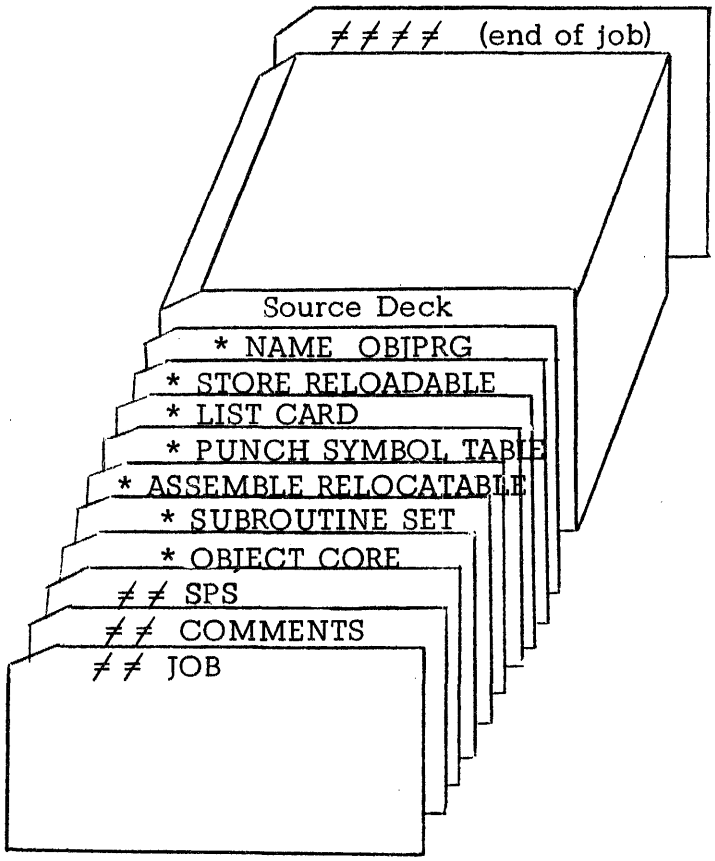
SUBR NOT LOCATED IN SUBROUTINE MAP

IMPROPER IND CODE IN SUBR XXXX
where XXXX is the first four digits of the subroutine I D no.

Typical SPS Deck Set-up



I. Execute program (stored on disk)



II. Assemble SPS program and store on disk.

SPS II D Modification Control Records

1	*
2-80	DEFINE OP CODE or DELETE OP CODE or DEFINE SYSTEM SYMBOL TABLE or LIST OP CODE or END LIB

Op Code Definition Card Format

12-15	Mneumonic op-code
16-75	Three digit defining code

Op Code Defining Codes

XY0	XY PPPPP QQQQQ	Any instruction
-XY1	Subr linkage to Subr XY	Subr macros
-XY2	8X PPPPP YQQQQ	SIOC instructions
-XY4	4X PPPPP Q00QY	Mask or Unmask
XY2	3X PPPPP Q07QY	Disk instructions
XY3	3X PPPPP Q0YQQ	I/O instructions
XY4	34 PPPPP Q0XQY	Control instructions
XY6	46 PPPPP QXYQQ	BI instructions
XY7	47 PPPPP QXYQQ	BNI instructions

Define Op Code Error Messages, Action

ALREADY DEFINED	Ignored
NO ROOM IN TABLE	Ignored

Delete Op Code Error Message, Action

NOT IN TABLE	No Change

Define System Symbol Table

Symbol Definition Record Format

6-11	Symbol to be defined
16-75	Symbolic (previously defined) or actual, but not asterisk, operand

Error Message, Action

UNDEFINED SYMBOL XXXXXX

As Distributed:

SYMBOL	EQUIVALENCE	DESCRIPTION
9RCYL0	00513	These are the low-order positions of four
9RCYL1	00515	2-digit fields which contain the numbers
9RCYL2	00517	of cylinders ($\bar{0}0$-$\bar{9}9$) where the disk access
9RCYL3	00519	arm is repositioned <u>after</u> a disk operation in which a reposition has been requested. The four fields refer to drives 0, 1, 2, 3, respectively.
9CCYL0	02099	These are the low-order positions of four
9CCYL1	02101	2-digit fields, similar to the previous four.
9CCYL2	02103	However, these positions contain the cylinder
9CCYL3	02105	numbers of the <u>current</u> access arm positions (the position of the arm after the last disk IORT operation).

PICK Subroutine Parameter Locations

Location	Content
02375	Return address of main line program
02380	Address of A operand data
02385	Address of B operand data
02398	Mantissa length in use
02401	Noise digit in use

User Written SPS Subroutine ID Number

Five digits as operand of DEND statement

XXNNE

where XX is Subr set no.
NN is Subr no.
E is no. of entry points

Example:

```
DEND  02182
```

Specifying Entry Points in User Written SPS Subroutine

Example: An entry of this type must appear at the beginning of source statements.

```
ORIGIN  DSA ENTRY 1, ENTRY 2, etc.
        DORG ORIGIN-4
```

where ENTRY 1, ENTRY 2, etc. are the labels of the subroutine entry points.

