Honeywell

SERIES 6000

GENERAL MACRO ASSEMBLY PROGRAMMING (GMAP) COURSE CODE 605

STUDENT HANDBOOK

Honeywell

SERIES 6000

GENERAL MACRO ASSEMBLY PROGRAMMING (GMAP) COURSE CODE 605

STUDENT HANDBOOK

Order Number: AF52, Rev. 1

File Number: 2ZZ2

SERIES 6000

GMAP PROGRAMMING

COURSE 605

STUDENT HANDBOOK

REVISED: 10/1/72

COURSE NAME: GMAP PROGRAMMING
COURSE CODE: 605

STUDENT HANDBOOK
TABLE OF CONTENTS

COURSE NAME: GMAP PROGRAMMING
COURSE CODE: 605

STUDENT HANDBOOK
TABLE OF CONTENTS

COURSE NAME: GMAP PROGRAMMING
COURSE CODE: 605

STUDENT HANDBOOK
TABLE OF CONTENTS

COURSE NAME: GMAP PROGRAMMING
COURSE CODE: 605

STUDENT HANDBOOK
TABLE OF CONTENTS

COURSE NAME: GMAP PROGRAMMING
COURSE CODE: 605

STUDENT HANDBOOK
TABLE OF CONTENTS

COURSE NAME: GMAP PROGRAMMING
COURSE CODE: 605

STUDENT HANDBOOK
TABLE OF CONTENTS

COURSE NAME: GMAP PROGRAMMING
COURSE CODE: 605

STUDENT HANDBOOK
TABLE OF CONTENTS

COURSE NAME: GMAP PROGRAMMING
COURSE CODE: 605

STUDENT HANDBOOK
TABLE OF CONTENTS

COURSE NAME: GMAP PROGRAMMING
COURSE CODE: 605

STUDENT HANDBOOK
TABLE OF CONTENTS

COURSE NAME: GMAP PROGRAMMING
COURSE CODE: 605

STUDENT HANDBOOK
TABLE OF CONTENTS

COURSE NAME: GMAP PROGRAMMING
COURSE CODE: 605

STUDENT HANDBOOK
TABLE OF CONTENTS

Course Name: GMAP Programming
Course Code: 605

General Course Outline

I. INTRODUCTION

A. Introduction of instructor, students, course

B. Class Administration

C. Course Outline

II. OVERVIEW OF 6000

A. Hardware Overview

B. Software Overview

C. General Characteristics of Processor

III. PROGRAM ORGANIZATION

A. GMAP Coding Form and Symbolic Card Format

B. Programming Conventions

C. Submitting GMAP Programs

D. Typical Deck Setups

E. Requirements for Submitting GMAP Programs

IV. DATA REPRESENTATION

A. Internal Data Representation

B. Processor Handling of Information

C. Instruction Formats

D. Data Word Formats

E. Introduction to Pseudo Operations

1. Definition

2. Functional Groups

F. Storage Allocation Pseudo Operations

G. Data Generation Pseudo Operations

H. Summary

V. INTRODUCTION TO INSTRUCTIONS

A. Format

B. Types

C. Data Movement

D. Introduction to Address Modification

E. Using Address Modification

F. Direct Operands

G. Effective Address Instructions

H. Store Register Instructions

VI. LITERALS

A. Introduction

B. Decimal Literals

C. Octal Literals

D. Alphanumeric Literals

E. Instruction Literals

F. Variable Field Literals

G. Literals Modified by Direct Operand

VII. INSTRUCTION REPERTOIRE

A. Review

B. Instructions to be Discussed

C. Miscellaneous Store Instructions

D. Register Load and Store

XIII. MACRO OPERATIONS

- A. Introduction
- B. Defining the Macro Prototype
- C. Using the Macro Prototype
- D. Arguments for Macros
- E. Created Symbols
- F. Nesting of Macros
- G. Pseudo Operations within Macros
- H. System Macros

XIV. EXTERNAL AND INTERNAL INTERRUPTS

- A. Execution of Interrupts
- B. Internal Interrupts
- C. Categories of Internal Interrupts
- D. Master Mode Entry Routines
- E. MME's for Activity Termination
- F. MME GESNAP

XV. INTRODUCTION TO INPUT/OUTPUT

- A. Levels of Input/Output in 6000
- B. Necessary Ingredients of I/O Coding
  1. Logical File Designator
  2. Input/Output Buffer
  3. Data Control List
  4. Input/Output Command

XVI. File and Record Control (GFRC)

- A. Introduction to Input/Output
- B. Introduction to File and Record Control (GFRC)

C. File Control Records

D. Standard System Formay

E. Logical Record Formats

F. File Control Block

G. File Designator Word

H. File and Record Control (GFRC) Calling Sequences

I. Functional Groups of Calling Sequences

J. Calling Sequences for File Preparation

K. Logical Record Processing

L. Input/Output Editor

M. Physical Record Processing

XVII. FLOATING POINT

A. Introduction

B. Purpose

C. Load and Store

D. Arithmetic Instructions

E. Miscellaneous Instructions

F. Compare Instructions

G. Summary of Arithmetic Operations

XVIII. REVIEW OF INSTRUCTIONS

A. Data Movement

B. Fixed-Point Arithmetic

C. Boolean Operations

D. Comparison

E. Transfer of Control

F. Miscellaneous Operations

XIX. ADDRESS MODIFICATION - REVIEW AND EXTENSION

- A. Review of Modification Capabilities
- B. Register Modification
- C. Indirect then Tally Modification
- D. Register then Indirect Modification
- E. Indirect then Register Modification
- F. Coding Examples

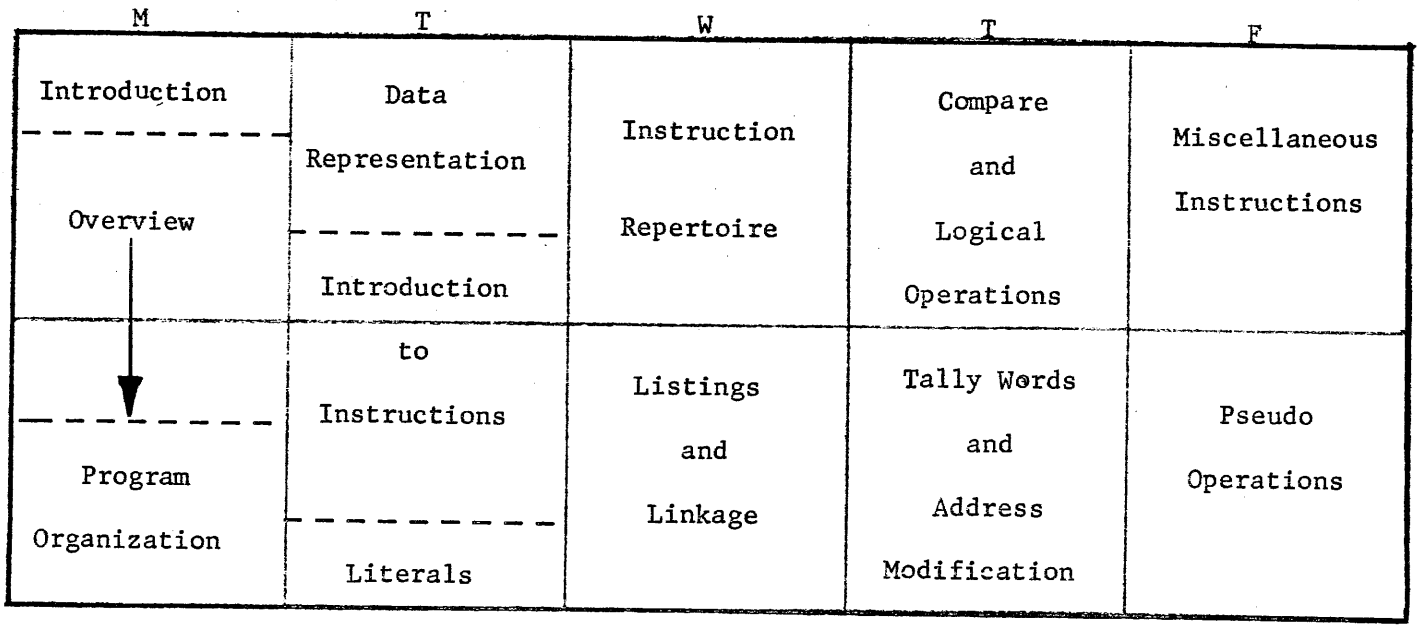
XX. EXTENDED INSTRUCTION SET

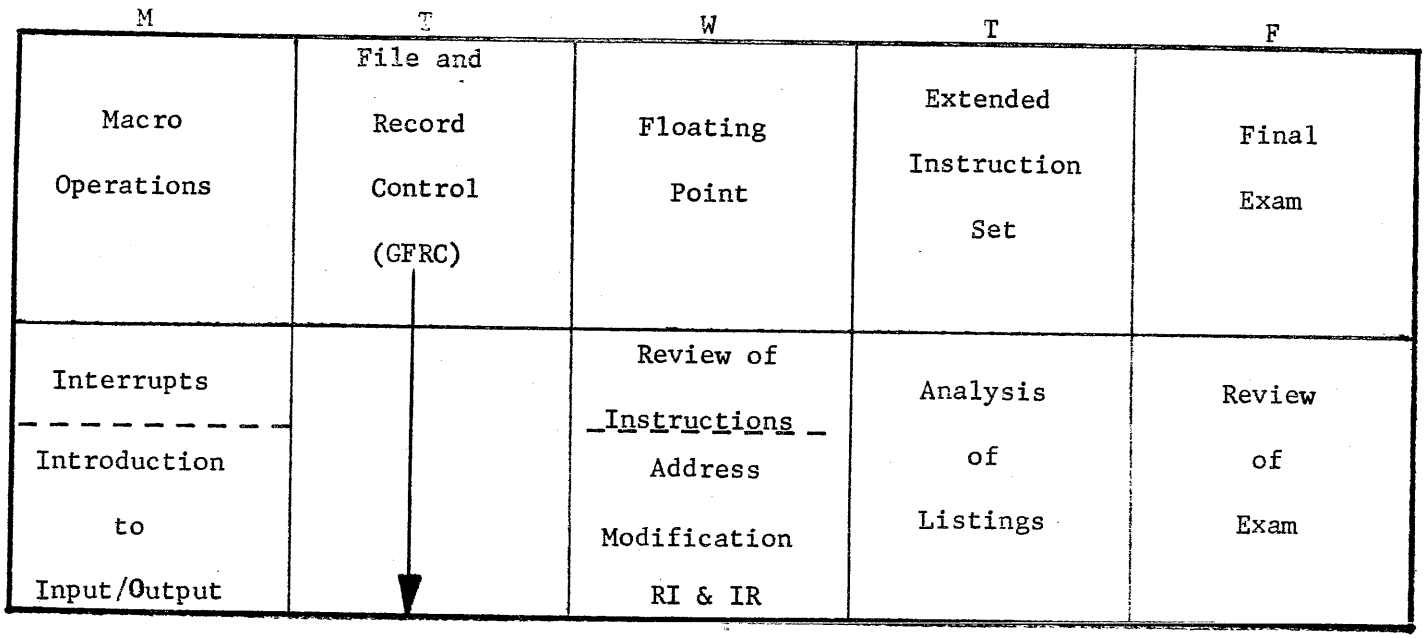
- A. Introduction
- B. Data Word Formats
- C. Instruction Word Formats
- D. Multi-word Instructions
- E. Single Word Instructions
- F. Micro Operations

XXI. ANALYSIS OF LISTINGS

- A. Review
- B. Files Produced by System Input (GIN)
- C. Accounting Information
  1. Job Information
  2. Activity Information
- D. Assembly Listing
- E. Memory Map

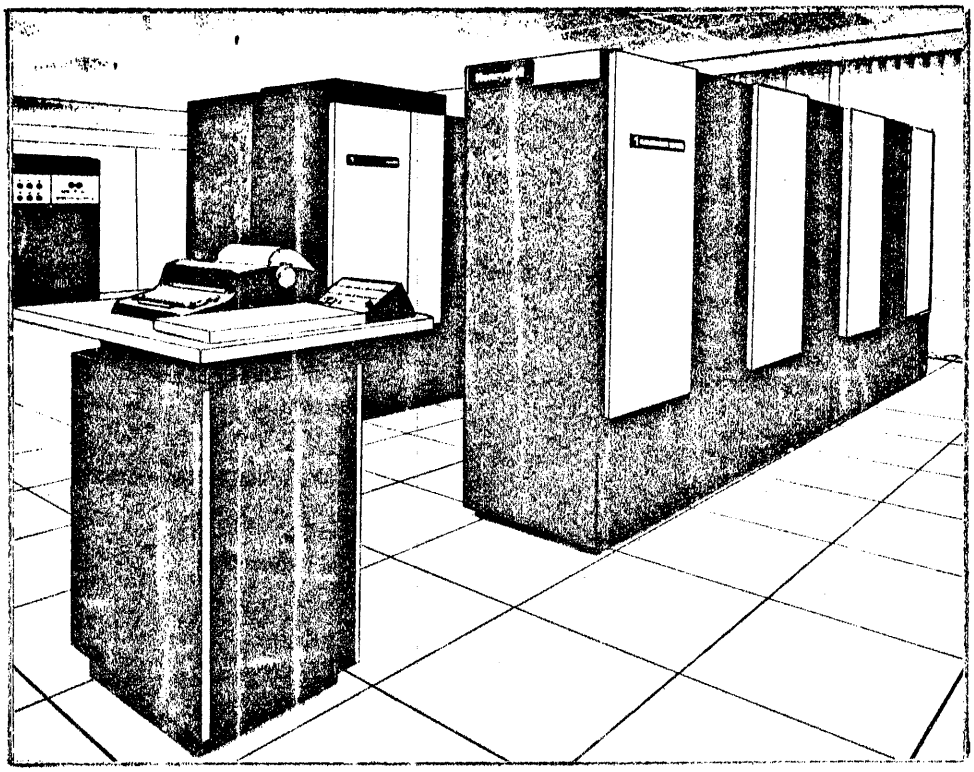
605 COURSE TOPIC MAP



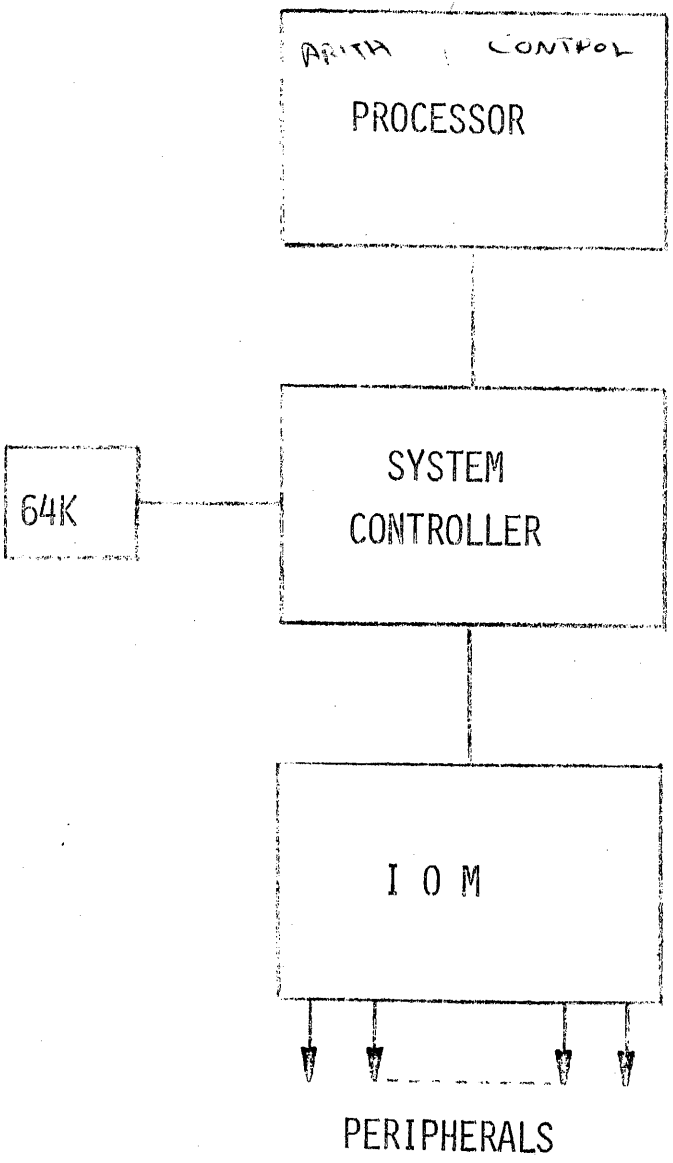


Honeywell

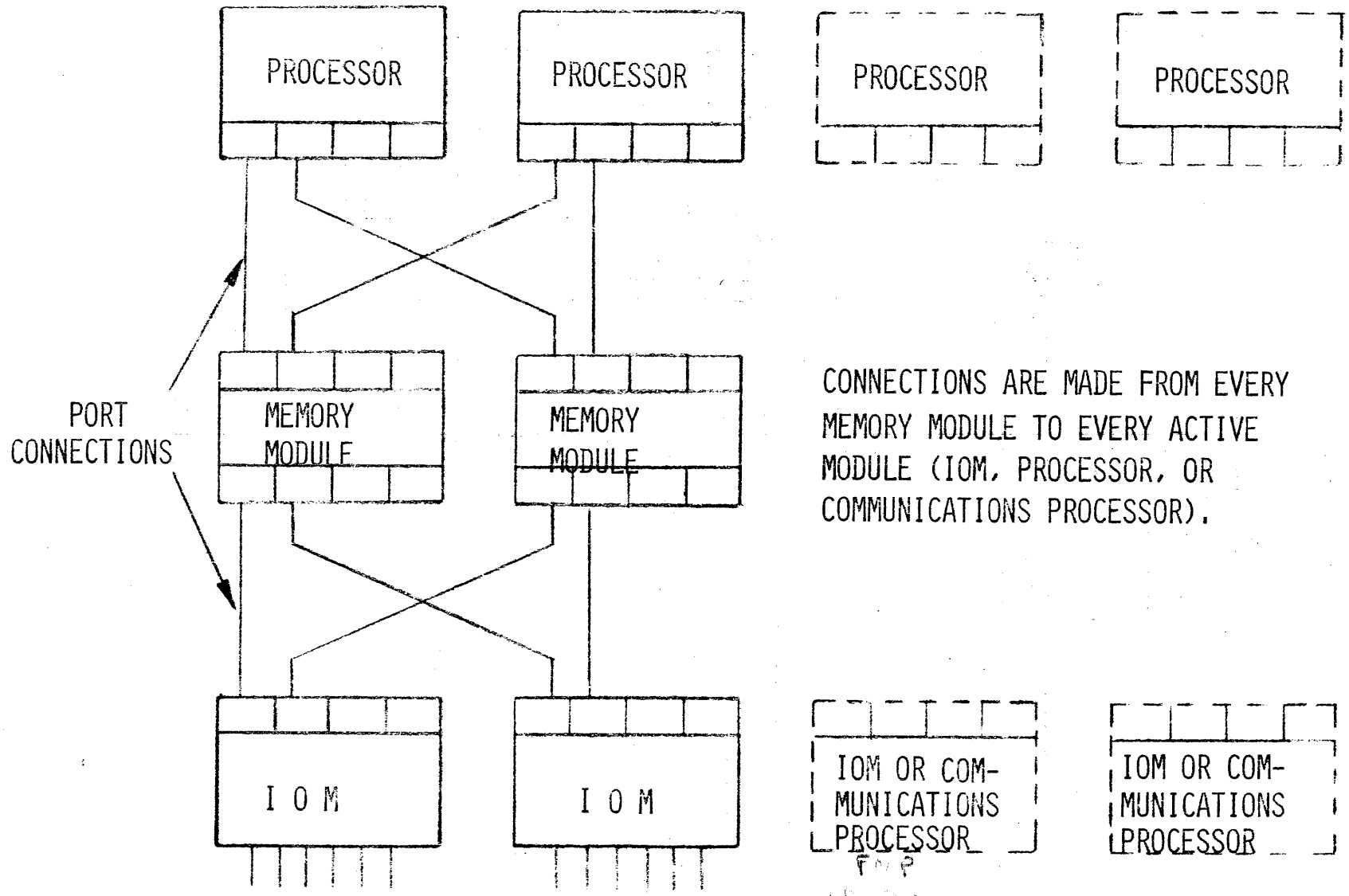
SERIES 6000



BASIC SYSTEM

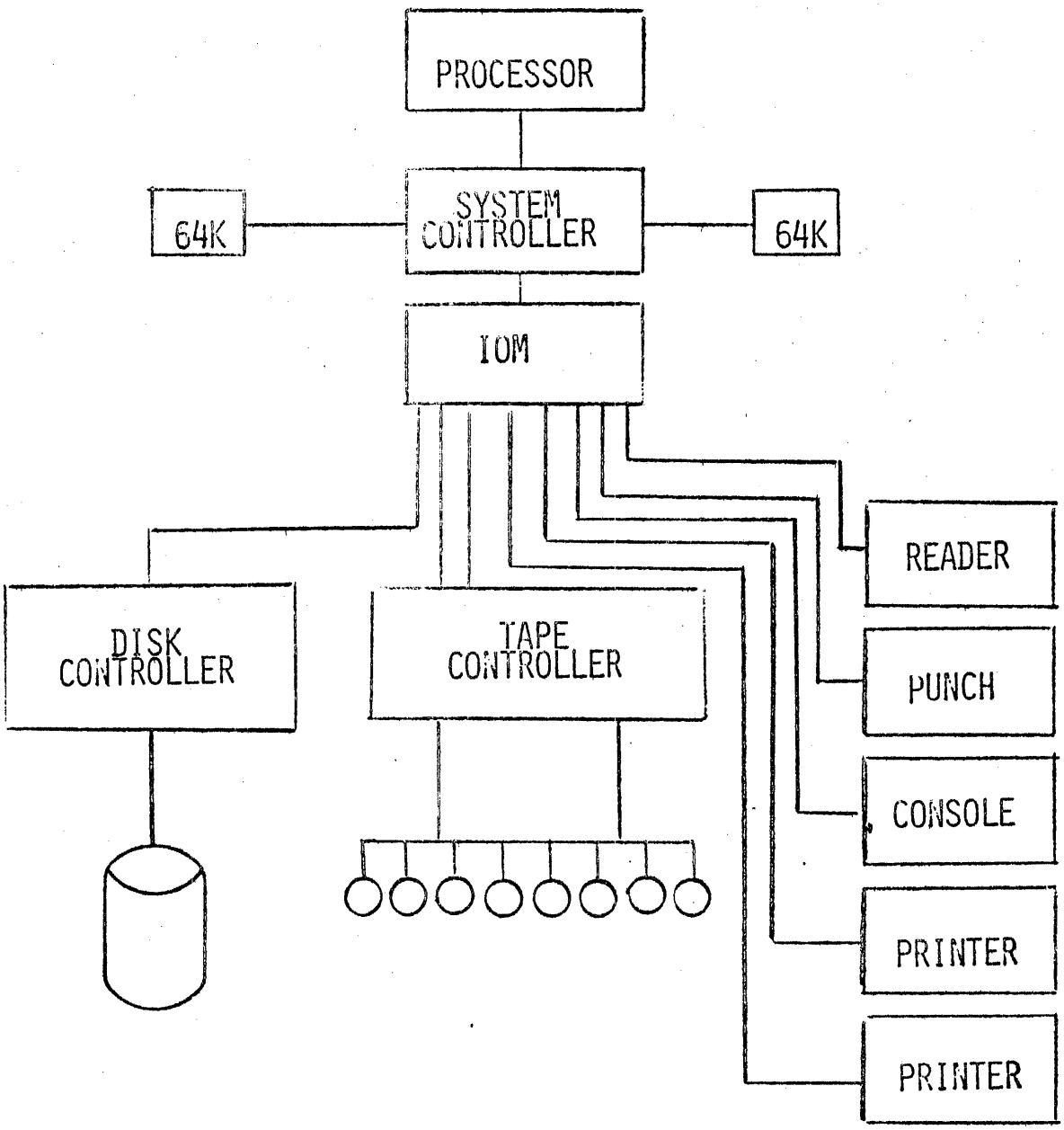


FUNCTIONAL MODULARITY



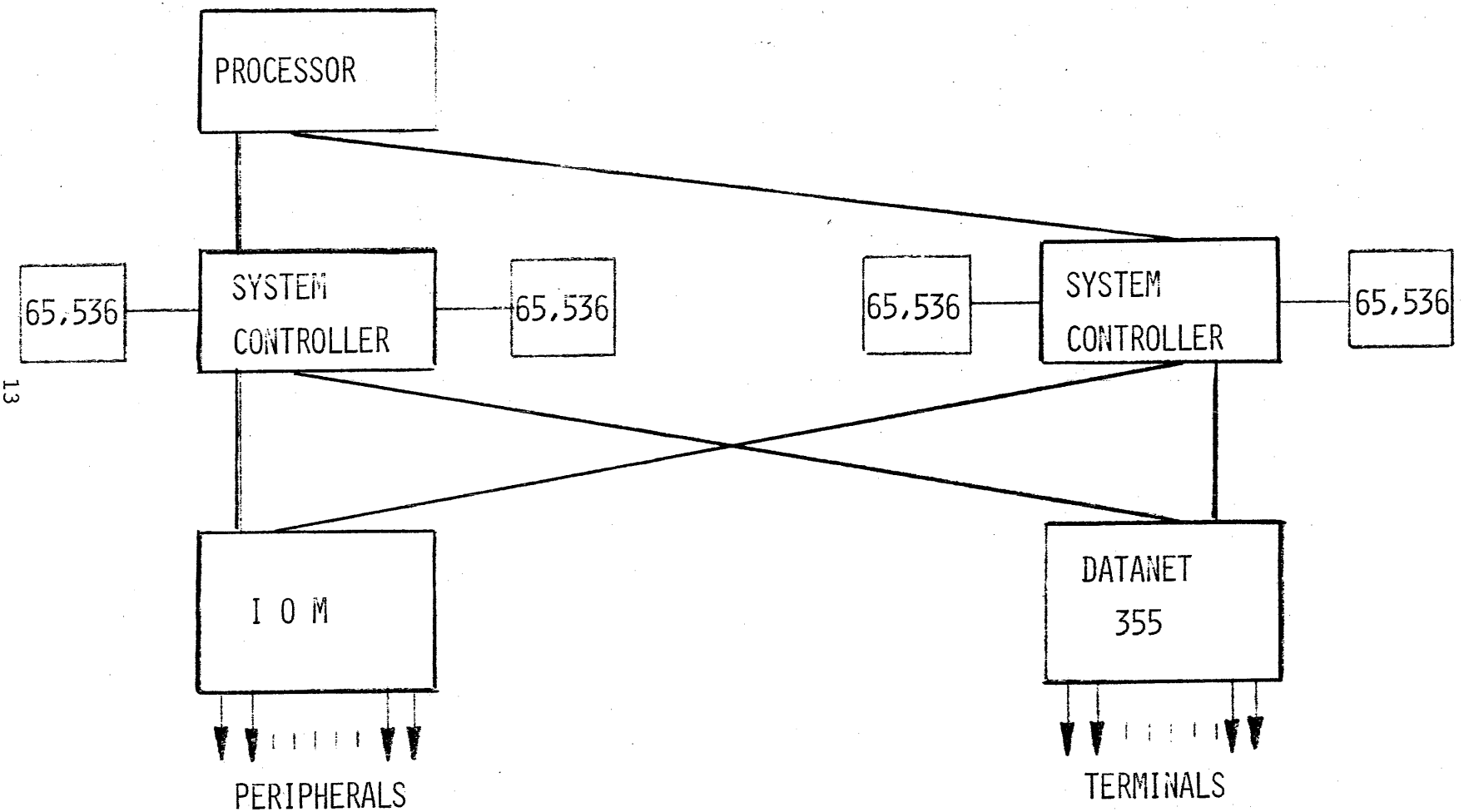
CONNECTIONS ARE MADE FROM EVERY MEMORY MODULE TO EVERY ACTIVE MODULE (IOM, PROCESSOR, OR COMMUNICATIONS PROCESSOR).