Adding SPS Subroutines

Modification with Respect to PICK and/or Mantissa Length

One DC statement can modify up to 25 instructions. Each instruction, whether or not it is to be modified, requires two digits in the pseudo constant , one for the P operand and one for the Q operand. The statement itself consists of three operands: the first specifies the length of the constant which may not be greater than 50 nor less than 2; the second, the actual constant; the third, the storage address of the constant. This address must be specified as an absolute address of 00350. The P and Q operand modifier constants follow:

P and Q Operand Modifiers	Modification
0	No Modification
1	Add L
2	Subtract L
3	Add 2L
4	Subtract 2L
5	Modify with respect to PICK, no L modification
6	Modify with respect to PICK, add L
7	Modify with respect to PICK, subtract L

8	Modify with respect to PICK, add 2L
9	Modify with respect to PICK, subtract 2L

The following examples shows how a variable length mantissa subroutine may be modified, by use of modifier constants.

```
DC      6, 527005, 350
TF      SAVE, 98
SF      SAVE
TF      PCK + 25, SAVE, 6
TF      PCK + 15, CHAR, 6
B7      PCK + 10 , , 6
```

NOTE:

Intervening DORG statements and constants between instructions are never modified in this manner.

SPS Subroutines

Base DIM Numbers

Base Number	Subroutine Set
130	Fixed Point Divide - 00
100	Fixed Length - 01
70	Variable Length - 02
40	Automatic Floating Point - 03

Example of Adding a Subroutine to SPS Library

MACRO: EXCH A , B

FUNCTION: To exchange floating point numbers between A and B

IDENTITY: Subr No 18, Set No 02

1. Modify op code table by

   A. Load Supervisor Program

   B. XEQ Monitor Control Statement specifying SPS LIB

Cols	1,2	≠≠
	3-5	XEQ
	7-12	SPSLIB

   C. DEFINE OP CODE Control record

   D. Op code definition Control record

Cols	12-15	EXCH
	16-19	-181

   E. ENDLIB statement

   NOTE: Statements B,C,D & E must all be entered from the same input device.

2. Write the subroutine in SPS language (See page 31.)

3. Use the following control records for assembling and loading the subroutine to disk ready for use.

   A. Supervisor Program Load

   B. SPS Monitor Control Statement

Cols	1,2	≠≠
	3-5	SPS

Example of Adding a Subroutine to SPS Library
Page 2

C. SPS Control Statements

(1) ASSEMBLE RELOCATABLE

(2) LIBR

(3) ID NUMBER 0088

70 + 18 = 88

(4) STORE RELOADABLE

(5) LIST CARD

(6) ERROR STOP

IBM

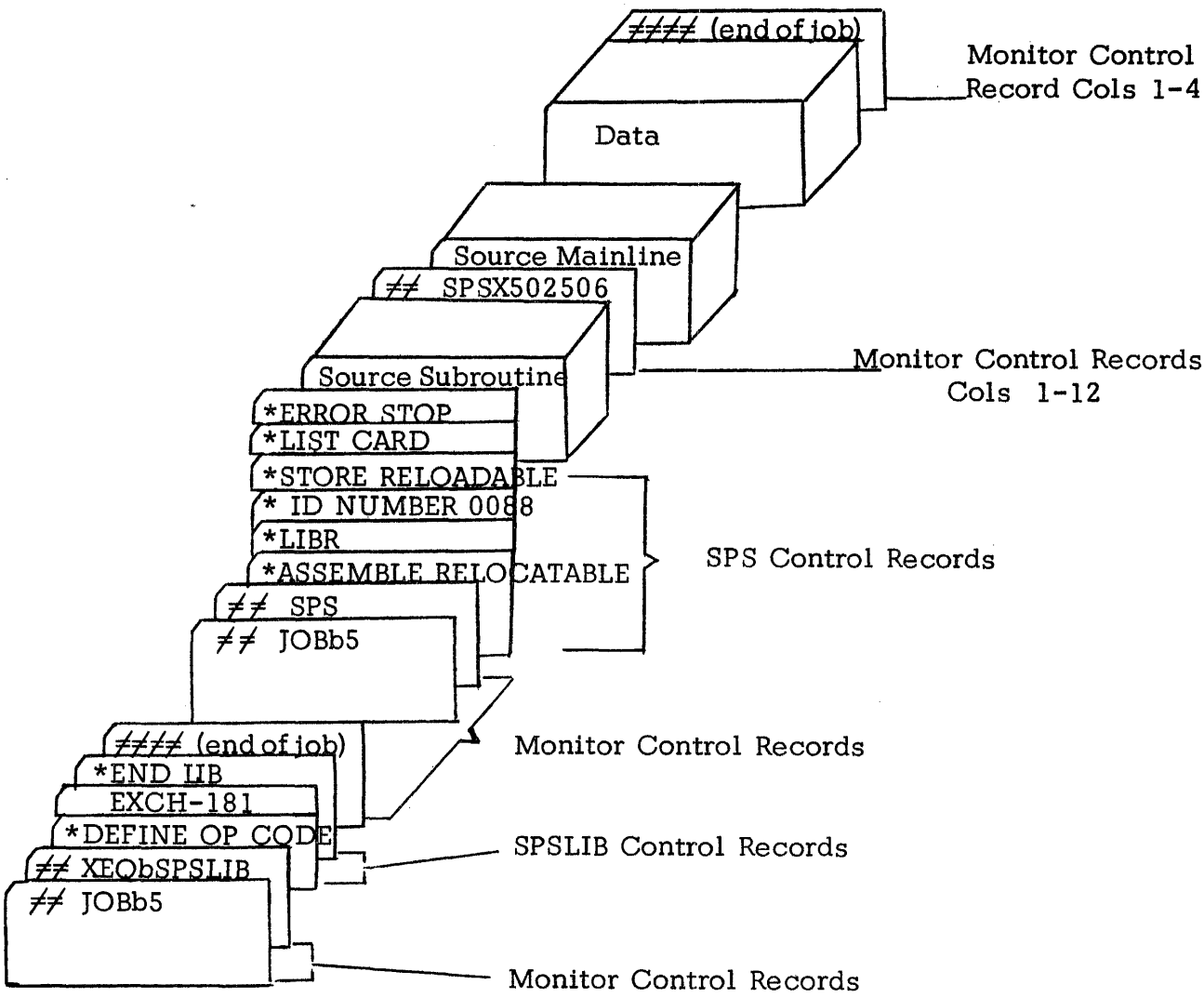
1620/1710 Symbolic Programming System
Coding Sheet