SERIES 6000

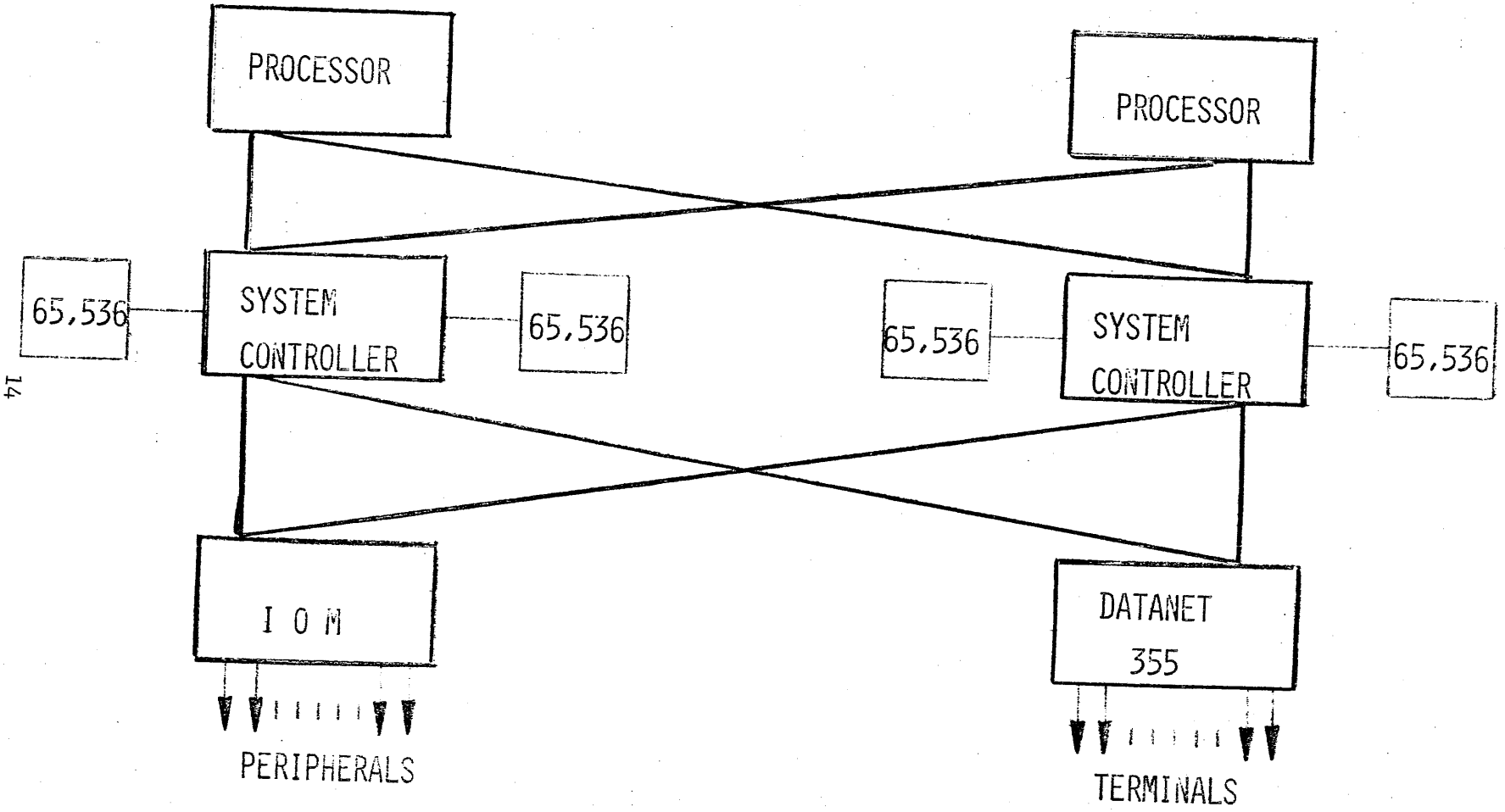


HARDWARE COMPONENTS

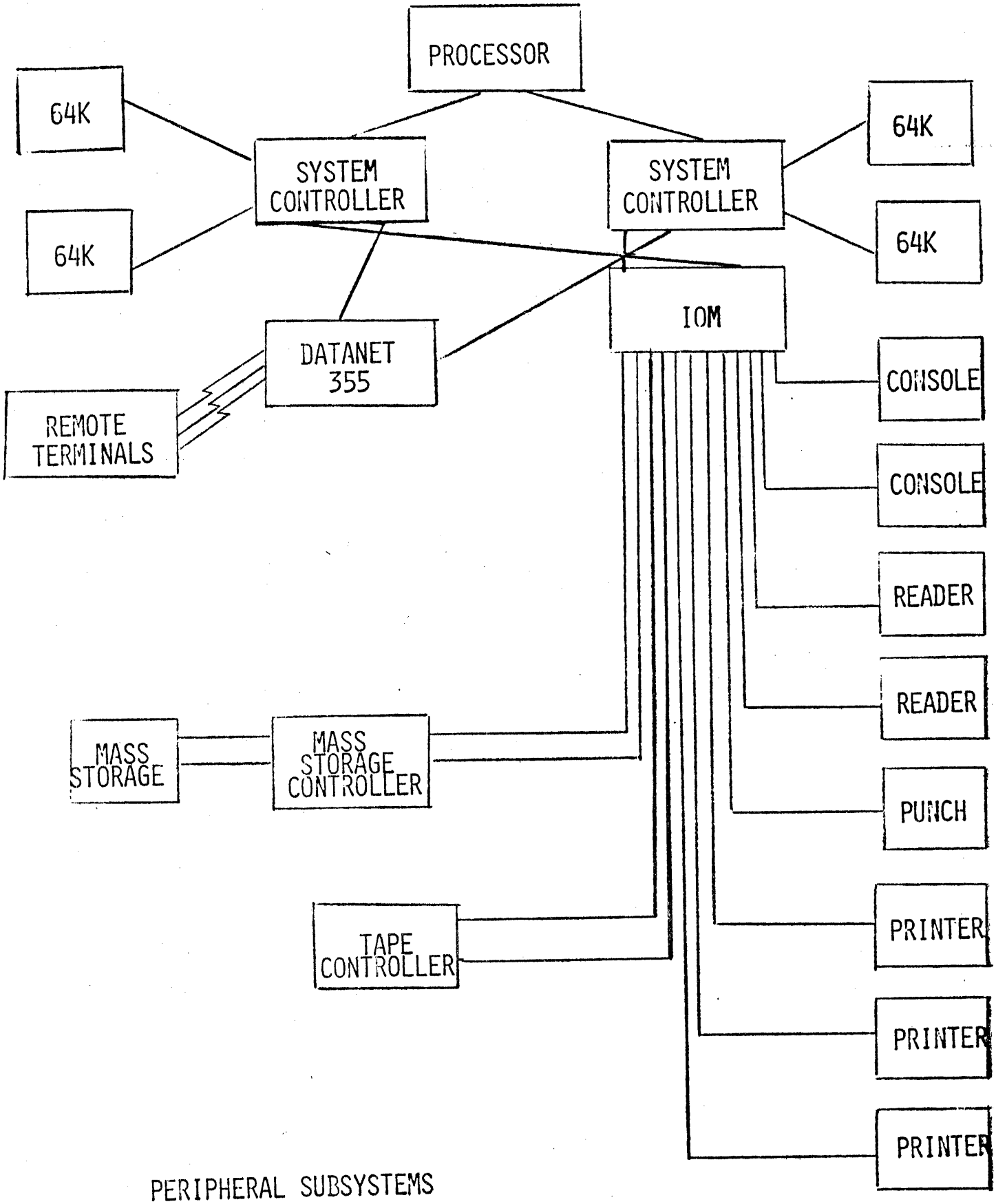
MULTIDIMENSIONAL SYSTEM



MULTIPROCESSOR SYSTEM



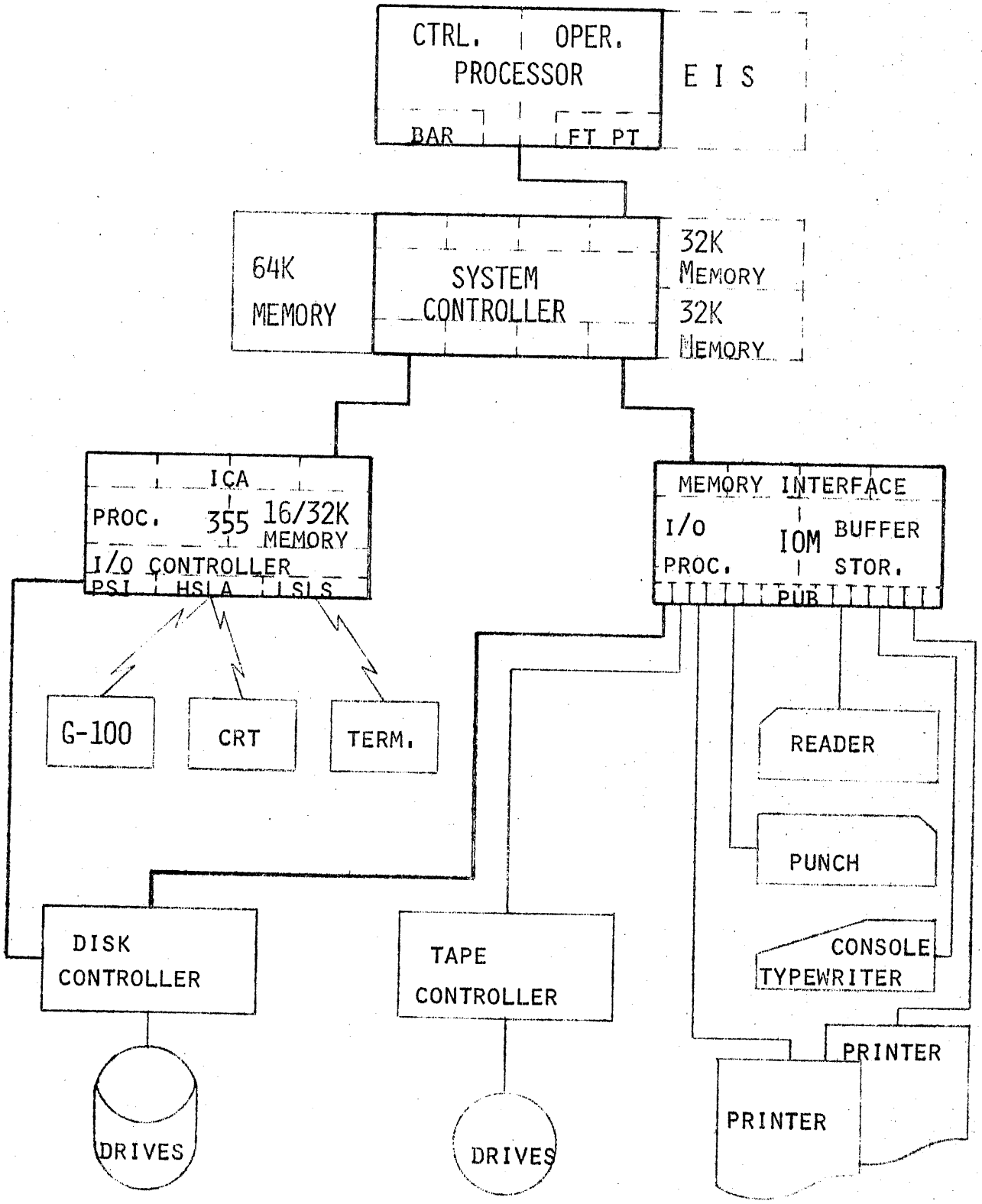
SERIES 6000

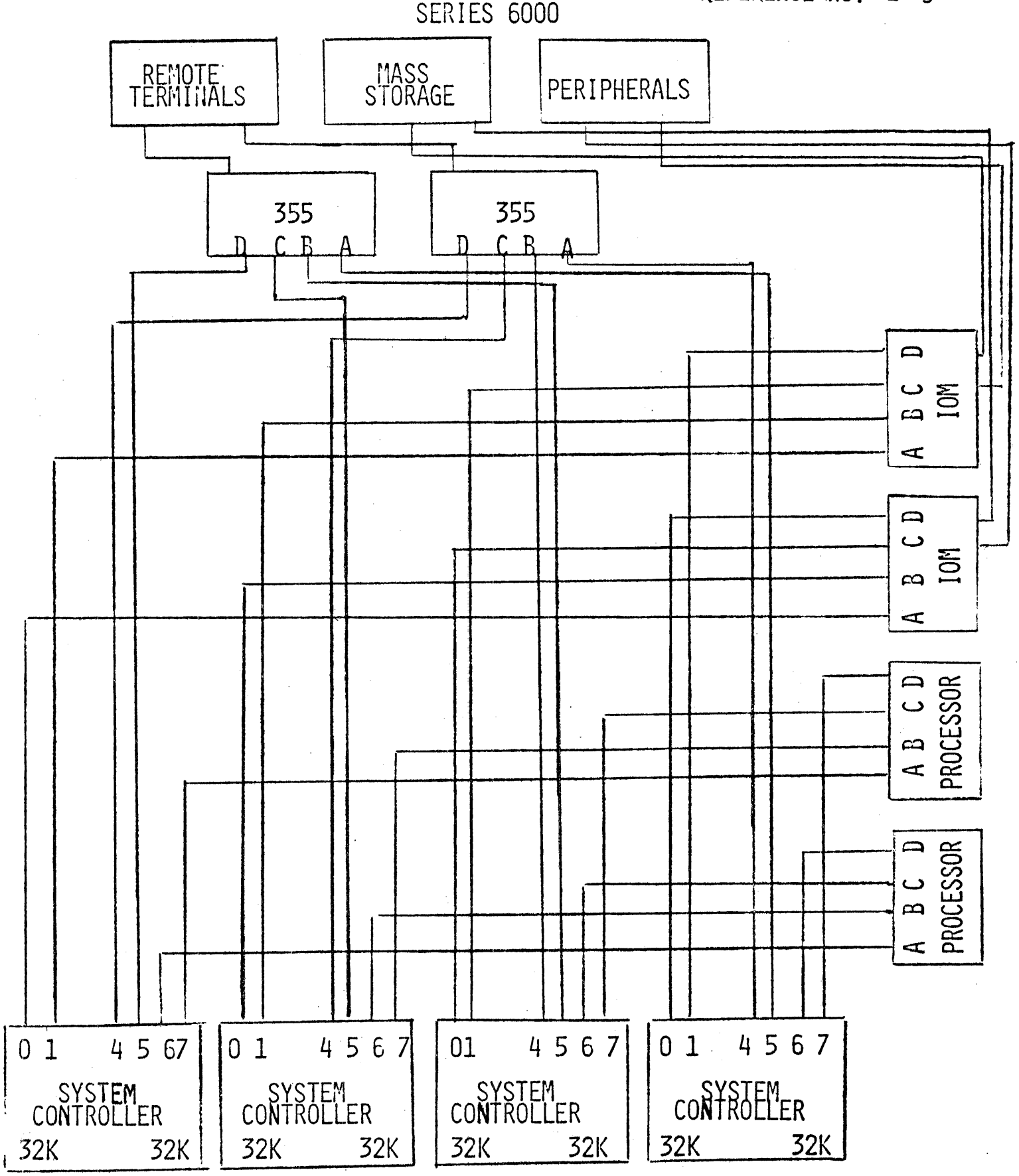


PERIPHERAL SUBSYSTEMS

H-6000 HARDWARE COMPONENTS

STUDENT HANDBOOK
REFERENCE NO. 2-8





MULTIPROCESSOR SYSTEM

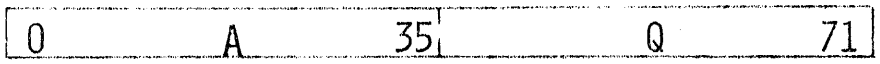
SERIES 6000 CHARACTERISTICS

	Model 6030/6040	Model 6050/6060	Model 6070/6080
Memory Size in words (36-bit word)	65,536 - 131,072	98,304 - 524,288	131,072 - 1,048,576
Cycle Time (microseconds per two words)	1.2	1.2	0.5
No. of Data Channels	16	24	24
Max. Transfer Rate Per IOM (chars/sec.)	1.3M	3.7M	6.0M
Peripheral Capacity (subsystems)	16	24	24
Interleaving	No	2&4-way	2&4-way
Instruction Overlapping	No	Yes	Yes
Instructions per Second (max.)	340,000	550,000	1,400,000

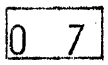
PROGRAM ACCESSIBLE REGISTERS

A REGISTER

Q REGISTER



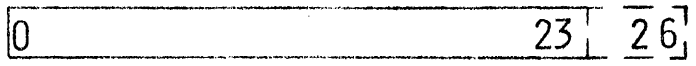
ACCUMULATOR



E REGISTER (FLOATING POINT EXPONENTS)

E/A REGISTER (FLOATING POINT)

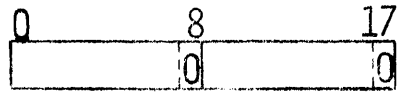
E/A/Q REG.



TIMER REGISTER

INSTRUCTION COUNTER

INDEX REGISTERS (0-7)

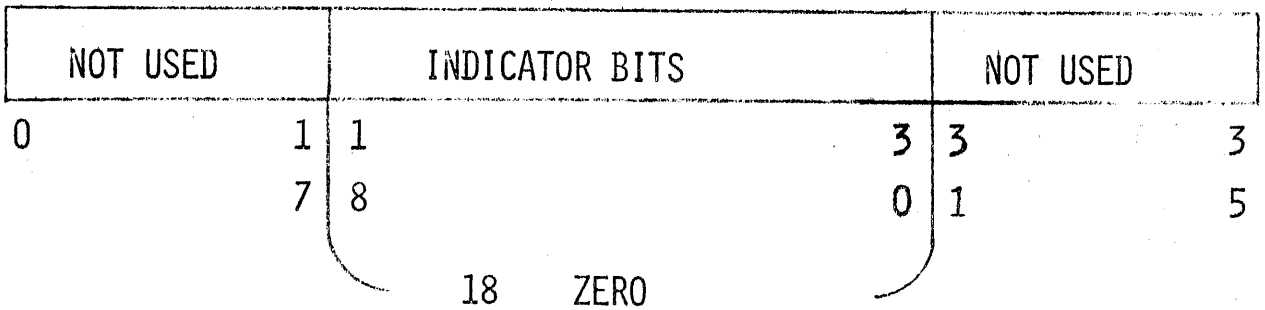


BAR

ADDR. REG.

INDICATOR REGISTER

STUDENT HANDBOOK
REFERENCE NO. 2-12



18	ZERO
19	NEGATIVE
20	CARRY
21	OVERFLOW
22	EXP. OVERFLOW
23	EXP. UNDERFLOW
24	OVERFLOW MASK
25	TALLY RUNOUT
26	PARITY ERROR
27	PARITY MASK
28	MASTER MODE
29	TRUNCATION (EIS)
30	MULTIWORD INSTRUCTION (EIS) INTERRUPT

H-6000 SOFTWARE SYSTEM MAJOR COMPONENTS

OPERATING SYSTEM (GCOS)

OTHER SYSTEM SOFTWARE

LANGUAGE PROCESSORS

TIME-SHARING SYSTEM

APPLICATIONS SOFTWARE

SYSTEM SOFTWARE

OPERATING SYSTEM - GCOS

GENERAL COMPREHENSIVE OPERATING SUPERVISOR

GENERAL CAPABILITIES

MULTIDIMENSIONAL MODES OF PROCESSING

LOCAL BATCH PROCESSING

REMOTE BATCH PROCESSING

REMOTE ACCESS PROCESSING

TIME-SHARING

TRANSACTION PROCESSING

MULTIPROGRAMMING

MULTIPROCESSING

CENTRAL FILE SYSTEM

FUNCTIONAL CHARACTERISTICS

RESOURCE MANAGEMENT

JOB SCHEDULING

JOB PRIORITY ALLOCATION

DATA BASE MANAGEMENT

PROGRAM AND DATA FILE SECURITY CONTROL

ON-LINE SYSTEM DEVELOPMENT

EASE OF USE

SYSTEM SOFTWARE

GENERAL LOADER

FILE AND RECORD CONTROL (GFRC)

UTILITY PACKAGE

SYSTEM EDITOR

BULK MEDIA CONVERSION (BMC)

SYSTEM MEDIA CONVERSION

- INPUT MEDIA CONVERSION (GIN)
- OUTPUT MEDIA CONVERSION (SYSOUT)

LANGUAGE PROCESSORS

MACRO ASSEMBLER (GMAP)

FORTRAN COMPILER

COBOL COMPILER

ALGOL COMPILER

JOVIAL COMPILER

MISC. PROCESSORS

- SORT/MERGE
- IDS
- ISP

TIME-SHARING SYSTEM

PROGRAMMING LANGUAGES
- BASIC
- FORTRAN
- ABACUS

REMOTE JOB ENTRY
- CARDIN
- JOUT
- SCAN
- RBUG

FILE EDITING, INSPECTION AND MAINTENANCE
- EDITOR
- RUNOFF
- DATA BASIC
- ACCESS
- IDS DATA QUERY

TIME-SHARING SYSTEM EXTENSION & MAINTENANCE
- LODX SUBSYSTEM
  - TDS
  - SABT
- MASTER

TIME-SHARING LIBRARY
- APPLICATION PROGRAMS
- FORTRAN LIBRARY GENERATOR AND EDITOR
  - TSLG
  - LIBED

APPLICATION SOFTWARE

LP6000 - Linear Programming System

APT - Automatic Programmed Tools

PERT/TIME - Program Evaluation Review Technique/Time

PERT/COST - Program Evaluation Review Technique/Cost

SIMSCRIPT - Simulation Language

BMD - Biomedical Statistical Programs

MATHPAC - Mathematical/Engineering Programs

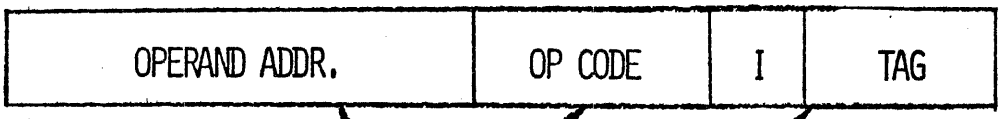
Series 6000 Parts Explosion System (GEPEXS)

Series 6000 Inventory Management System (GEIMS)

CEP - Civil Engineering Package

Series 6000 Time Series Forecasting (GECAST)

CODING FORM



LOCATION 1 6	E/O 7	OPERATION 8 15	ADDR/MOD 16	COMMENTS 32	ID 73 80
GO	E	LDA	WORD,7	THIS IS A SAMPLE	0010
6 CHAR. MAX. (A-Z) (0-9) (.) 1ST CHAR. MUST NOT BE ZERO AND MUST HAVE AT LEAST ONE NON-NUMERIC CHAR.	E Ø 8	INST. MNEMONICS, PSEUDO OPS, MACRO NAMES. NO SPECIAL CHARS. OR BLANKS PERMITTED	VARIABLE LENGTH FIELD. SUBFIELDS SEPARATED BY COMMAS. ABSOLUTE, SYMBOLIC, OR RELATIAVE ADDRESSING. ALGEBRAIC-BOOLEAN EXPRESSIONS. OPERATORS, * + - / ALSO =	← OPTIONAL →	

GMAP COMPILE

(SINGLE GMAP ASSEMBLY)

1	8	16
$	SNUMB	34871
$	IDENT	1234,FIELDTRNG.....
$	GMAP	NDECK,COMDK

(SOURCE DECK)

.

.

.

$

ENDJOB

***EOF

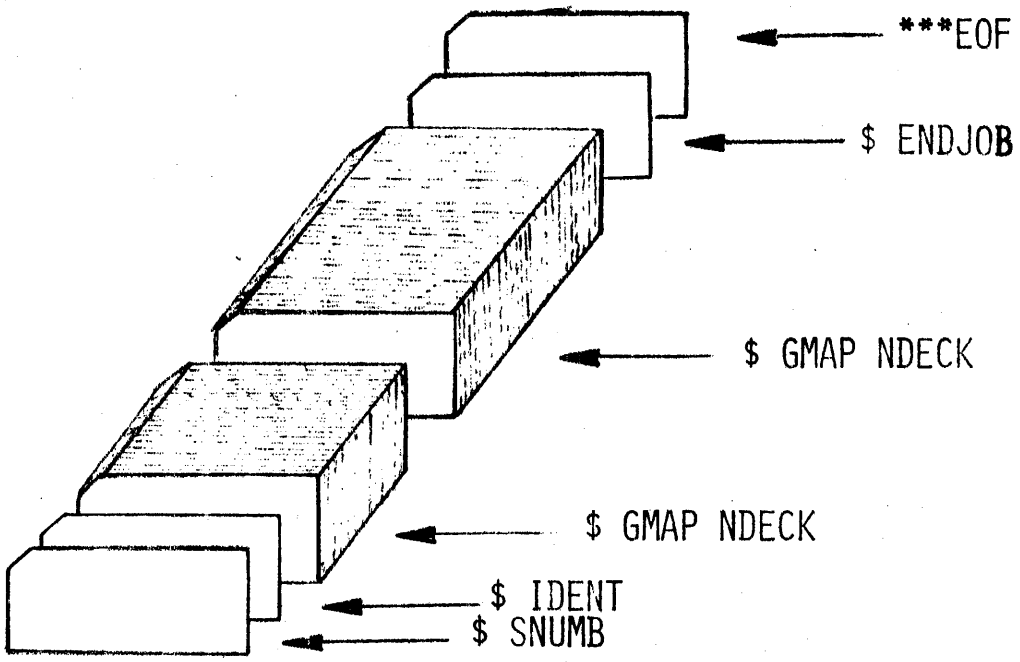
GMAP COMPILE

(MULTIPLE GMAP ASSEMBLIES)

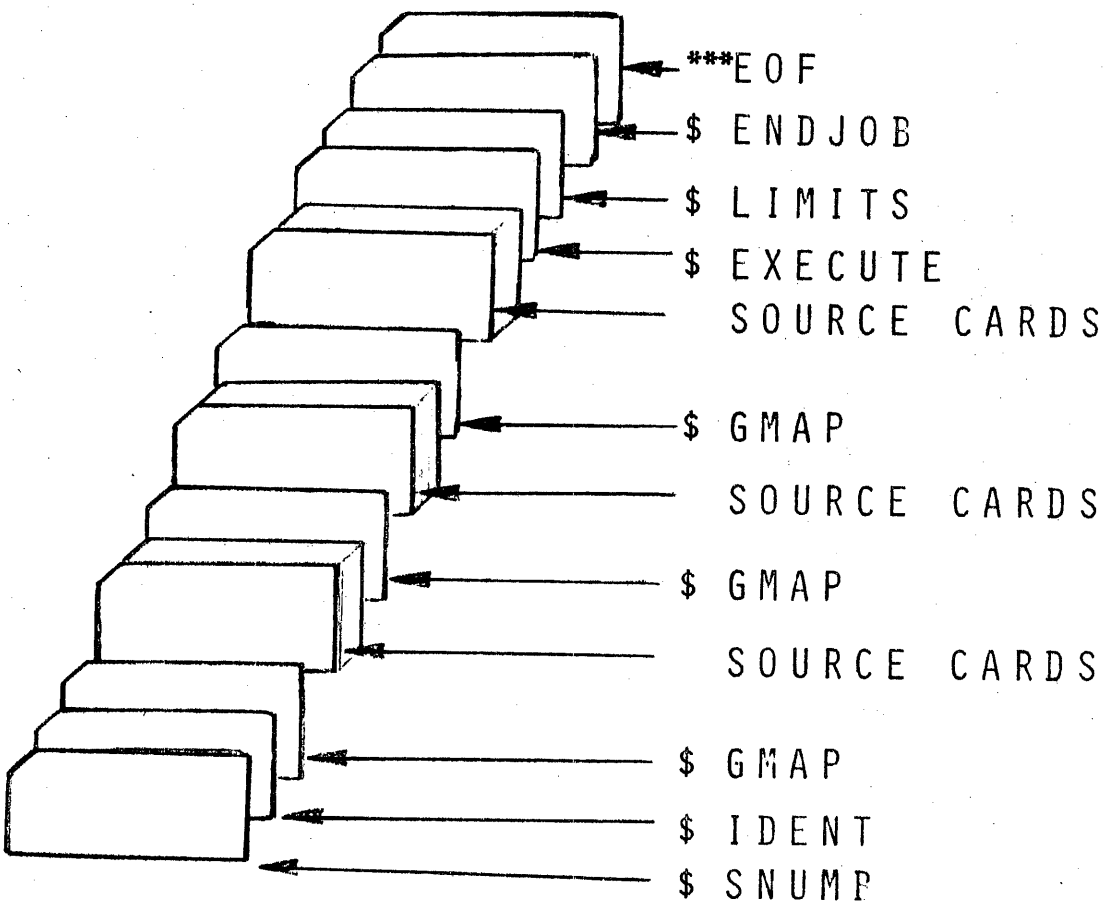
```
1               8                16
-----------------------------------------------
$               SNUMB            34872
$               IDENT            1234,FIELDTRNG,....
$               GMAP
        (SOURCE DECK)
$               GMAP
        (SOURCE DECK)
$               GMAP
        (SOURCE DECK)
$               GMAP
        (SOURCE DECK)
$               ENDJOB
***EOF
```

GMAP COMPILE

CARD DECK ARRANGEMENT



MULTIPLE GMAP ASSEMBLY JOB



GMAP CONTROL RECORDS

(COMPILE AND EXECUTE)

```
1               8                16
-----------------------------------------------------------
$               SNUMB            34873
$               IDENT            1234,FIELDTRNG,....

$               GMAP             NDECK,COMDK
              (SOURCE DECK)
$               EXECUTE
$               LIMITS           3,4K,,1000
$               SYSOUT           AL      (PRINTER OUTPUT)
$               DATA             IN
            (DATA FOR EXECUTION)
$               ENDJOB
***EOF
```

EXECUTING OBJECT PROGRAMS

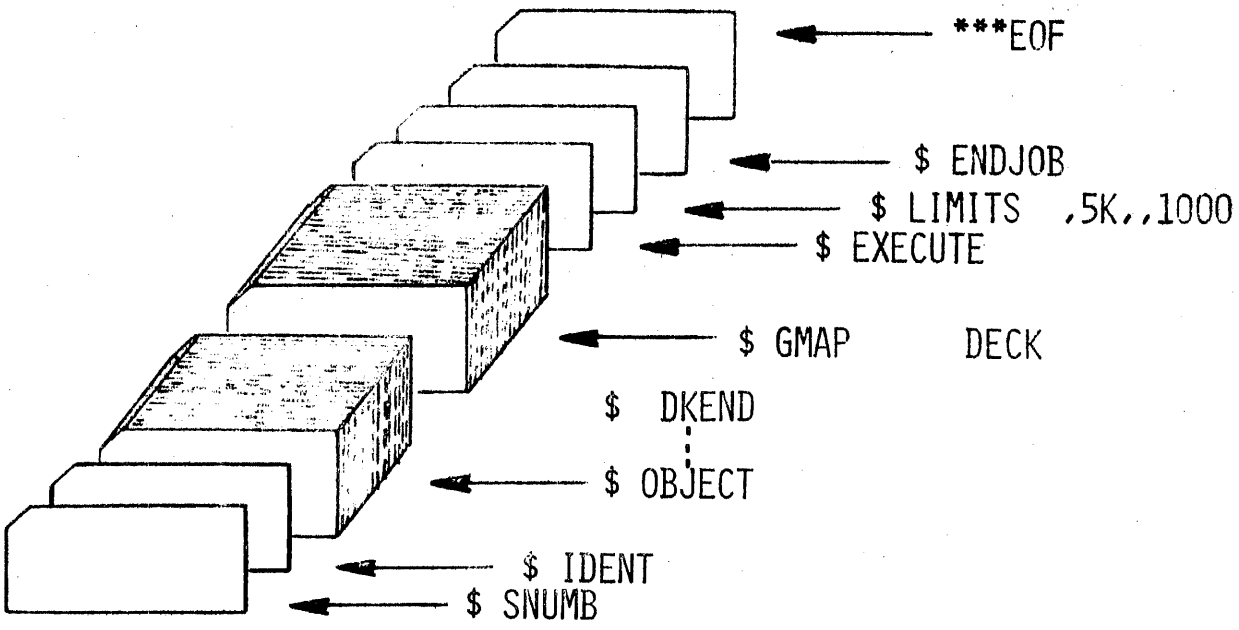
```
1               8                   16
-----------------------------------------------------------
$               SNUMB               34874
$               IDENT               1234,FIELDTRNG,....
$               OBJECT
            (BINARY DECK)
$               DKEND
$               OBJECT
            (BINARY DECK)
$               DKEND
$               EXECUTE             DUMP
$               LIMITS              4,6K,,2000
$               SYSOUT              X2      (PRINTER OUTPUT)
$               DATA                IN
        (DATA FOR OBJECT PROGRAM)
$               ENDJOB
***EOF
```