Program: EXCHANGE A AND B SUBRØUTINE Date: _____ Page No. 1 2 of _____

Routine: _____ Programmer: _____

Line	Label	Operation	Operands & Remarks
3 5	6 11	12 15	16 20 25 30 35 40 45 50 55 60 65 70 75
010			
020		DSA	ENTRY
030	*		DSA IS USED TØ DEFINE SUBRØUTINE ENTRY PØINT
040	ZERØ	DØRG	*-4 , , ,DEFINES RELØCATABLE ZERØ
050	BETA	DS	,ZERØ+196 , ,DEFINES RELØCATABLE SYMBØL
060	PCK	DS	,2365 , ,DEFINES ABSØLUTE SYMBØL
070	ENTRY	TFM	PCK+5 ,*+20 , ,
080		B7	PCK , ,6 ,LINKAGE TØ PICK
090		TF	PCK+20 ,PCK+15 ,611 ,MØVE A EXP TØ B ØPERAND EXP PØSITN
100		TF	PCK+30 ,PCK+25 ,611 ,MØVE A MANT TØ B ØPERAND MANT PØSIT
110		DC	4 ,0505 ,350 ,PSEUDØ CØNSTANT FØR PICK AND L REF
120		TF	PCK+15 ,BETA ,6 ,MØVE B TØ A
130		TF	PCK+25 ,BETA-2 ,6 ,NØTE THAT PICK PUT B IN BETA
140		B7	PCK+10 , ,6 ,RETURN TØ MAINLINE
150		DEND	02 18 1 , , ,ID FØR SUPERVISØR
160			
170			
180			
190			
200			

Stacked Input for Adding a Subroutine, Assembling and Executing a Program That Requires the Added Subroutine



FORTRAN II-D

TABLE OF CONTENTS

FORTRAN II-D LIBRARY SUBROUTINES

Type of Function	Symbolic Name	Subroutine Number	DIM Entry Number	Transfer Address
Logarithm (natural)	LOGF	1	10	02246
Exponential	EXPF	2	11	02251
Subscripting (1 dimension)	ENTSCI	3	12	02256
Subscripting (2 dimensions)	ENTSC2	4	13	02261
Subscripting (3 dimensions)	ENTSC3	5	14	02266
FIND	ENTFID	6	15	02271
RECORD	ENTREC	7	16	02276
FETCH	ENTFET	8	17	02281
Routine to load or unload disk buffer	ENTSWD	9	18	02286
Routine to write or read arrays	ENTARR	10	19	02291
Routine to complete FETCH or RECORD	ENTCPT	11	20	02296
Cosine	COSF	12	21	02301
Sine	SINF	13	22	02306
Arctangent	ATANF	14	23	02311
Square Root	SQRTF	15	24	02316
Absolute Value	ABSF	16	25	02321

FORTRAN ARITHMETIC AND INPUT/OUTPUT SUBROUTINES

Subroutine	Symbolic Name	Operation
Floating Point Arithmetic		
Add	FAD	FAC + A ⟶ FAC
Subtract	FSB	FAC - A ⟶ FAC
Reverse Subtract	FSBR	A - FAC ⟶ FAC
Multiply	FMP	FAC x A ⟶ FAC
Divide	FDV	FAC / A ⟶ FAC
Reverse Divide	FDVR	A / FAC ⟶ FAC
Set FAC to zero	ZERFAC	0 ⟶ FAC
Fixed Point Arithmetic		
Add	FXA	FAC + I ⟶ FAC
Subtract	FXS	FAC - I ⟶ FAC
Reverse Subtract	FXSR	I - FAC ⟶ FAC
Multiply	FXM	FAC x I ⟶ FAC
Divide	FXD	FAC / I ⟶ FAC
Reverse Divide	FXDR	I / FAC ⟶ FAC
Common Subroutines		
Load FAC	TOFAC	A → FAC or I → FAC
Store FAC	FMFAC	FAC → A or FAC → I
Reverse Sign of FAC	RSGN	- FAC → FAC
Fix a Floating Point Number	FIX	FIX (FAC) → FAC
Float a Fixed Point Number	FLOAT	FLOAT (FAC) → FAC
Exponentiation		
Fixed Point J ** I	FIXI	FAC ** I ⟶ FAC
Floating Point A ** (±I)	FAXI	FAC ** (±I) → FAC
Floating Point A ** (±B)	FAXB	FAC ** (±B) → FAC
Input/Output		
Read Card	RACD	
Read Tape	RAPT	
Read Typewriter	RATY	
Write Card	WACD	
Write Tape	WAPT	
Write Typewriter	WATY	

3408
3452

FAC - simulated accumulator
A & B - floating point variables
I & J - fixed point variables
⟶ - store in

FORTRAN II-D SUBROUTINE SET I. D.

1. no floating-point feature - read in when required
2. no floating-point feature - in core
3. floating-point feature - read in when required
4. floating-point feature - in core

Any digit higher than four will result in the error message below:

ER L8

NEW FORTRAN SUBROUTINE ERROR CONDITIONS

Code	Cause	Corrective Action
D1	Disk I/O used without a DEFINE DISK statement	Typewriter calls for correction in format NNXXXXX NN = N1 XXXXX = N2
D2	Logical record exceeds N2	Record not written. Argument may be incorrect.
D3	No group mark found at end of array that was read from disk.	

FORTRAN II-D CONTROL RECORD FORMATS

Positions	Parameters	Punch Symbol Table	Punch Object Prog.	Load to Disk
1	*	*	*	*
2-6	FANDK	PSTSN	POBJP	LDISK
7		n	n	
7-8	ff			
9-10	kk			
7-12				name
13-16				DIM no.

where n = 2 means output on paper tape
n = 4 means output on cards
ff is the floating mantissa length (02-28)
kk is the fixed length (04-10)

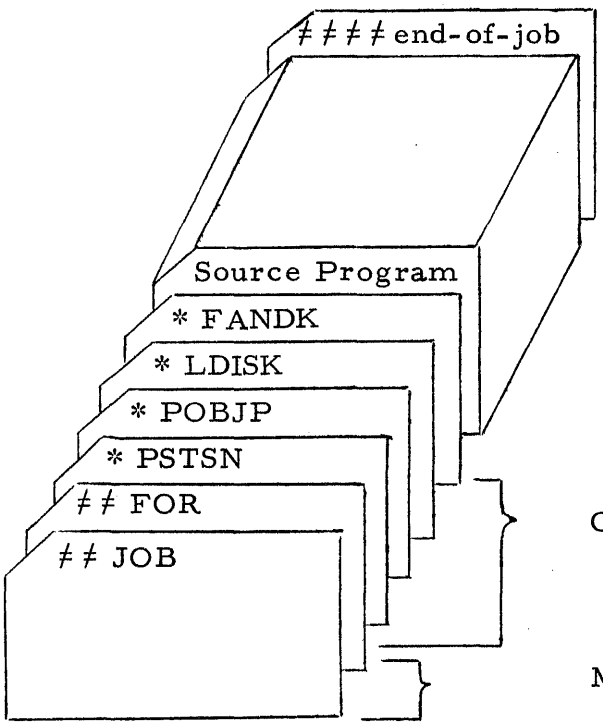
Error Messages

ERROR, INVALID CONTROL RECORD
F OR K OUTSIDE RANGE
ERROR, INVALID OUTPUT UNIT CODE

Error Correction Procedure

Ready, correct record in input unit, start.