COMPILE AND EXECUTE

WITH OBJECT DECKS

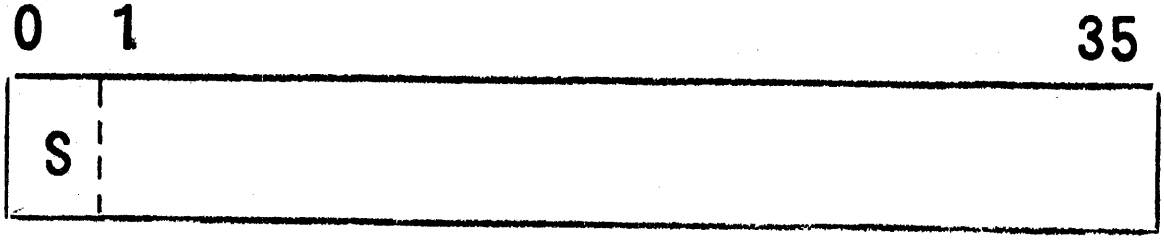
```
1               8               16
$               SNUMB           34875
$               IDENT           1234,FIELDTRNG,...
$               GMAP
          (SOURCE DECK)
$               OBJECT
          (OBJECT DECK)
$               DKEND
$               EXECUTE
$               LIMITS          3,5K,,2500
$               SYSOUT          B3          (PUNCH OUTPUT)
$               DATA
          (DATA FOR OBJECT PROGRAM)
$               ENDJOB
***EOF
```

COMPILE AND EXECUTE WITH OBJECT DECK

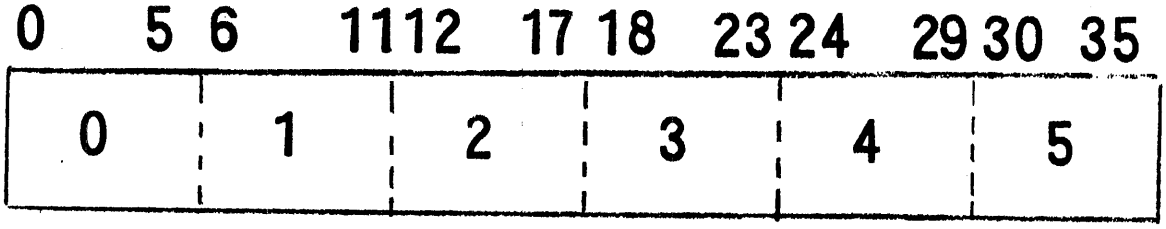


WORD FORMATS

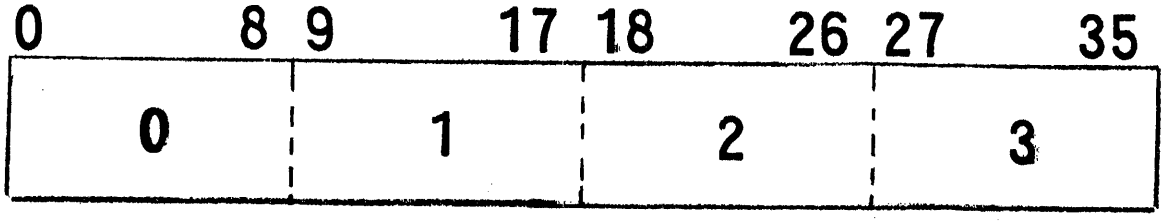
BINARY



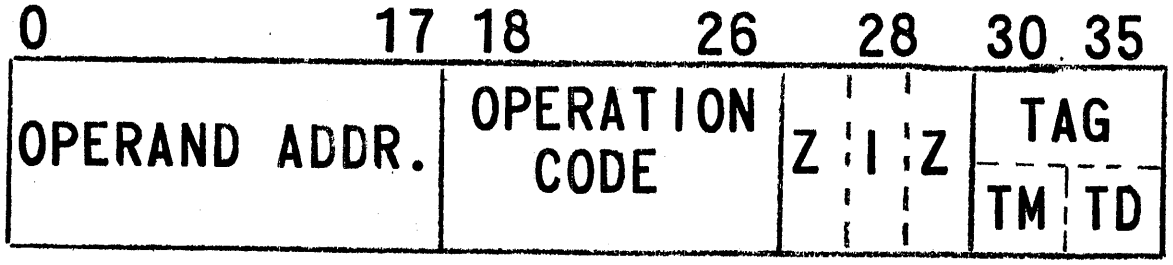
BCD



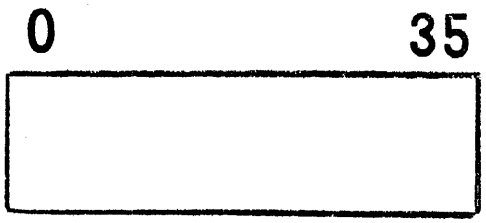
9-BIT BYTE



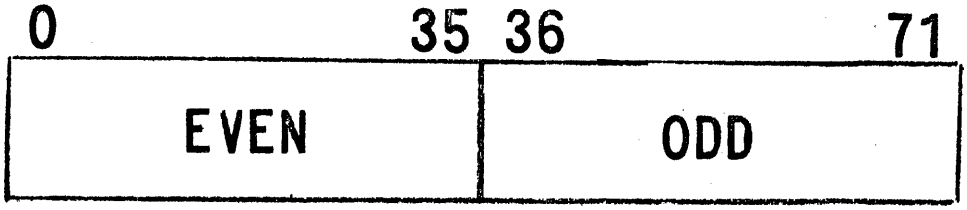
INSTRUCTION



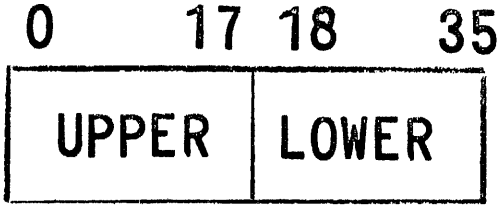
SINGLE/DOUBLE PRECISION



SINGLE



DOUBLE OR Y-PAIR



HALF

INSTRUCTION PRECISION

PRECISION	REPRESENTATION
HALF-WORD	UPPER HALF
	LOWER HALF
SINGLE WORD	
DOUBLE WORD	

FUNCTIONAL GROUPS OF PSEUDO OPERATIONS

CONTROL

LOCATION COUNTER

SYMBOL DEFINING

DATA GENERATING

STORAGE ALLOCATION

SPECIAL

MACRO

CONDITIONAL

PROGRAM LINKAGE

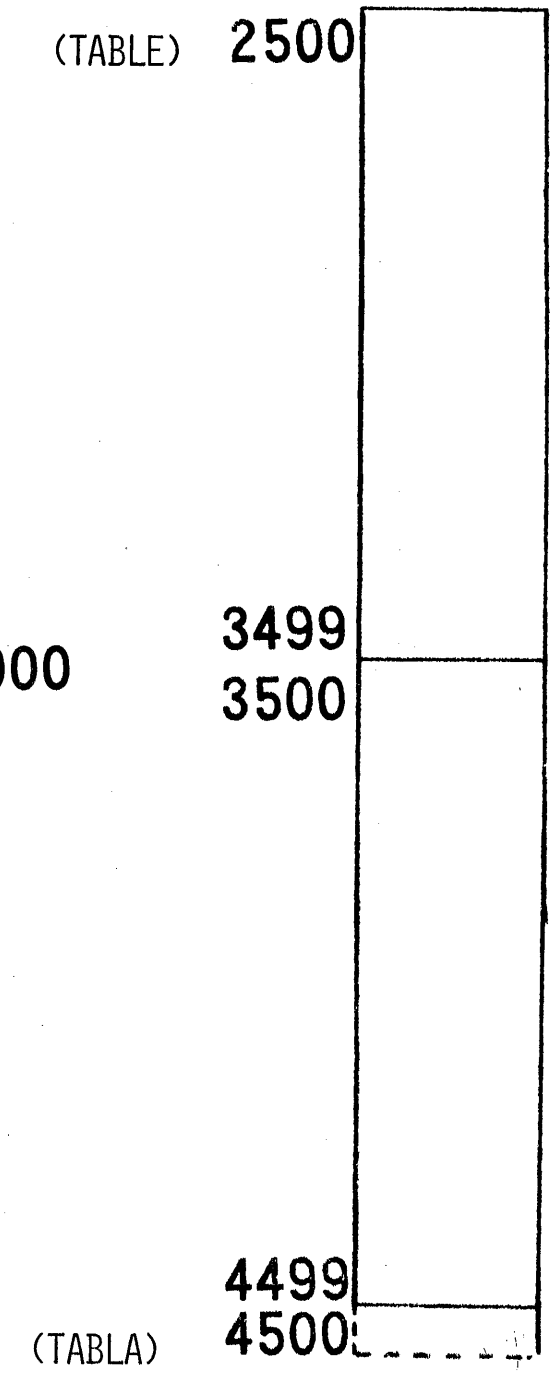
ADDRESS TALLY

MISCELLANEOUS

STORAGE ALLOCATION

PSEUDO-OPERATIONS

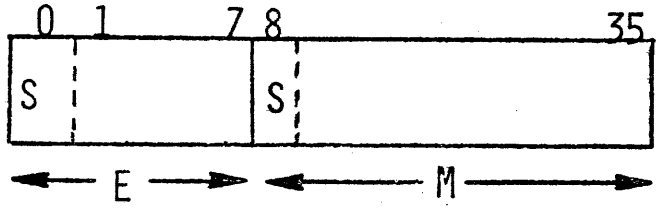
```
1       8     16
TABLE   BSS   1000

TABLA   BFS   1000

        STA   TABLE

        STA   TABLA-1000

        STAQ  TABLE

        STAQ  TABLA-2

*       LDAQ  TABLE+5
        STAQ  TABLA-2
```



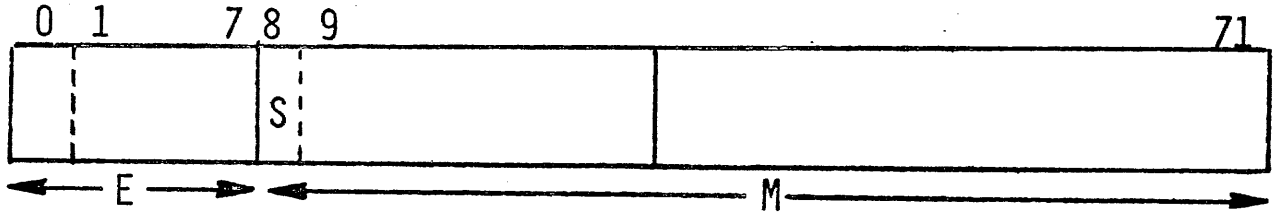
* Addressing Error

BINARY FLOATING POINT

SINGLE - WORD PRECISION



DOUBLE - WORD PRECISION



LOWEST PERMISSIBLE EXPONENT = -128

LARGEST POSSIBLE EXPONENT = +127

NUMBER CONVERSION TECHNIQUE

1. DECIMAL 11.054_{10}

2. OCTAL INTEGER CONVERSION

$8 \mid 11 \quad 3$

$8 \mid 1 \quad 1$

0

$13.$

3. OCTAL FRACTION CONVERSION

$.054$

$\times 8$

0.432

$\times 8$

3.456

$\times 8$

3.648

$\times 8$

5.184

$\times 8$

1.472

$\times 8$

4.576

$.033514_8$

4. OCTAL $+13.033514_8$

5. BINARY CONVERSION $001011.000011011101001100_2$

6. FLOATING POINT

S	EXPONENT	S	MANTISSA
0	0000100	0	1011.000011011101001100$_2$

MOVE FOUR PLACES LEFT

7. FLOATING POINT (OCTAL)

010541564600_8

NUMBER CONVERSION TECHNIQUE

8. OCTAL INTEGER CONVERSION

13_8
$\times 8$

8
+3

$(11)_{10}$

10. DECIMAL (BCD)

11.054_{10}

9. OCTAL FRACTION CONVERSION

033514_8
$\times 12$

067230
033514

$(0)424370$
$\times 12_8$

1050760
424370

$(5)314660$
$\times 12_8$

631540
314660

$(4)000340$

0 5 4

1. DATA MOVEMENT

   LOAD STORE SHIFT

2. FIXED POINT ARITHMETIC

   ADD-SUBTRACT-MULTIPLY-DIVIDE

3. LOGICAL

   BOOLEAN (AND-OR-EXOR), COMPARE

4. TRANSFER OF CONTROL

   CONDITIONAL & UNCONDITIONAL

5. FLOATING POINT

   F. P. DATA MOVEMENT
   F. P. ARITHMETIC
   F. P. COMPARES

6. MASTER MODE

   CONNECT I/O, M. C. CMNDS., DIS

7. MISCELLANEOUS

   REPEATS, BCD, EFFECTIVE ADDRESS

LOAD/STORE

LDA	STA	LCA
LDQ	STQ	LCQ
LDAQ	STAQ	LCAQ
LDXn	STXn	LCXn
LXLn	SXLn	
ZERO	NONE	ZERO
NEG		NEG
		O'FLO

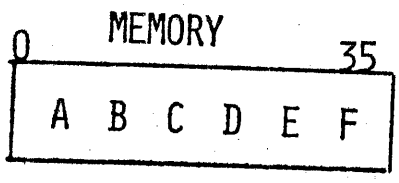
LOAD/STORE
SAMPLE CODING

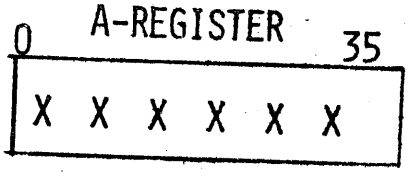
```
(1)          (8)          (16)
                  LDA      IN+2
R1                STA      B
                   ≀
R4                LDQ      A
                  LDA      =6HbFLDbA
                  STAQ     P1
                  STQ      P2
                  LDX7     R1
                  STX7     R4
                  LCAQ     =5DOB71
                  STAQ     C1
                   ≀
                   ≀
C1                EDEC     0,0
P1                BCI      2,
P2                DEC      0
A                 BSS      1
B                 OCT      -777777777777
```

LDAC INSTRUCTION

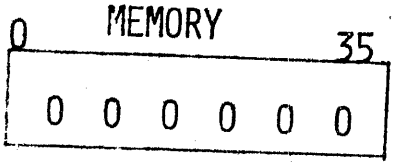
(LDAC MEMORY)

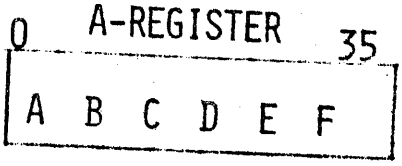
BEFORE EXECUTION:





AFTER EXECUTION:





ADDRESS MODIFICATION

INSTRUCTION FORMAT:

ADDRESS FIELD	OP CODE		TM	TD

TM - TYPE OF MODIFICATION

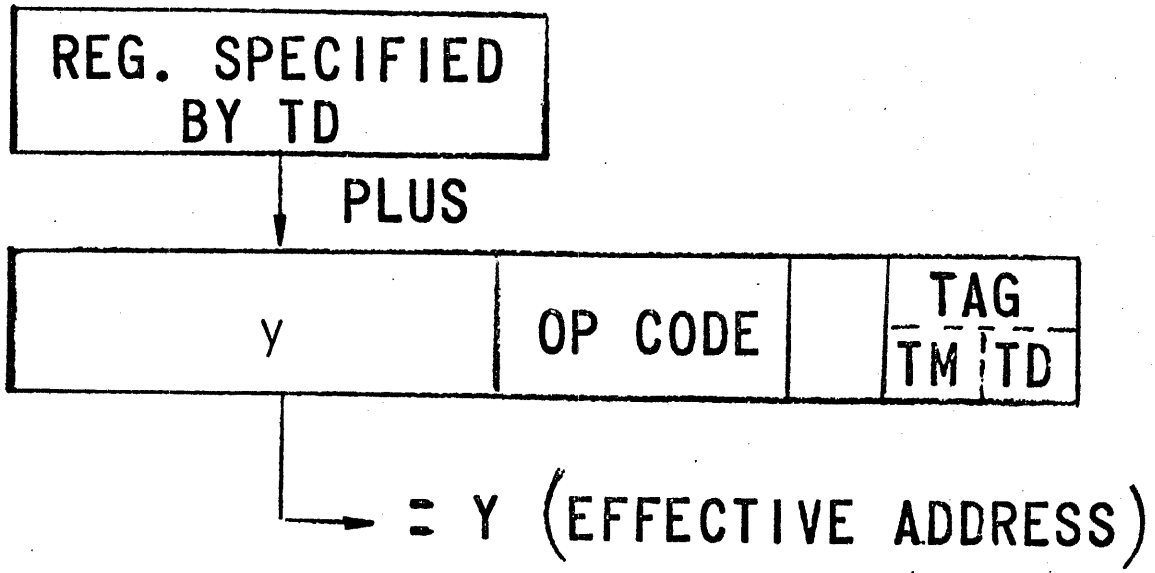
00 - REGISTER R
01 - REGISTER THEN INDIRECT RI
11 - INDIRECT THEN REGISTER IR
10 - INDIRECT THEN TALLY IT

TD - REGISTER OR TALLY DESIGNATOR

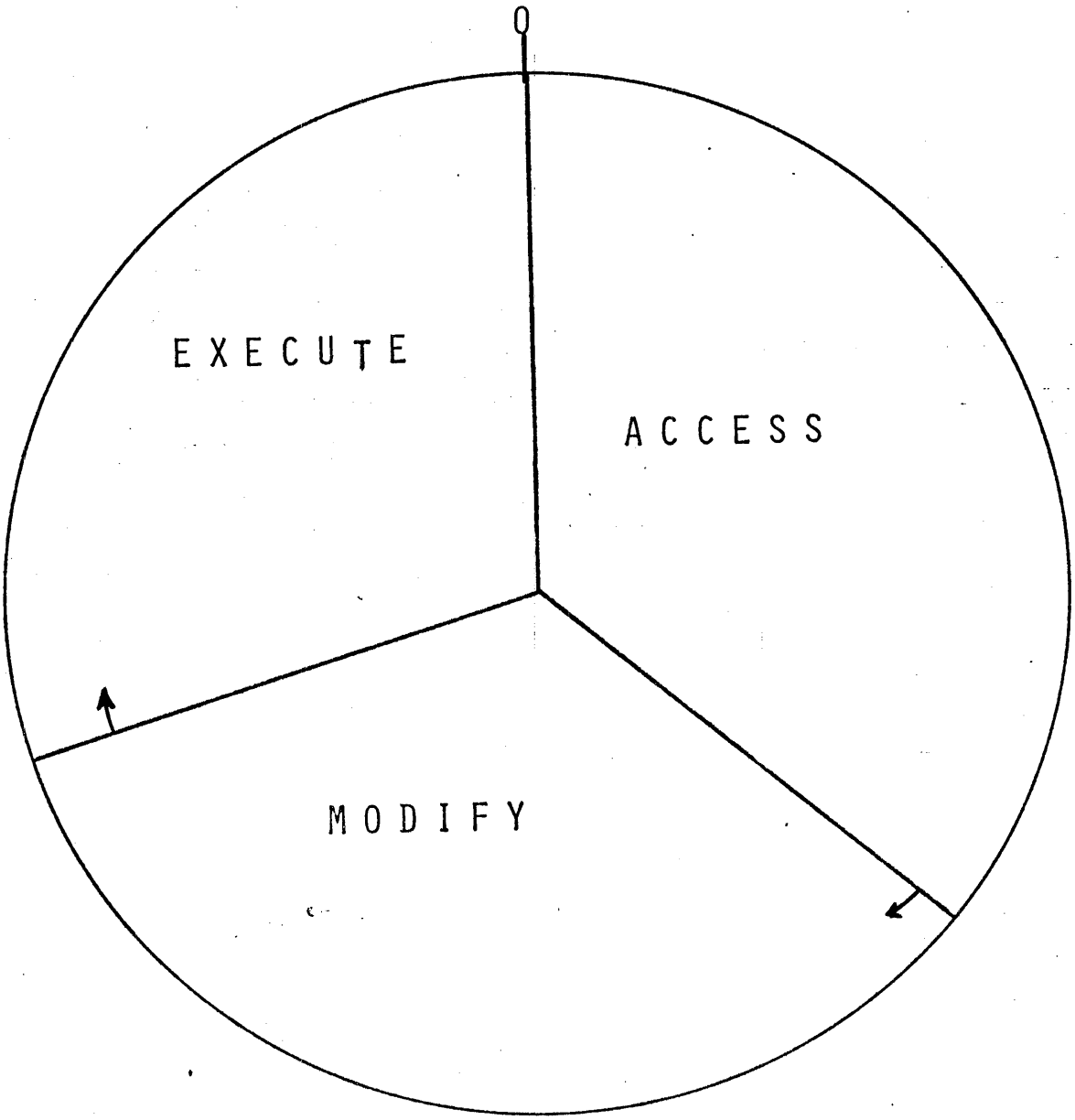
R
IR
RI
} REGISTER DESIGNATOR

IT - TALLY DESIGNATOR

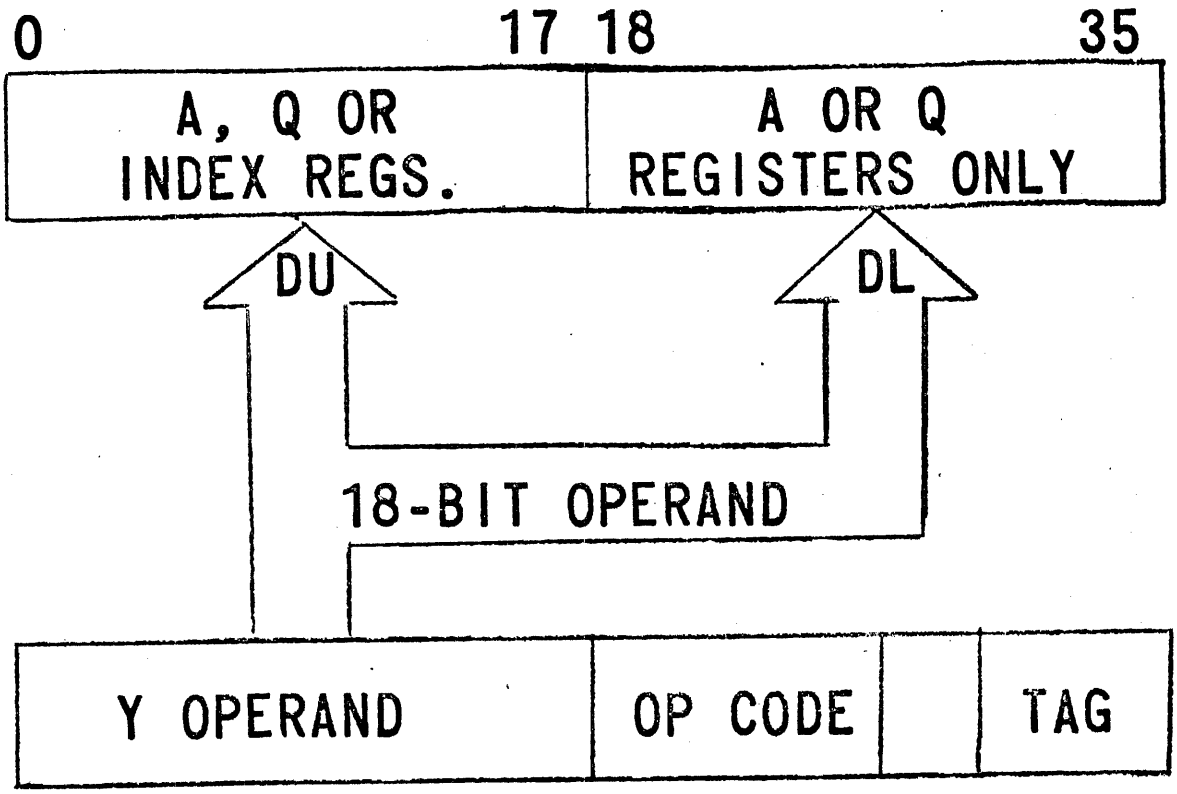
INDEXING



ALGORITHM			CODING
Xn Y =	y+C(Xn)	0-17	Y,0-7
AU Y =	y+C(A)	0-17	Y,AU
AL Y =	y+C(A)	18-35	Y,AL
QU Y =	y+C(Q)	0-17	Y,QU
QL Y =	y+C(Q)	18-35	Y,QL
IC Y =	y+C(IC)	0-17	Y,IC

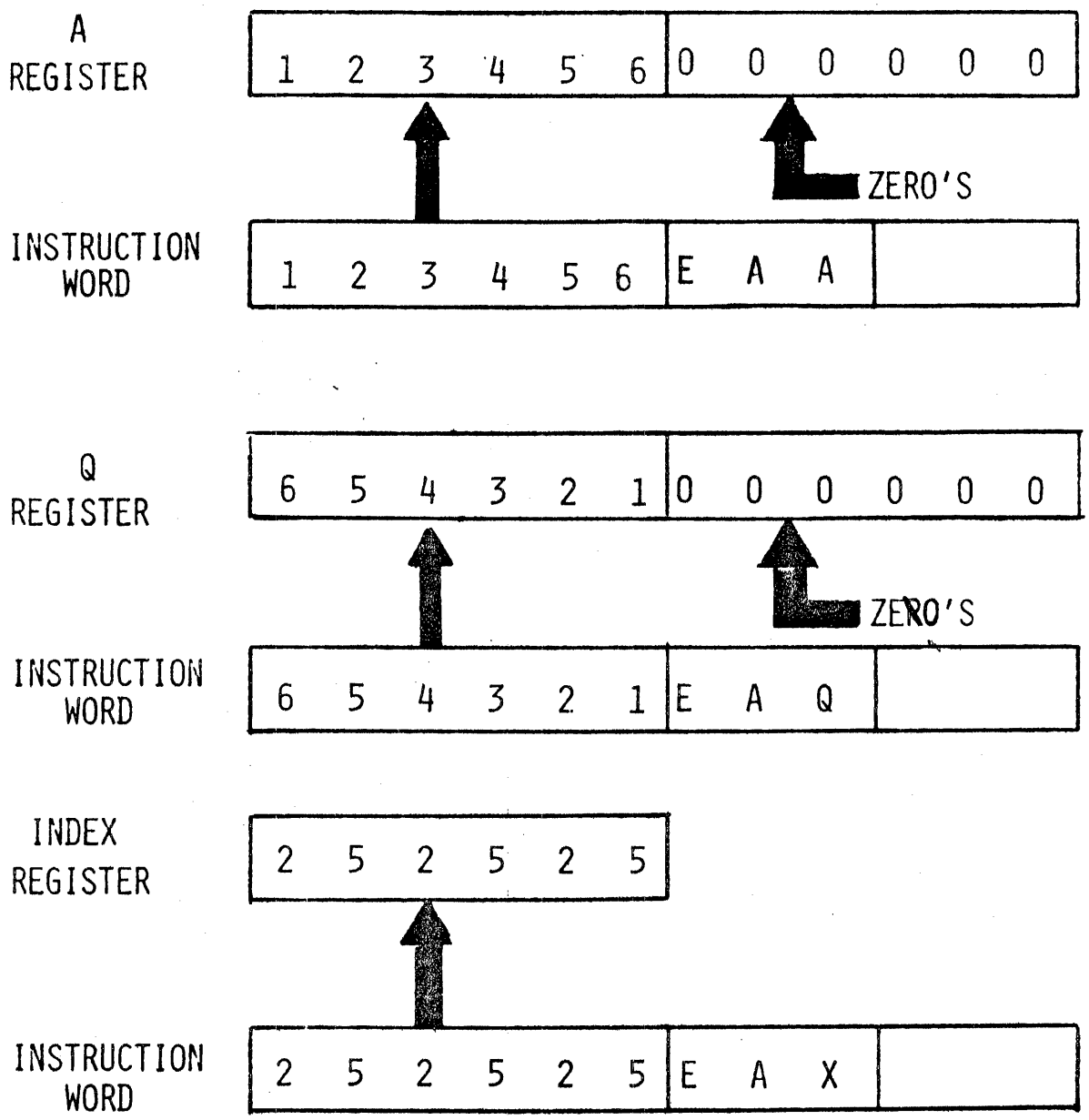

0
EXECUTE
ACCESS
MODIFY

DIRECT OPERAND



ALGORITHM	CODING
DU Y⇒C(R) 0-17	Y,DU
DL Y⇒C(R) 18-35	Y,DL

EFFECTIVE ADDRESS



LITERALS AND DATA GENERATING PSEUDO OPS

LITERALS	PSEUDO OPS
DECIMAL	DEC
OCTAL (=O)	OCT
HOLLERITH (=H)	BCI ASCII UASCI
INSTRUCTION (=M)	ARG
VARIABLE FIELD (=V)	VFD

ALL LITERALS MUST BE PRECEDED WITH AN EQUAL SIGN IN COLUMN 16 OF THE VARIABLE FIELD OF THE CODING SHEET.

DECIMAL LITERALS

INTEGERS:

DISTINGUISHED BY THE ABSENCE OF A DECIMAL POINT, THE LETTER B, THE LETTER,E, OR THE LETTER D.

SINGLE PRECISION FLOATING POINT:

DISTINGUISHED BY THE PRESENCE OF A DECIMAL POINT, THE LETTER E, OR BOTH.

DOUBLE PRECISION FLOATING POINT:

DISTINGUISHED BY THE PRESENCE OF THE LETTER D.

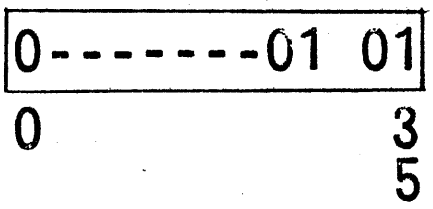
FIXED POINT:

DISTINGUISHED BY THE PRESENCE OF THE LETTER B. (OVERRIDES E OR D)

DECIMAL VALUES

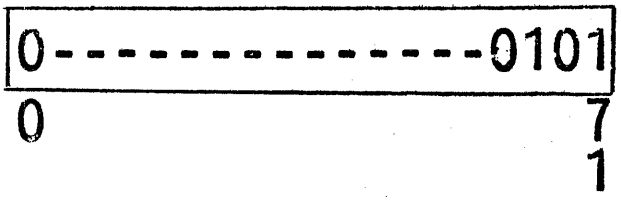
SINGLE PRECISION

```
        DEC   5
        LDA   =5
        DEC   5B35
        LDA   =5B35
        DEC   5E0B35
```



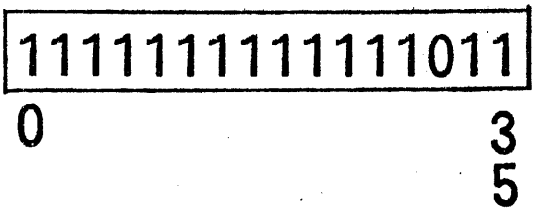
DOUBLE PRECISION

```
        DEC   0,5
        DEC   5D0B71
        LDAQ  =5D0B71
```



SIGNED VALUES

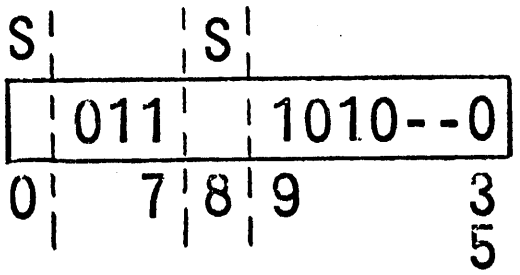
```
        DEC   -5
        LDA   =-5
```



FLOATING POINT VALUES

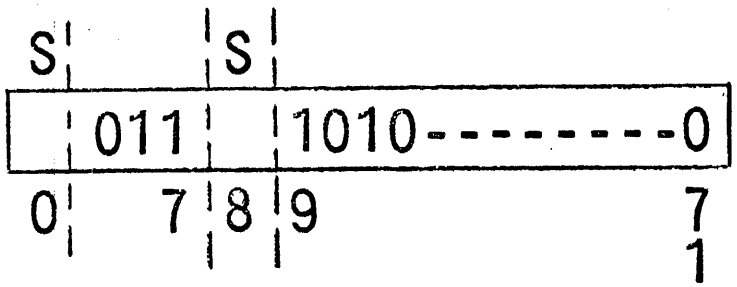
SINGLE PRECISION

```
DEC     5.
LDA     =5.
DEC     5EO
LDA     =5EO
```



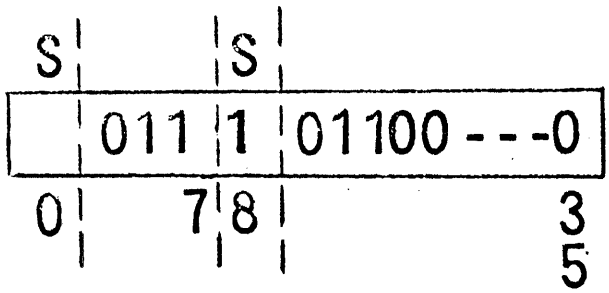
DOUBLE PRECISION

```
DEC     5DO
LDAQ    =5DO
```



NEGATIVE VALUES

```
DEC     -5.
DEC     -5EO
LDA     =-5.
```



OCTAL LITERALS

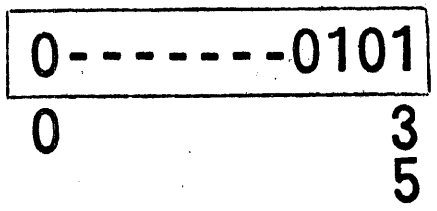
THE OCTAL LITERAL CONSISTS OF THE CHARACTER Ø, FOLLOWED BY A SIGNED OR UNSIGNED OCTAL INTEGER WHICH MAY BE FROM ONE (1) TO TWELVE (12) DIGITS IN LENGTH PLUS A SIGN.

THE ASSEMBLER WILL STORE THE INTEGER IN A WORD RIGHT-JUSTIFIED. THE WORD WILL BE STORED IN ITS REAL FORM AND WILL NOT BE COMPLEMENTED IF THERE IS A PRESENCE OF A MINUS SIGN (-).

OCTAL VALUES

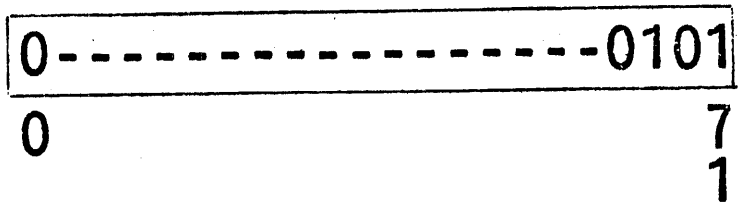
SINGLE PRECISION

```
OCT     5
LDA     =Ø5
```



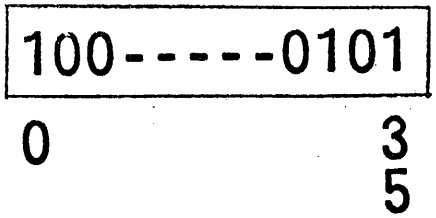
DOUBLE PRECISION

```
OCT     ,5
```



SIGNED VALUES

```
OCT     -5
```

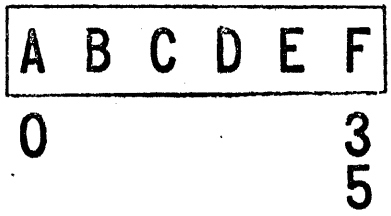



```
LDA     =Ø-5
```

HOLLERITH DATA

SINGLE PRECISION

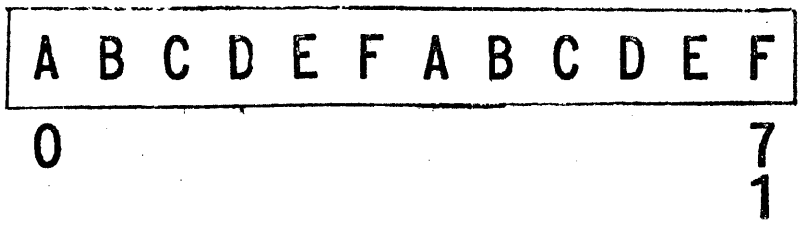
```
BCI     1,ABCDEF
```




```
LDA     =6HABCDEF
```

DOUBLE PRECISION

```
BCI     2,ABCDEFABCDEF
LDAQ    =12HABCDEFABCDEF
```



INSTRUCTION LITERAL

```
LDA     =MTRA CAT,7
```

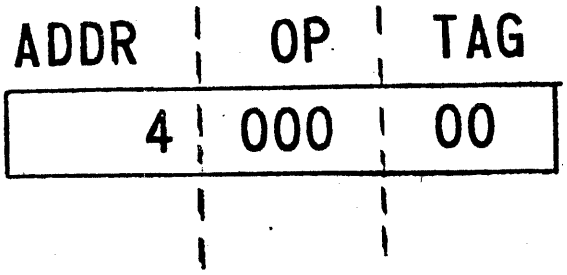
ADDR	OP	TAG
CAT	TRA	07

```
LDA     =MLDA DOG,DU
```