FORTRAN COMPILATION DECK SET-UP



Monitor Control Record

Source Deck

Optional FORTRAN Control Records

Monitor Control Records

CONSOLE SWITCH USE BY FORTRAN

A. Compilation - if on:

1. Type symbol table
2. Compile arithmetic trace
3. Compile IF trace
4. Goof switch

B. Execution - if on:

4. Type trace output

NEW FORTRAN COMPILATION ERRORS

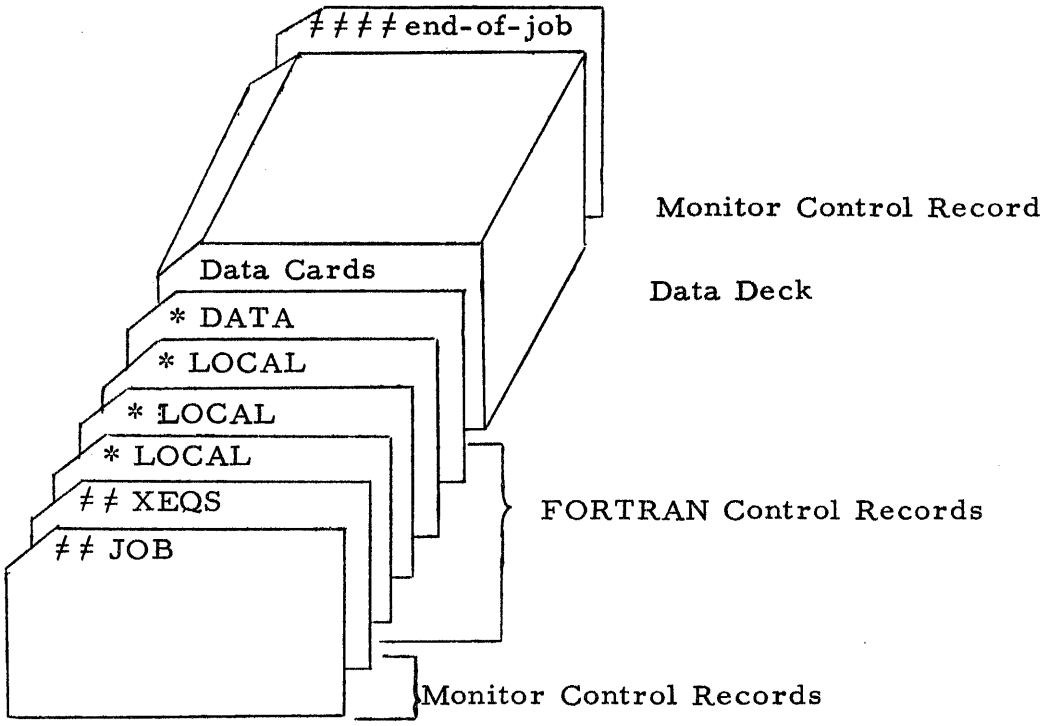
TYPE I

41	Syntox error in disk I/O statement
42	Disk I/O list omitted
43	Disk I/O list contains both simple and array names
44	COMMON exceeds core storage size

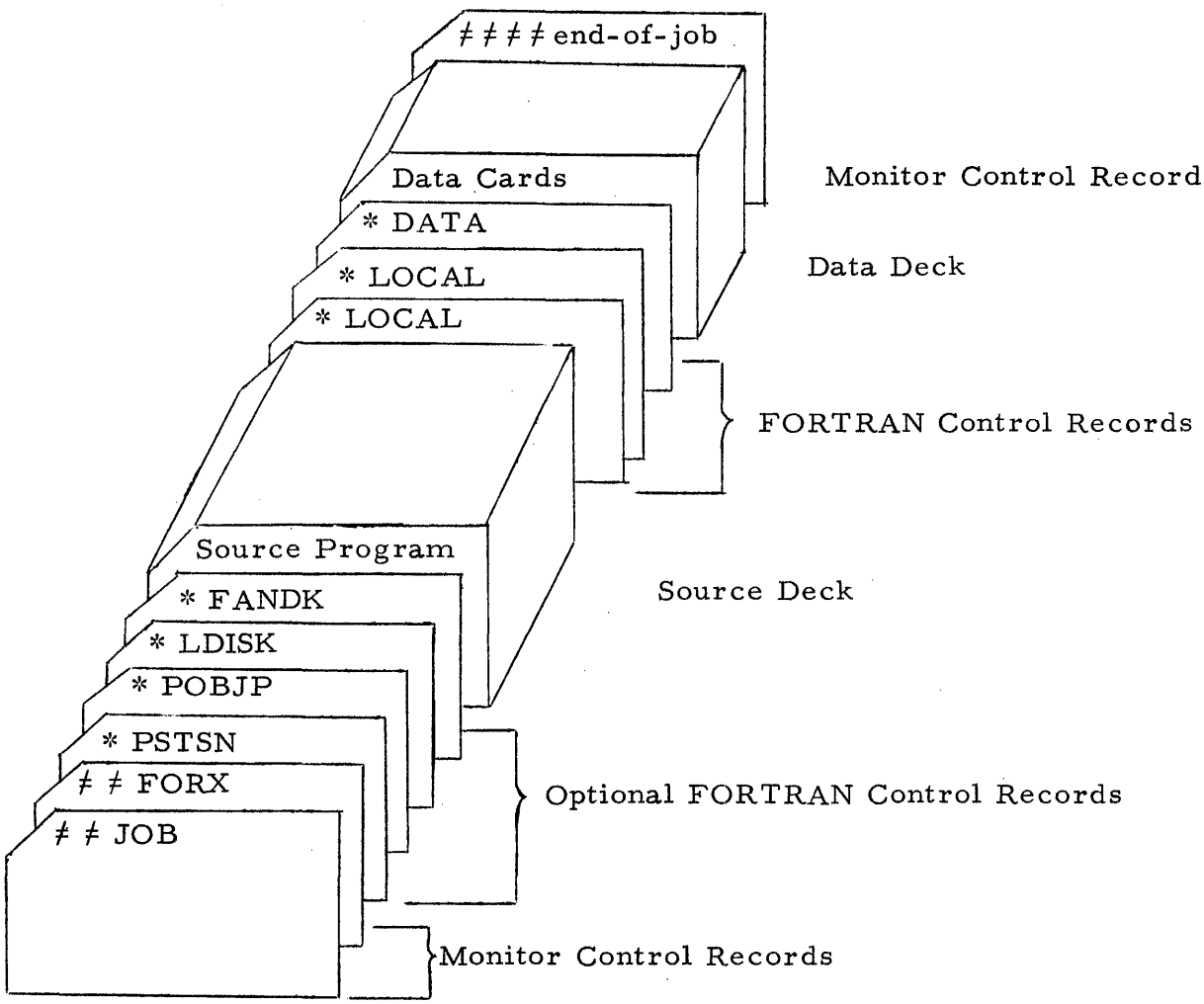
TYPE II

60	DEFINE DISK statement missing syntax error, or invalid use

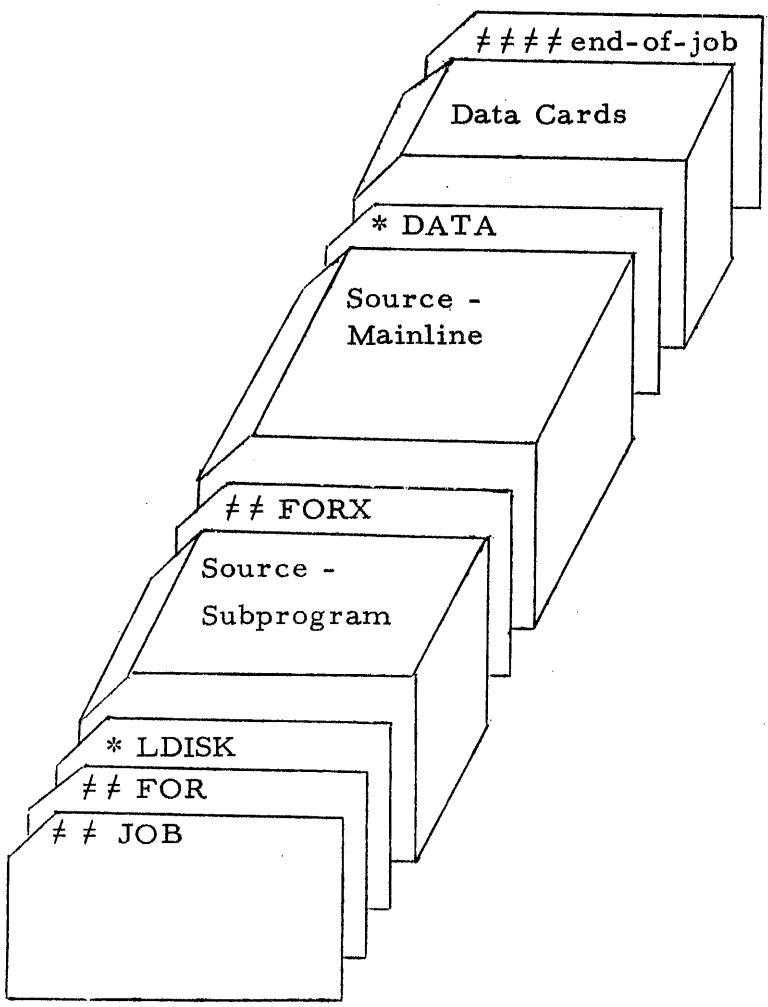
FORTRAN OBJECT PROGRAM EXECUTION FROM DISK



FORTRAN COMPILATION AND EXECUTION DECK SET-UP



FORTRAN SUBPROGRAM AND MAINLINE COMPILATION AND EXECUTION



ERRORS DURING FORTRAN OBJECT LOADING

LOAD XXXXXX	where XXXXXX is the name of a called subprogram which the loader cannot find in the equivalence table.
NNNNNN XXXXX OVERLAP JOB ABANDONED	Program or link NNNNNN requiring XXXXX locations would overlap storage. If program has no name $ is printed instead.
NN XXXXX OVERLAP	Subroutine no. NN requiring XXXXX locations would overlap storage
FLIPER XXXXX OVERLAP	FLIPER is the name of the LOCAL Subprogram control program. XXXXX is its length.

FORTRAN LOADER ERROR CODE

Error Code	Meaning, Reason	Result
L1	Invalid LOCAL control record Word LOCAL misspelled, misplaced, or no asterisk	Halt, branch to MONCAL*
L2	Invalid subprogram name in LOCAL record Not formed according to FORTRAN rules	Halt, branch to MONCAL*
L3	Multiple subprogram name in LOCAL record Same subprogram name appears more than once for same program or link	Halt, branch to MONCAL*
L4	LOCAL subprogram table full Greater than 50 LOCAL subprograms not allowed	Ignore above 50th LOCAL subprogram
L5	Invalid header record Does not conform to standard FORTRAN header record	Branch to MONCAL*
L6	Unequal F or K Subprogram F and/or K does not compare with main program F or K	Subprogram not loaded
L7	New subroutine called from LOCAL subprogram LOCAL subprogram cannot call new subroutine	Subprogram loaded, subroutine not loaded
L8	Invalid arithmetic and input/output subroutine set Not defined as 1, 2, 3, 4	Set 1 or 3 loaded, depending on version loading
L9	In-core subprogram table full Greater than 50 subprograms not allowed	Ignore above 50th subprogram
L10	New subprogram called from LOCAL subprogram LOCAL subprogram cannot call new subprogram	LOCAL subprogram loaded; new subprogram not loaded

*MONCAL is the symbolic name for Monitor Control Record Analyzer routine.