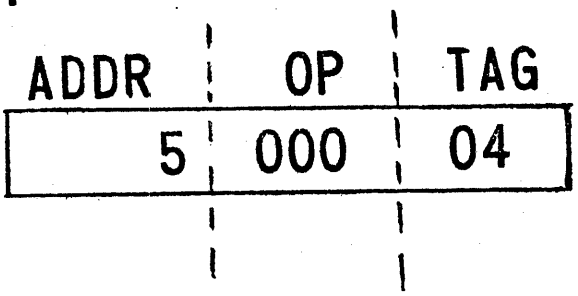
ADDR	OP	TAG
DOG	LDA	DU

ARG PSEUDO OPERATION

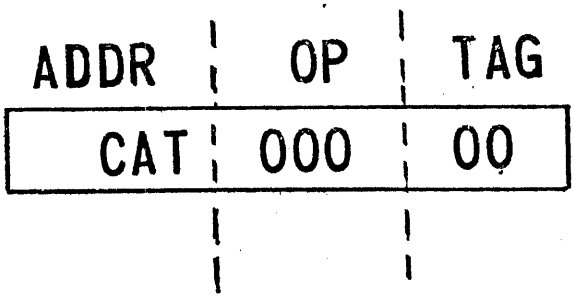
ARG 4



ARG 5,4

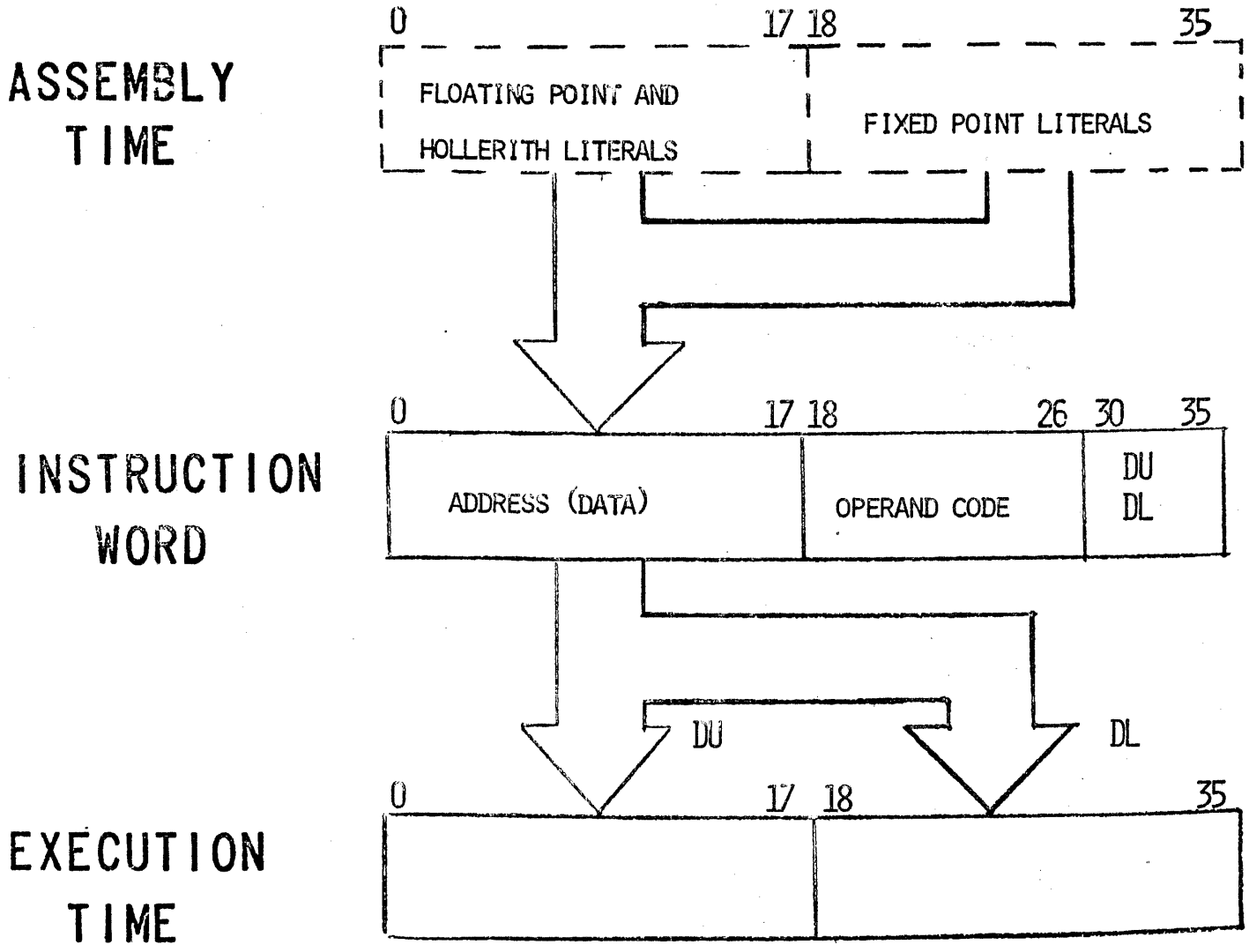


ARG CAT



LITERALS MODIFIED BY DIRECT OPERANDS

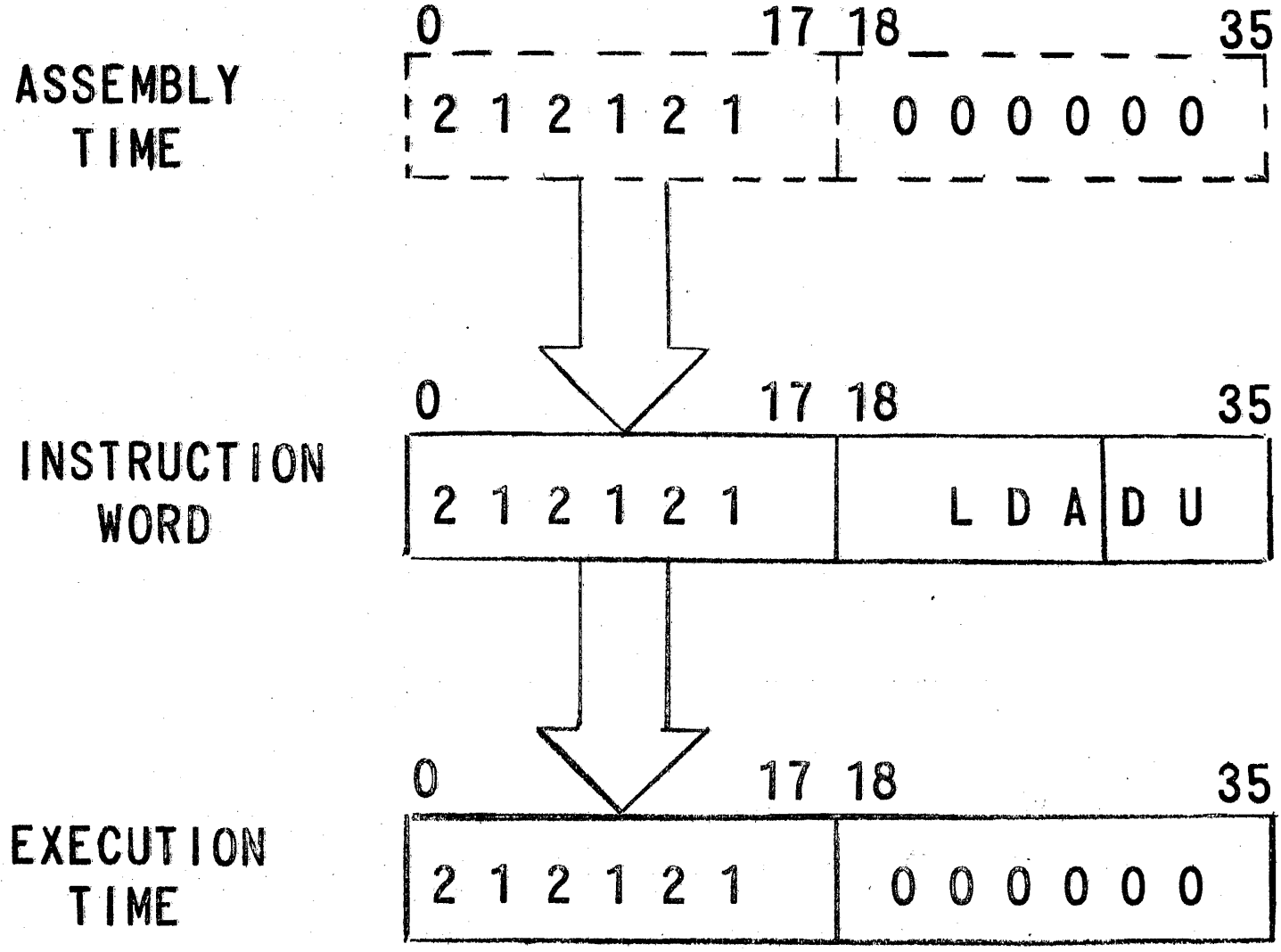
LDA =3HAAA,DU	LDQ =10,DU
LDQ =3HQQQ,DL	LXL_N =124,DL
FLD =327D2,DU	LDX_N =1B18,DU
	*LDA =1B5,DU



*Incorrect coding - will not generate proper literal.