FORTRAN OBJECT TIME CORE MAP

00000 - 00099	Product Area, etc.
00100 - 00401	Arith Tables
00402 - 02401	Supervisor Communication Area
02402 - 07500 or 14000	Arith and I/O Subroutines
	Mainline Program
	In-core Subprograms
	Library Subroutines
	LOCAL Subprogram Loader
	LOCAL Subprogram Area
	Unused Area
	COMMON Area

FORTRAN OBJECT TIME DISK WORK STORAGE MAP

218 Sectors	Arith and I/O Blocking Area
	Object Program Working Storage
	Unused Sectors
	LOCAL Subprogram hold area
	Program Call Lists
21 Sectors	COMMON Save Area

PROCEDURE FOR ADDING A SUBROUTINE TO THE FORTRAN LIBRARY

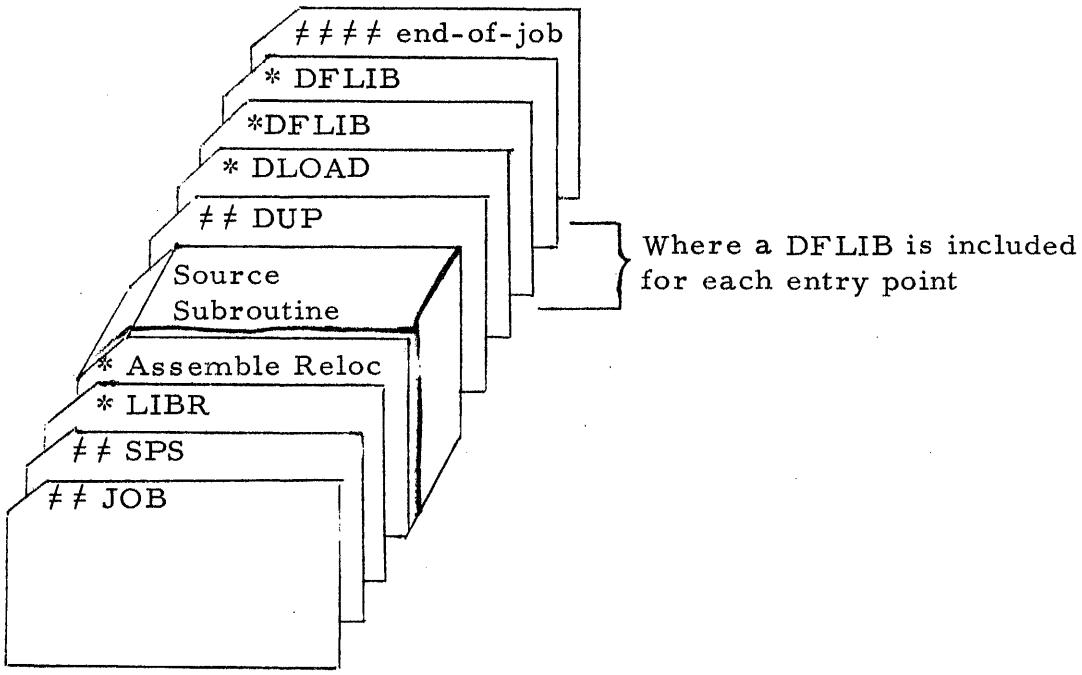
1. Write subroutine in SPS 2-D

2. The following or equivalent must appear as the first statements.

```
SUB   DSA   ENTRY1,ENTRY2,.....,ENTRY9
      DORG  SUB-4
```

   where ENTRY1, etc. are the symbolic entry points to the subroutine up to a maximum of nine.

3. Assemble the following deck or equivalent



PROCEDURE FOR WRITING FORTRAN SUBPROGRAMS

IN SPS 2-D

1. Write subprogram in SPS 2-D
2. Include the SPS equivalent of a FORTRAN subprogram header record as the first statement

```
S     DS      ,101
      DC      6,987898,5-S
      DAC     6,NAMEbb,7-S
      DVLC    22-S,5, LAST, 2, ff, 2, kk,
              5, Entry Address-6, 5, 0, 30, 0,
      DC      17,01234567891234567,316
      DORG    S-100
```

where LAST (length of program) is symbolic or actual for the last-plus-one location of (or used by) the program.

3. If the subprogram calls library subroutines, the 30 digit constant, shown as zero, will require a one-digit in each position corresponding to the subroutine called.

4. Linkage to library subroutines is coded

```
BTM    LIBSUB, A      where
```

A is the name of the operand. The function is returned to FAC.

5. If the subprogram calls other subprograms, the following statements must occur just prior to the DEND but should not be included in the header record as part of the program length:

```
DAC    6, NAMEAA
DSA    N1LOC
```

where NAMEAA is the name of the called subprogram and N1LOC is the label of a reserved address field within the calling subprogram.

These two statements must be repeated for each called subprogram.

All sets should be followed by:

```
DC     1,@, *+1
DEND   START
```

6. Linkage to subprograms is similar to

```
GGG          BTM    NAMEAA, *+11
             DSA    ARG1, ARG2, ....
(N1LOC       DS     ,GGG+6 )
```

7. If the subprogram is to act as a FUNCTION rather than a SUBROUTINE, then the following statement should appear prior to the return:

```
BTM   TOFAC, ANS
```

8. Examples of methods of picking up subprogram arguments follow.

WRITING FORTRAN SUBPROGRAMS IN SPS - OBTAINING PARAMETERS

In addition to the header record, linkage to obtain the subprogram parameters must be included in the subprogram. If no parameters are needed, or if the subprogram knows the location of the parameters, the user writes:

```
          DC    5,0
SUBNAM    AM    SUBNAM-1,1,10
                                  } The subprogram
          B     SUBNAM-1,,6
```

If one parameter is needed, and this subroutine is never nested within any other function or subprogram, the user writes:

```
          DC    5,0
SUBNAM    AM    SUBNAM-1,5,10
          TF    INSUB, SUBNAM-1,11
          AM    SUBNAM-1,2
                                  } The subprogram
          B     SUBNAM-1,6
```

If several parameters are to be moved to the subprogram, a loop may be utilized to conserve core storage. The parameters must be stored in the subprogram in consecutive order. An example of the coding to accomplish this for three parameters is shown below:

```
INSUB     DSA     0,0,0
          DC      1,@
          DC      5,0
SUB       TFM     TF+6, INSUB-4
          AM      TF+6, 4, 10
          AM      SUB-1, 5, 10              (1)
          TF      CF+11, SUB-1,11           (2)
          BNF     *+36, CF+11               (3)
CF        CF      CF+11                     (4)
          TF      CF+11, CF+11, 11          (5)
```

```
TF      TF      INSUB, CF+11            (6)
        AM      TF+6, 1, 10
        BNR     SUB+12, TF+6, 11
        AM      SUB-1, 2, 10            (7)

        B       SUB-1,,6                (8)
```

The instructions that constitute the body of the subprogram are placed between number 7 and 8 above. Instruction 7 must add "two" if the number of parameters is an odd number, or "one" if this number is even.

The instruction numbered 8 (B SUB-1, , 6) returns control to the calling program. When writing subprograms in SPS the user must place this instruction at every point that a return is required.

FORTRAN OBJECT PROGRAM HEADER RECORD

Mainline or Link

$\overline{0}0100\overline{0}$	2	$\overline{6}7$	$\overline{9}87898$	$\overline{N}1$	$\overline{N}2$	$\overline{W}$ord Length	$\overline{R}$ec Len
5	1 2	6		2	5	2	1

digits

Reserved	LENGTH	$\overline{f}f$	$\overline{k}k$	$\overline{F}$irst Core	$\overline{N}$ext Common	Subroutine Indicators
2	5	2	2	5	5	30

Subprogram

$\overline{0}0100\overline{0}$	2	$\overline{6}7$	$\overline{9}87898$	$\overline{S}$ubprogram Name
5	1 2	6		12

LENGTH	$\overline{f}f$	$\overline{k}k$	$\overline{E}$ntry Address Less Six	$\overline{N}$ext Common	Subroutine Indicators
5	2	2	5	5	30

$\overline{0}0100\overline{0}$	The address of the origin of the program less 100.
2	An indicator to the relocating loader that a constant to be relocated is forthcoming.
$\overline{6}7$	The number of digits in the forthcoming constant.
$\overline{9}87898$	An arbitrarily chosen constant to identify this as a header record for a FORTRAN program.
Subprogram Name	The name of the program in double digit alpha format right justified. Only used in subprograms and FORTRAN function headers.
N1	The number of words per disk record. (From the DEFINE DISK FORTRAN statement) This field is only present for mainline programs and links.
N2	The number of logical records in the disk, as used by the FORTRAN program. (From the DEFINE DISK FORTRAN statement) This field is only present in mainline program and link header records.

Word Length	The number of digits in the words used to determine a logical record. This value is the larger of the floating word and the fixed word length. Only found in mainline and link header records.
Rec. Len	The number of sectors to be used when reading or writing logical records. This value is limited to the numbers 1 and 2. Only found in mainline and link header records.
Reserved (2 Positions)	This field is reserved for system use in the mainline and link headers.
LENGTH	The length of the program (This must be an even number)
ff and kk	The length of the mantissa and the fixed point words in this program.
Entry Address Less Six	The first location in the subprogram less six to enter the subprogram.
First Core	The first location in the program to begin execution. Present only for mainline programs.
Next Common	The next location available in COMMON. (i.e., 19999) This field is used only if the program is a mainline program.
Sub. Indicators	A digit position for each library subroutine name in the FORTRAN System (i.e., SINF, SQRTF, etc.)

This completes the data outputted on one output record.

The identification record occupies one whole sector when it is on the disk. The balance of this sector beyond the record shown above is ignored by the subprogram loader. The format for the balance of this sector, if the program is in relocatable format, is shown below:

$\overline{0}0300\overline{2}\overline{1}701234567891234567$

FORTRAN LOCAL AND DATA CONTROL RECORD FORMATS

CHARACTER	LOCAL	DATA
1	*	*
2-6	LOCAL	DATA b
7-80	Freestyle: (1) Mainline Link Name (2) Comma (3) Local Subprogram Name (4) repeat (2) and (3) as often as required	blank

Printed in U.S.A.

R20-8005-1

IBM®

International Business Machines Corporation
Data Processing Division
112 East Post Road, White Plains, N.Y. 10601
(USA Only)

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
(International)