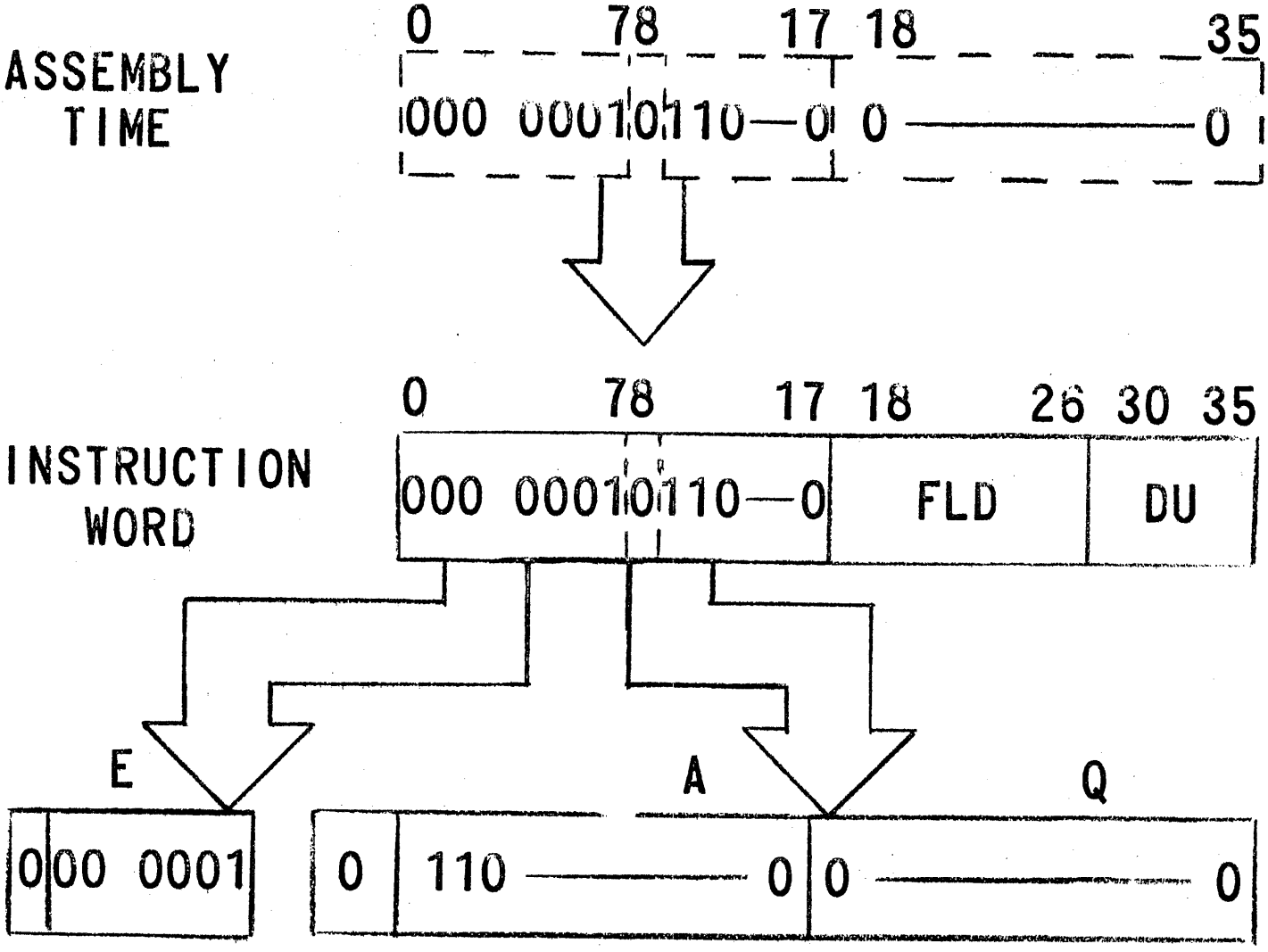
HOLLERITH LITERAL

EXAMPLE: LDA =3HAAA,DU



FLOATING-POINT LITERAL

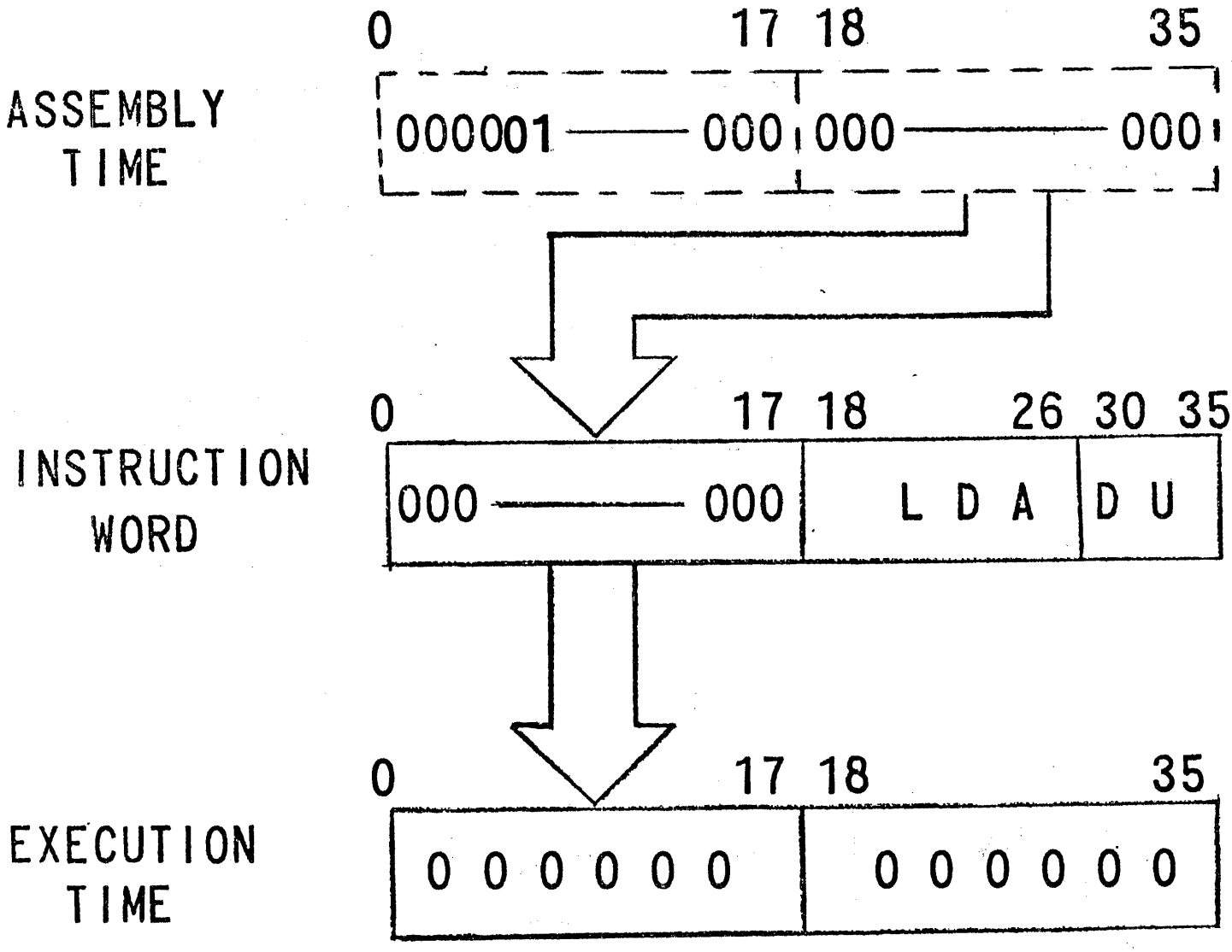
EXAMPLE: FLD =1.5,DU



EXECUTION TIME

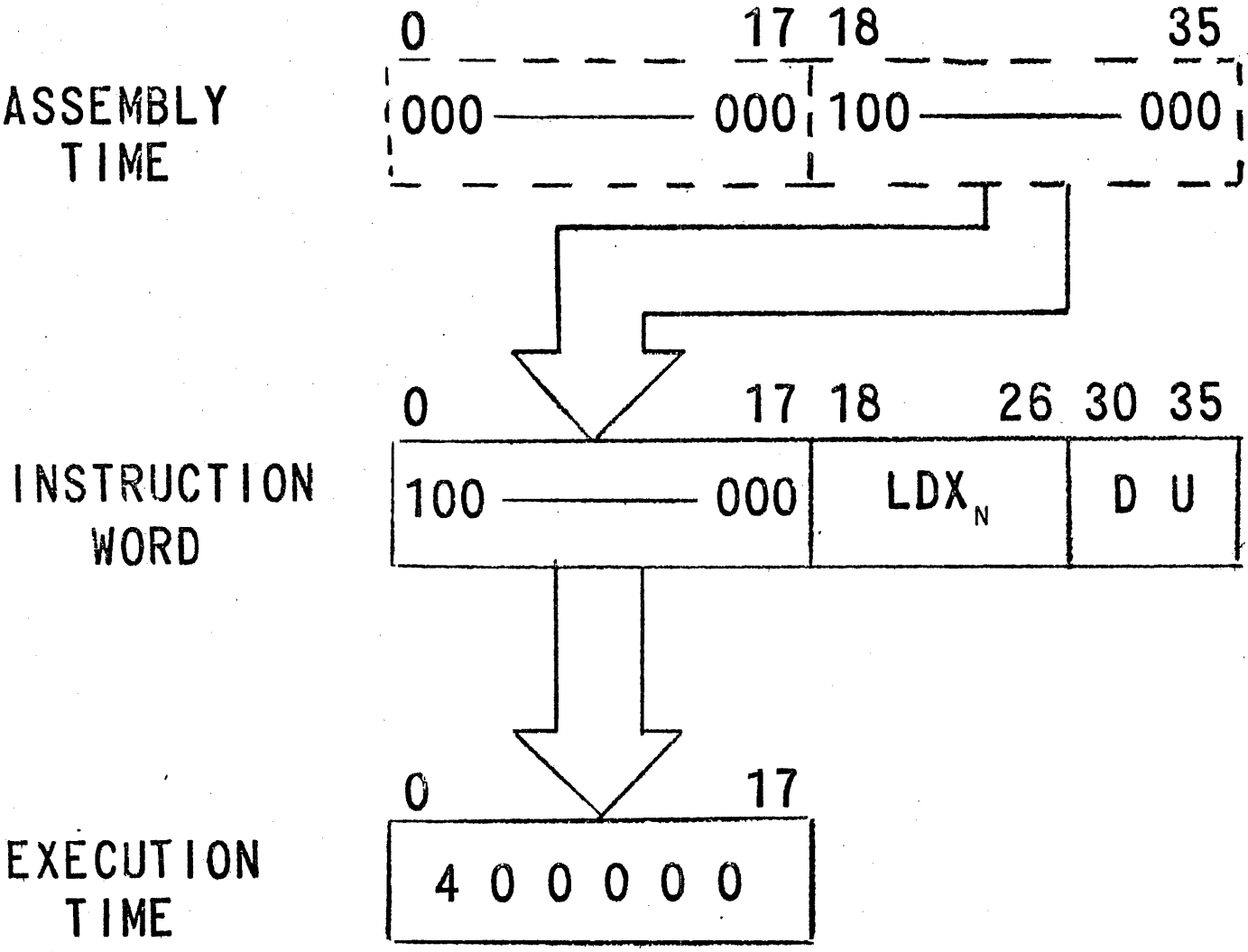
FIXED-POINT LITERAL

EXAMPLE: LDA =1B5,DU

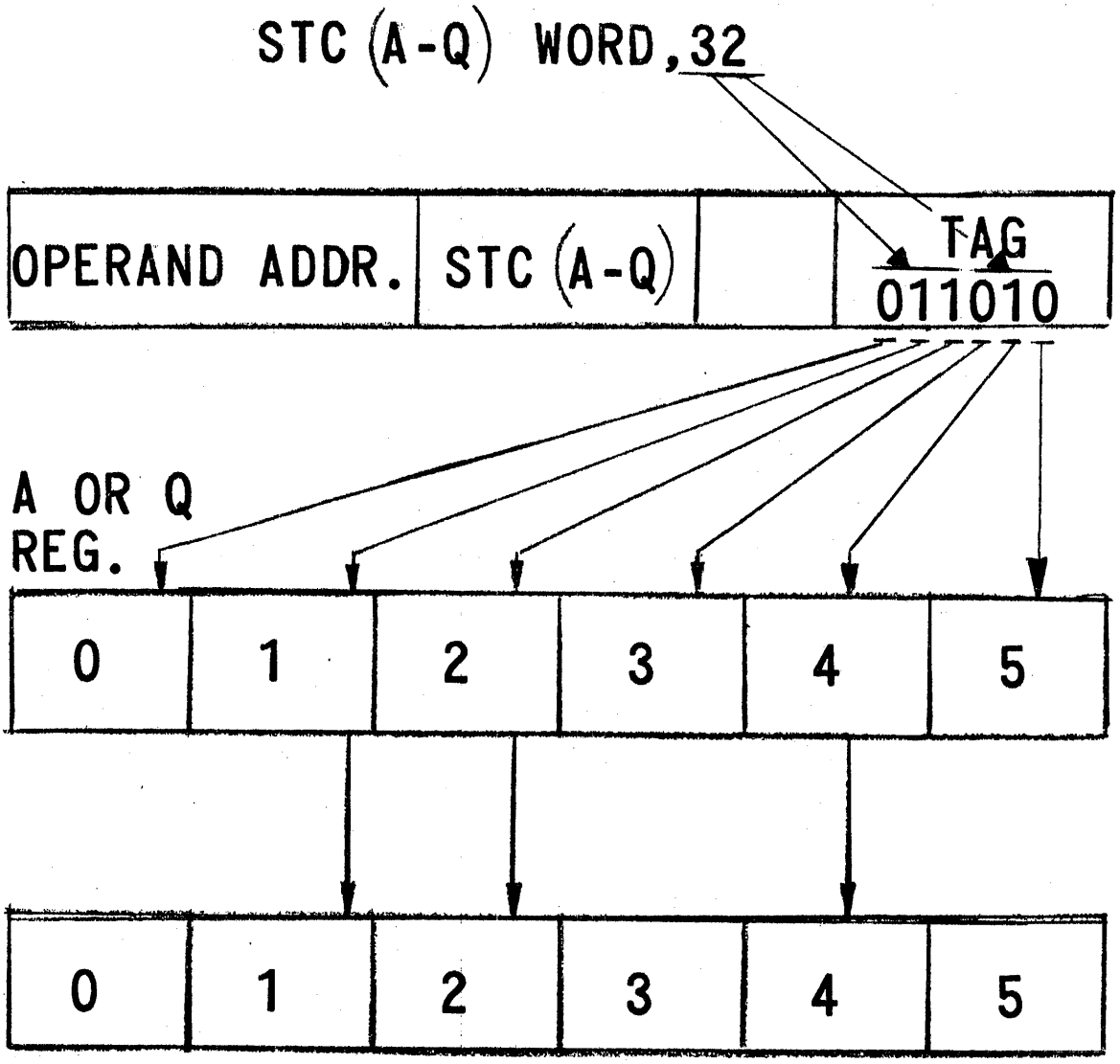


FIXED-POINT LITERAL

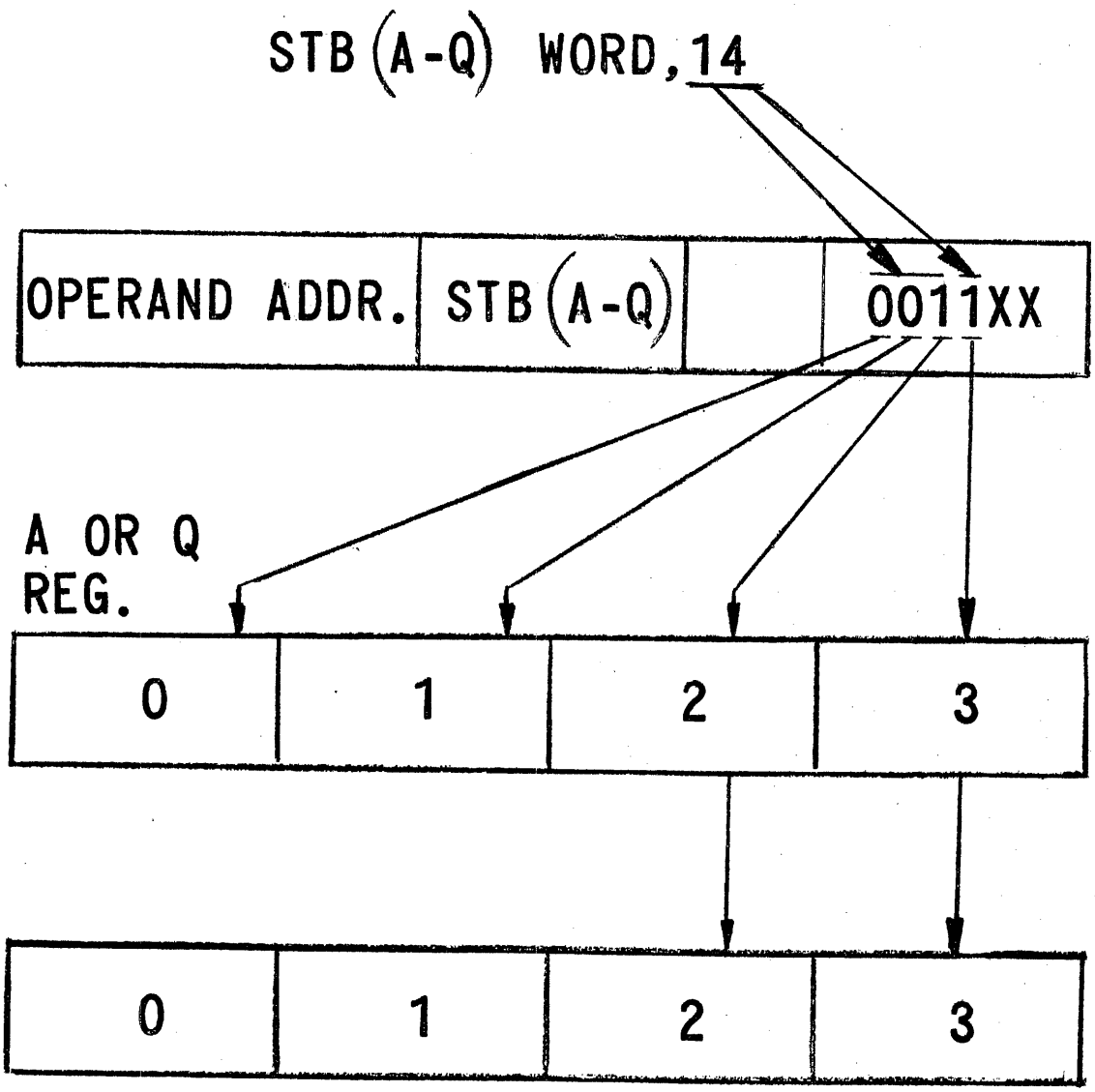
EXAMPLE: LDX_N =1B18,DU



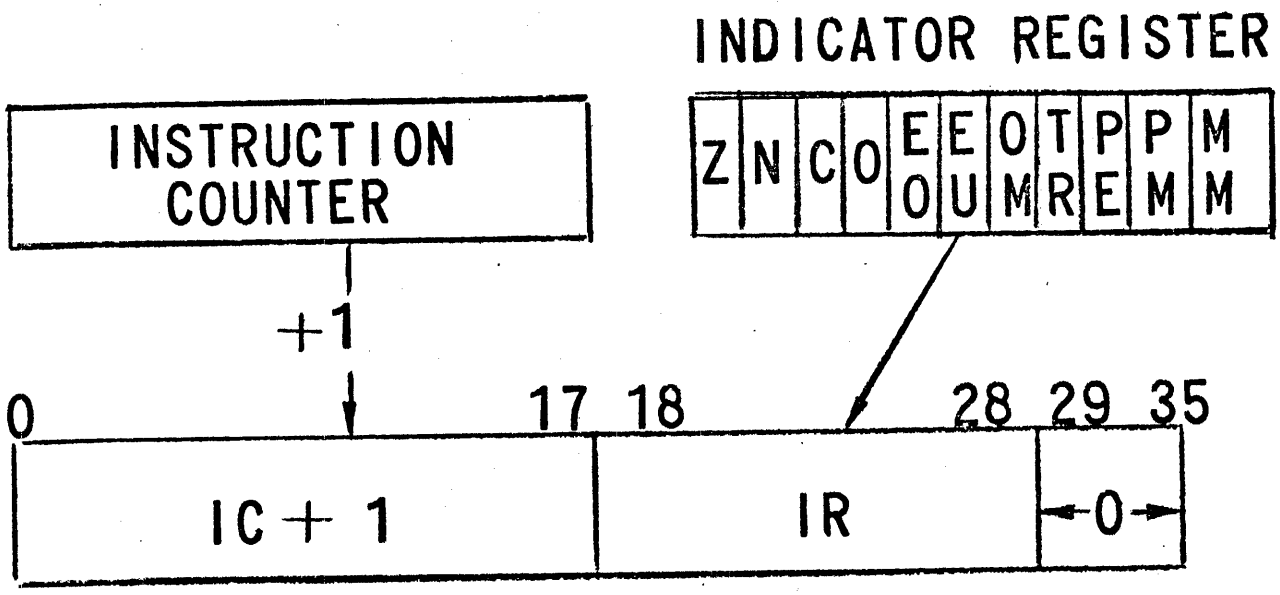
STORE CHARACTER



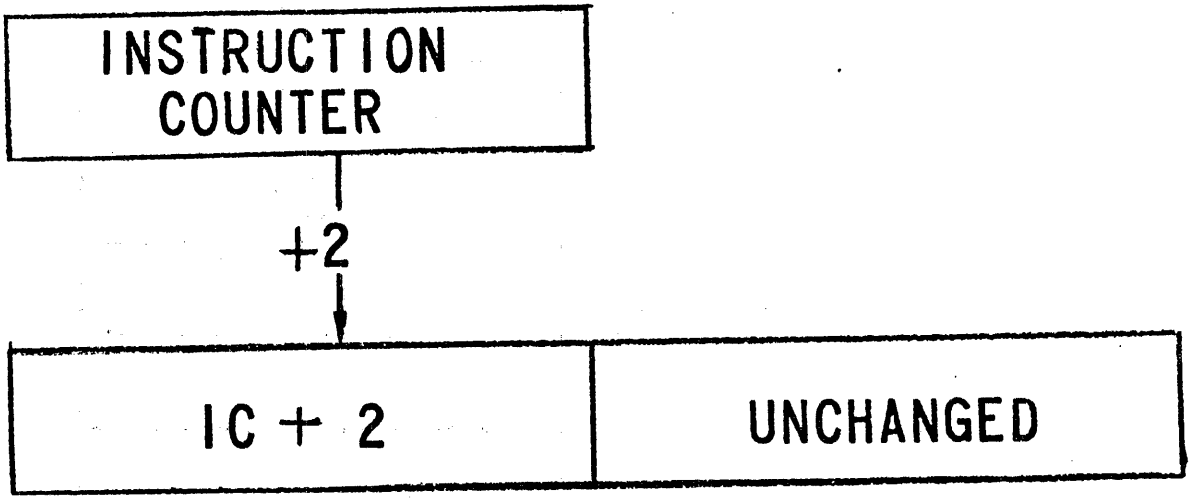
STORE BYTE



STC1


INDICATOR REGISTER
INSTRUCTION COUNTER
Z N C O EO EU OM TR PE PM MM
+1
0
17 18
28 29 35
IC + 1
IR
0

STC2


INSTRUCTION COUNTER
+2
IC + 2
UNCHANGED

SPECIAL
LOAD/STORE

*	LBAR	SBAR
*	LDT	STT
	LDI	STI
	LREG	SREG
*	MASTER MODE	STZ

LREG/SREG

WORD		
0	X0	X1
1	X2	X3
2	X4	X5
3	X6	X7
4	A	
5	Q	
6	E	← 0 →
7	TR (SREG only)	← 0 →

SHIFT INSTRUCTIONS

REGISTER LEFT SHIFTS

1. VACATED POSITIONS ARE FILLED WITH ZEROS.
2. BITS VACATING BIT POSITION ZERO ARE LOST.

REGISTER RIGHT SHIFTS

1. ALL VACATED POSITIONS ARE FILLED WITH THE CONTENTS OF BIT ZERO.
2. BITS VACATING BIT POSITION 35 ARE LOST.

REGISTER RIGHT LOGICAL SHIFTS

1. ALL VACATED POSITIONS ARE FILLED WITH ZEROS.
2. BITS VACATING BIT POSITION 35 ARE LOST.

REGISTER LEFT ROTATE

THE REGISTER BECOMES CIRCULAR. BITS VACATING POSITION ZERO ENTER AGAIN AT BIT POSITION 35.

FIXED POINT ARITHMETIC

ADA	SBA
ADQ	SBQ
ADAQ	SBAQ
ADXn	SBXn
ASA	SSA
ASQ	SSQ
ASXn	SSXn

INDICATORS

ZERO-NEGATIVE-CARRY-OVERFLOW

LOGICAL ADD/SUBTRACT

ADLA	SBLA
ADLQ	SBLQ
ADLAQ	SBLAQ
ADLXn	SBLXn

INDICATORS

ZERO-NEGATIVE-CARRY

O'FLO INDICATOR NOT AFFECTED

CARRY AND OVERFLOW INDICATORS

* CARRY = SCO
* OVERFLOW = $(SCI \cdot \overline{SCO}) + (\overline{SCI} \cdot SCO)$

EXAMPLES USING ALGEBRAIC ADDITION:

	4-BIT REPRESENTATION	INDICATOR SET ON
1. R = +7	0111	
Y = +1	0001	SCI · SCO
+8		
2. R = -5	1011	
Y = -4	1100	SCI · SCO
-9		
3. R = -1	1111	
Y = -4	1100	SCI · SCO
-5		
4. R = -5	1011	
Y = +4	0100	SCI · SCO
-1		

SPECIAL ADD/SUBTRACT

AWCA

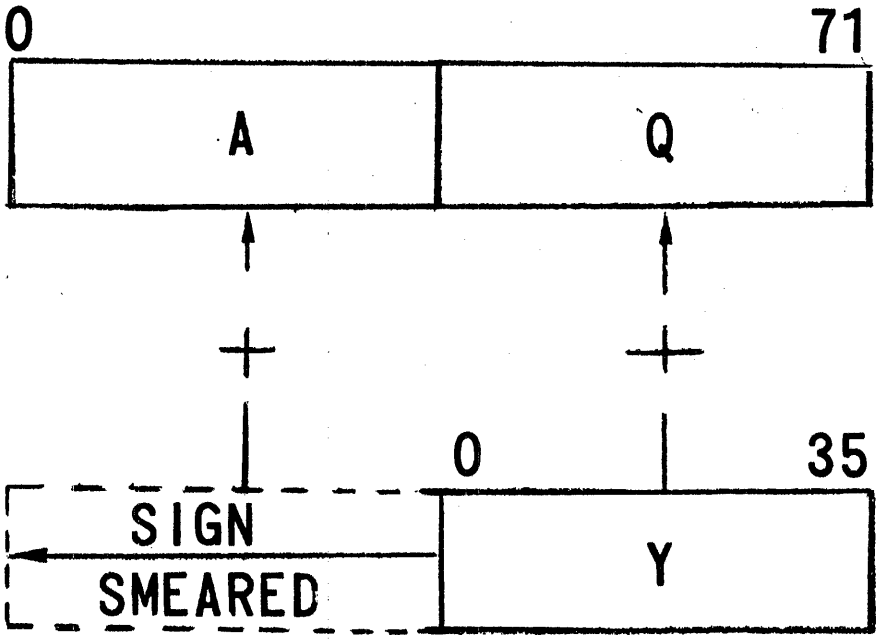
AWCQ

AOS

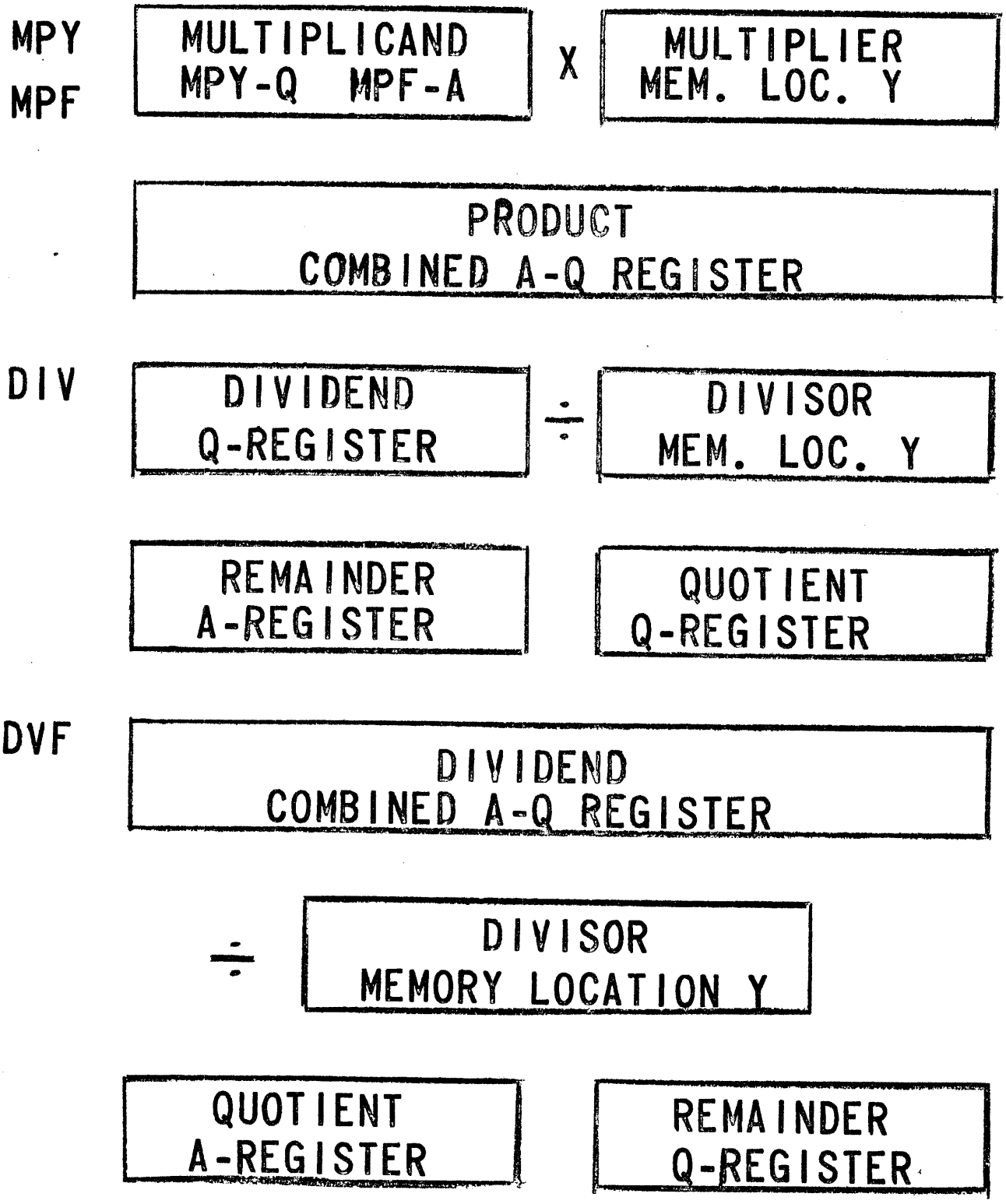
ADL

SWCA

SWCQ



MULTIPLY/DIVIDE



TRANSFER CONTROL

UNCONDITIONAL

TRA	TSS
RET	TSX_N

CONDITIONAL

TZE/TNZ	* TOV
TMI/TPL	* TEO
TRC/TNC	* TEU

* RESETS RESPECTIVE INDICATOR

```
0002  $      IDENT   4817,=TRNG-RGK,J959,C959100,8177
```

```
$$    46221 ENTERED FIN635 AT 17,419 FROM CD RDR   0.06.00

0001  $       $NUMB   46221
0002  $       IDENT   4817,.TRNG-RGK.J959,C959100.8177
0003  $       GMAP    NDECK,COMDK
0004  $       EXECUTE DUMP
0005  $       LIMITS  ,1K,,1000
0006  $       ENDJOB
   TOTAL CARD COUNT THIS JOB = 000068


 * BEGIN ACTIVITY =01-  GMAP     10/06/71   SW=2150000000000
  * NORMAL TERMINATION     AT 002428  INDICATORS  4020

     START 17,443     LINES   166     PROC  0,0003     I/O   0,001    IU  5   MEMORY   24K
     STOP  17,444     LIMIT 10000     LIMIT 0,0400     LIMIT          CU  5   MaT      120

     LAPSE  0,001   FC D TYPE     BUSY    IP/AT    EP/RT    IS/SC MS/SE    ADDRESS L#/T#

                    G. R D167  .    98       0        3        3     3     0-00-08   30
                    P. SYOUT
                    K. SYOUT
                    C. SYOUT
                    .1 R D167      356       0        0        4     4     0-00-03  153
                    B. S D167       80       0        1        2     2     0-00-04  139

     COMDK   10 CARDS
     LIST   156 LINES

 * BEGIN ACTIVITY =02-  GELOAD   10/06/71   SW=4G0000000000
  * USERS 00 PME GEBORT     AT 001601  INDICATORS  0000

     START 17,444     LINES    89     PROC  0,0002     I/O   0,001    IU  5   MEMORY    1K
     STOP  17,448     LIMIT  1000     LIMIT 0,0500     LIMIT          CU  5   MaT       59

     LAPSE  0,004   FC D TYPE     BUSY    IP/AT    EP/RT    IS/SC MS/SE    ADDRESS L#/T#

                    B. R D167       30       1        1        2     2     0-00-04  139
                    R. R D167  .    38       0        0        1     1     0-00-08   30
                    P. SYOUT
                    L. R D167      573       0        0       28   28R     0-00-01  305

     LIST    89 LINES
```

PREFACE

```
PROGRAM BREAK    225
COMMON LENGTH      0
V COUNT BITS       5

PRIMARY SYMDEF  ENTRY

FLOAT              0

SECONDARY SYMDEF  ENTRY

BLOCK      LENGTH

SYMREF
```

```
                                        1          LBL     RGK3
                    000000              2 FLOAT    SAVE
000000  000002710000     010
000001  000222630000     010
000002  000222754000     010
000003  000222741000     010
000004  000101 2350 00   010        3              LDA     TLYREG
000005  000100 7550 00   010        4              STA     REGTLY
000006  000004 6260 00   000        5              EAX6    4               LOAD X6 WITH 4
000007  000010 6270 00   000        6              EAX7    8               LOAD X7 WITH 8
000010  000011 6170 00   010        7 AGAIN        TOV     *+1             RESET OVERFLOW INDICATOR
000011  000000 6360 00   000        8              EAQ     0               ZERO Q-REGISTER
000012  106000 4110 03   000        9              LDE     =35825,DU       SET UP E-REGISTER FOR FP CONVERSION
000013  777777 0660 03   000       10              ABX6    -1,DU           DECREMENT X6 BY ONE
000014  000002 1670 03   000       11              SBX7    2,DU            DECREMENT X7 BY TWO
000015  000020 2350 07   000       12              LDA     =C20,DL
000016  000032 2350 07   000       13 EMINUS      LDA     =C92,DL
000017  000104 0540 00   010       14              AOS     SEQBIN
000020  000140 2370 00   010       15 RTNX        LDAQ    RCWRK
000021  000014 7370 00   000       16              LLS     12
000022  000014 7320 00   000       17              QRS     12
000023  000140 7570 00   010       18              STAQ    RCWRK
000024  000110 3770 00   010       19 CONVRT      ANAQ    MASKA           ZERO ZONE BITS OF A, ALL OF Q
000025  000224 4020 00   010       20              MPY     =10
000026  000112 0360 52   010       21              ADLQ    TAL1,SC
000027  000113 2350 00   010       22              LDA     TAL2
000030  000112 7550 00   010       23              STA     TAL1
000031  000010 0010 00   000       24              MME     GEBCRT
000032  000000000010     000       25 ABIN        DEC     8
000033  000000011007     000
                    000034          26 AFLOATEBSS      2
000036  000002710004     000
                    000040          27              EIGHT
                    000040          28 SAVREG      BSS     32
000100  000040 0004 10   010       29 REGTLY      TALLYD  SAVREG,4,8
000101  000040 0004 10   010       30 TLYREG      TALLYD  SAVREG,4,8
000102  000000000000     000       31 EBIN        DEC     0
000103  000000000000     000       32 FBIN        DEC     0
000104  000000000000     000       33 SEQBIN      DEC     0
000105  000000000000     000       34 ESIGN       DEC     0
000106  000000000000     000       35 FSIGN       DEC     0
000107  000000011007     000
000110  171717171717     000       36 MASKA EOCT      171717171717
000111  000000000000     000       37              OCT     000000000000
000112  000000 0006 00   000       38 TAL1        TALLY   MYWRK,6,0
000113  000000 0006 00   000       39 TAL2        TALLY   MYWRK,6,0
000114  000000001440     000       40 CONTB       DEC     800,640,512
000115  000000001200     000
000116  000000001000     000
000117  073465450000     000       41 CONTAB      DEC     8E9B35,6468B35,512E7B35,4096E6B35,32768000000,2621448000
000120  057536040000     000
000121  046113200000     000
```

```
000122  036411000000    000
000123  030324000000    000
000124  023420000000    000
000125  017500000000    000     42          DEC     2097152000,1677721600,1342177280,1073741824
000126  014400000000    000
000127  012000000000    000
000130  010000000000    000
000131  000000011007    000
                000132          43 EBCD    EBSS    2
                000134          44 FBCD    BSS     2
                000136          45 DUMMY   BSS     1
                000137          46 SEQBCD  BSS     1
000140  202020202020    000     47 RDWRK   EBCI    5,      8  2    2          1
000141  201020202002    000
000142  202020202002    000
000143  202020202020    000
000144  202020202001    000
                000145          48         BSS     9
000156  202020202020    000     49 PRTOUT  OCT     202020202020
000157  202020000000    000     50 PRSEQ   OCT     202020008800
000160  202020202020    000     51         OCT     202020202020
000161  202020202020    000     52 PRE     BCI     3,
000162  202020202020    000
000163  202020202020    000
000164  202020202020    000     53 PRF     BCI     3,
000165  202020202020    000
000166  202020202020    000
000167  202020202020    000     54 PRVAR   BCI     6,
000170  202020202020    000
000171  202020202020    000
000172  202020202020    000
000173  202020202020    000
000174  202020202020    000
000175  202020202020    000     55         BCI     6,
000176  202020202020    000
000177  202020202020    000
000200  202020202020    000
000201  202020202020    000
000202  202020202020    000
000203  770200000000    000     56         OCT     770200000000
                000204          57 PUNOUT  BSS     0
000204  202020202020    000     58 PUE     BCI     2,
000205  202020202020    000
000206  202020202020    000     59 PUF     BCI     2,
000207  202020202020    000
000210  202020202020    000     60 PUSEQ   BCI     1,
000211  202020202020    000     61 PUVAR   BCI     9,
000212  202020202020    000
000213  202020202020    000
000214  202020202020    000
000215  202020202020    000
000216  202020202020    000
```

```
000217  202020202020    000
000220  202020202020    000
000221  202020202020    000

ERROR LINKAGE

000222  000000000000    000
000223  2643462163 20    000

  LITERALS

000224  000000000012    000
                                    62        END
  225 IS THE NEXT AVAILABLE LOCATION,   GMAP VERSION JRRA/030271 JRRB/030271 JRRC/030871
THERE WERE      2 WARNING FLAGS IN THE ABOVE ASSEMBLY
ON PAGE NO.
    2
```

OCTAL	SYMBOL	REFERENCES BY ALTER NO.			
10	GEBOOT		24		
222	.E.L..		2		
110	MASKA	36	19	36	
140	MDWRK	47	15	18	47
100	REGTLY	29	4	29	
40	SAVREG	28	28	29	30
104	SEQBIN	33	14	33	
112	TAL1	38	21	23	38
113	TAL2	39	22	39	
101	TLYREG	30	3	30	

** 18235 WORDS OF MEMORY WERE USED BY GMAP FOR THIS ASSEMBLY.

UNDEFINED SYMBOLS

MYWRK

ORIGIN 062170 ENTRY LOCATION ENTRY LOCATION ENTRY LOCATION ENTRY LOCATION ENTRY LOCATION

SUBPROGRAMS INCLUDED IN DECK.

001550 100671 FLOAT 001550

SUBPROGRAMS OBTAINED FROM SYSTEM LIBRARY.

```
001472  021271  ;SETU, 001500  ,FRSIQ 001672
001464  073069  ;FLTPR 001464
```

	RANGE	SIZE
ALLOCATED CORE	000000 THRU 001777	002000
OBJECT PROGRAM		
RELOCATABLE	001464 THRU 001777	000314

1K, IS THE MINIMUM MEMORY NEEDED TO LOAD THIS ACTIVITY

EXECUTION PROGRAM ENTERED AT 001550

```
EI 001662755000  OI 000010001000  IC 001601  IR 000001  ER 043  AR 000000000600  QR 000000000000  TR 00007064
BA 434002  X0 000000  X1 001542  X2 000000  X3 000000  X4 000000  X5 000000  X6 000003  X7 000006

0000005  000000000000  000000000000  000000000000  000000000000  000000000000  000000000000  000000000000  000000000000
000010   000000000000  000000000000  000000000000  001601020080  000000000000  000045074214  000000000000  000000000000
000020   000000000000  000000000000  001601000001  000000000000  001550000000  000000000000  000000000000  000000000000
000030   000000000000  000000000000  000000000000  000000000080  000000000000  000000000000  020406020201  000101001463
000040   001334007141  001440433752  777212434000  000012000000  000000000000  000175000001  000000000000  000075220000
000050   000000000000  000000000000  000000000000  000000000000  000000000000  000000000000  000000000000  000000000000
000060   000000000000  000000000000  000000000000  000000000000  000000000000  000000000000  202020041001  077352635145
000070   275251274273  411105117323  110511010000  731001070720  202020202020  202020202020  202020202020  202020202020
000100   000000000000  000000000000  000000000000  000000000000  000000000000  000000000000  000000000000  000000000000
001460*  000000000000  000000000000  000000000000  000000000000  000007450000  000011450000  000000710011  000103010611
001470   000000000000  332643634751  000000710011  000101001463  000000000000  001464000000  000000000000  001550000000
001500   001475222000  001476235000  000100102003  001510600000  000001122003  000000755012  000100102003  001504001000
001510   000000221003  030200520201  000000450011  000031450000  001475235000  000000635005  001520600000  001542035000
001520   000015755000  001474235000  000025755000  001477235000  000024755000  001473235000  000037755000  001464701000
001530   001532710000  001545000047  000000220003  000000222003  000000223003  000000224003  000000225003  000000226003
001540   000000227003  001477201051  000000710000  001545000050  000102050701  000000000000  336225636433  000000000000
001550   001552710000  001772630000  001772754000  001772741000  001651235000  001650795000  000804626000  000010627000
001560   001561617000  000000636000  106000411003  777777066003  000002167003  000020235007  000092235007  001654054000
001570   001710237000  000014737000  000014732000  001710757000  001660377000  001774402000  001662036052  001663235000
001600   001662755000  000010001000  000000000010  000000011007  000000000000  000002710004  000000000000  000000000000
001610   000000000000  000000000000  000000000000  000000000000  000000000000  000000000000  000000000000  000000000000
001650*  001610000410  001610000410  000000000000  000000000000  000000000001  000000000000  000000000000  000000011007
001660   171717171717  000000000000  000000000600  000000000600  000000001440  000000001200  000000001000  073465450000
001670   057536040000  046113200000  036411000000  030324000080  023320000000  017500000000  014400000000  012000000000
001700   010000000000  000000011007  000000000000  000000000000  000000000000  000000000000  000000000000  000000000000
001710   202020202010  000020202002  202020202002  202020202020  202020202001  000000000000  000000000000  000000000000
001720   000000000000  000000000000  000000000000  000000000000  000000000000  000000000000  202020202020  202020000000
001730   202020202020  202020202020  202020202020  202020202020  202020202020  202020202020  202020202020  202020202020
001750*  202020202020  202020202020  202020202020  770200000000  202020202020  202020202020  202020202020  202020202020
001760   202020202020  202020202020  202020202020  202020202020  202020202020  202020202020  202020202020  202020202020
001770   202020202020  202020202020  001542502000  264346216320  000000000012  000000000000  000000000000  000000000000
```

SAVING INSTRUCTION COUNTER IN INDEX REGISTER

MAIN LINE CODING:

```
          .
          .
          .
RETURN ->  TSX7     SUB     STORES IC+1 IN X7
          .
          .
          .
```

SUBROUTINE CODING:

```
SUB       XXX
          .
          .
          .
          TRA       0,7  <-- X7 CONTAINS ADDRESS OF
                             RETURN
```

SAVING INSTRUCTION COUNTER IN MEMORY

MAIN LINE CODING:

```
                .
                .
                .
        STC2    BACK    STORES IC+2
        TRA     SUB
RETURN  .
        .
        .
```

SUBROUTINE CODING:

```
SUB     XXX
        .
        .
        .
BACK    TRA     **  <-- INDICATES ADDRESS REPLACED
                        BEFORE USE
```

SAVING INSTRUCTION COUNTER AND INDICATORS (STC1)

MAIN LINE CODING:

```
                .
                .
                .
        STC1    BACK    STORES IR + IC+1
        TRA     SUB
RETURN ------>  .
                .
                .
```

SUBROUTINE CODING:

```
SUB     XXX
        .
        .
        .
        EAX7    1
        ASX7    BACK    INCREMENTS ADDRESS PORTION OF
*                       BACK BY 1 - ADDRESS OF RETURN
        RET     BACK    RESTORES INDICATORS AND TRANS-
*                       FERS INDIRECT TO RETURN
        .
        .
        .
BACK    BSS     1       MEMORY LOCATION FOR INDICATORS
*                       AND IC
```

SAVING INSTRUCTION COUNTER AND INDICATORS (STI)

MAIN LINE CODING:

```
                :
                :
                :
                STI     BACK    STORES IR
                STC2    BACK    STORES IC+2
                TRA     SUB
RETURN ------>  :
                :
                :
```

SUBROUTINE CODING:

```
SUB             XXX
                :
                :
                :
                RET     BACK    RESTORES INDICATORS AND
*                               TRANSFERS INDIRECT
                :
                :
BACK            BSS     1       STORAGE FOR IC & IR
```

SAVING REGISTERS

INDICATORS AND IC

MAIN LINE CODING:

```
                .
                .
                .
        SREG    SAVREG
        STI     SAVIND
        STC2    SAVIND
        TRA     SUB
RETURN  .
        .
        .
```

SUBROUTINE CODING:

```
SUB     XXX
        .
        .
        .
        LREG    SAVREG      RESTORE REGISTERS
        RET     SAVIND      RESTORE IR & IC
        .
        .
        .
        EIGHT               JUMP TO 0-MOD-8 LOCATION
SAVREG  BSS     8           STORE REGISTERS HERE
SAVIND  BSS     1           SAVE IR & IC
```

CALL PSEUDO-OPERATION

CODING:

```
CALL      SUB
```

GENERATED CODING:

```
TSX1      SUB
TRA       * + 2
ZERO      .E.L... N*
  .
  .
  .
```

*N = ALTER NUMBER OF CALL

CALL PSEUDO-OP

EXPANDED VERSION

CODING:

```
CALL    SUB(A1,A2,A3)E1,E2'E.I.'
```

GENERATED CODING:

```
TSX1    SUB
TRA     * + 2 + 3 + 2
ZERO    .E.L.,.E.I.
ARG     A1   }
ARG     A2   }   ARGUMENTS FOR SUB
ARG     A3   }
TRA     E2   }
TRA     E1   }   ERROR RETURNS*
 .
 .
 .
```

* NOTE REVERSE ORDER

SAVE PSEUDO OPERATION

EXPANDED VERSION

CODING:

```
SYMBOL      SAVE      0,5,7
```

GENERATED INSTRUCTIONS:

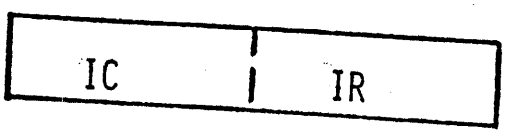
```
SYMBOL      TRA       *+2+3
            LDX0      **,DU
            LDX5      **,DU
            LDX7      **,DU
            RET       .E.L..
            STI       .E.L..
            STX1      .E.L..
            STX0      SYMBOL+1
            STX5      SYMBOL+2
            STX7      SYMBOL+3
              .
              .
              .
```

CALL SAVE AND RETURN PSEUDO OPS

```
1       8       16
NOW     CALL    SUB

NOW     TSX1    SUB
        TRA     *+2
        ZERO    .E.L..,0
        XXX
```

```
SUB     SAVE

SUB     TRA     *+2
        RET     .E.L..
        STI     .E.L..
        STX1    .E.L..
          )
          )     SUBROUTINE BODY
        RETURN  SUB

        TRA     SUB+1
```

NOTE: .E.L..

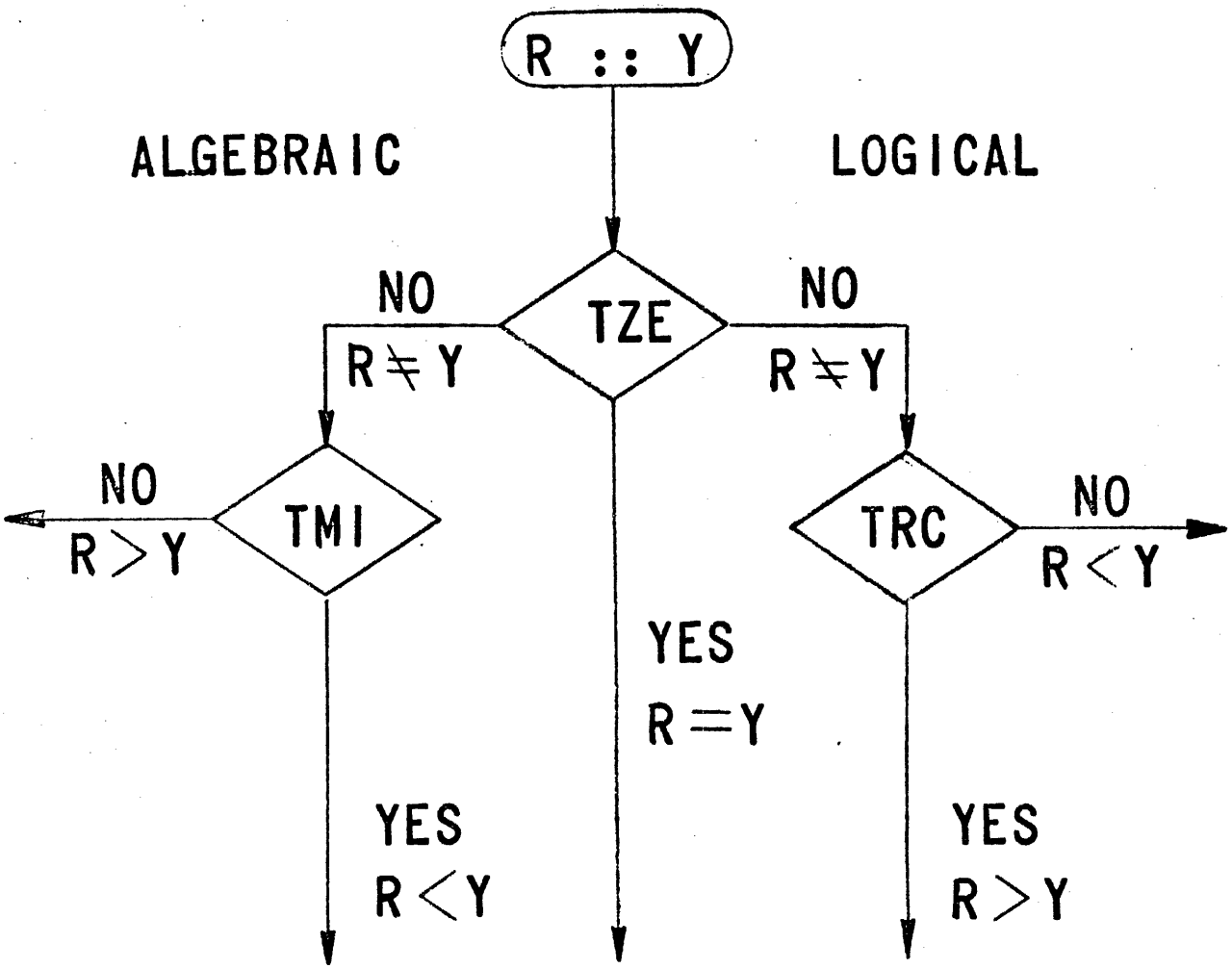


CALL SAVE AND RETURN PSEUDO OPS

```
START     CALL     SUB,(LOC1,LOC2,LOC3),ERR1,ERR2

START     TSX1     SUB
          TRA      *+7
          ZERO     .E.L..,0
          ARG      LOC1          DATA LOCATIONS TO BE PASSED
          ARG      LOC2          TO SUBROUTINE 'SUB'.
          ARG      LOC3
          TRA      ERR2          LOCATIONS OF ERROR ROUTINES,
          TRA      ERR1          OR MME GEBORTS.
          XXX
```

```
SUB       SAVE     5,7,2

SUB       TRA      *+5
          LDX5     **,DU         RESTORE INDEX REGISTERS 5,7, & 2.
          LDX7     **,DU
          LDX2     **,DU
          RET      .E.L..
          STI      .E.L..
          STX1     .E.L..
          STX5     SUB+1         SAVE INDEX REGISTERS 5,7, & 2.
          STX7     SUB+2
          STX2     SUB+3
          .
          .        SUBROUTINE BODY
          .
ERROR1    RETURN   SUB,1

ERROR1    LDX1     .E.L..,*      REGISTER & INDIRECT TO .E.L..
          SBX1     1,DU
          STX1     .E.L..
          TRA      SUB+1

ERROR2    RETURN   SUB,2

ERROR2    LDX1     .E.L..,*
          SBX1     2,DU
          STX1     .E.L..
          TRA      SUB+1
```

COMPARE LOGIC FLOW



POSITIVE NUMBERS AND COMPARISON INDICATORS

REGISTER = 8
MEMORY = 8

001000
001000

001000
111000
1000000

SCO · SCI

ZERO
CARRY

REGISTER = 8
MEMORY = 7

001000
000111

001000
111001
1000001

SCO · SCI

CARRY

REGISTER = 8
MEMORY = 9

001000
001001

001000
110111
111111

NEGATIVE

NEGATIVE NUMBERS AND COMPARISON INDICATORS

REGISTER = -8	111000	111000
MEMORY = -8	111000	001000
		1000000
		ZERO
	SCO · SCI	CARRY

REGISTER = -8	111000	111000
MEMORY = -7	111001	000111
		111111
		NEGATIVE

REGISTER = -8	111000	111000
MEMORY = -9	110111	001001
		1000001
	SCO · SCI	CARRY

COMPARISON INDICATORS

POSITIVE REGISTER - NEGATIVE MEMORY

REGISTER = 8	001000	001000
MEMORY = -8	111000	001000
		010000

NO INDICATORS SET

REGISTER = 8	001000	001000
MEMORY = -7	111001	000111
		001111

NO INDICATORS SET

NEGATIVE REGISTER - POSITIVE MEMORY

REGISTER = -8	111000	111000
MEMORY = 8	001000	111000
		1110000

SCO · SCI — NEGATIVE CARRY

ALGEBRAIC COMPARISON

ZERO	NEGATIVE	CARRY	RELATION	SIGN
0	0	0	C(A) > C(Y)	$C(A)_0 = 0,\ C(Y)_0 = 1$
0	0	1	C(A) > C(Y)	$C(A)_0 = C(Y)_0$
1	0	1	C(A) = C(Y)	
0	1	0	C(A) < C(Y)	
0	1	1	C(A) < C(Y)	$C(A)_0 = 1,\ C(Y)_0 = 0$

LOGICAL COMPARISON

ZERO	CARRY	RELATION
0	0	C(A) < C(Y)
1	1	C(A) = C(Y)
0	1	C(A) > C(Y)

FIXED POINT COMPARE

CMPA	Y
CMPQ	Y
CMPAQ	Y-PAIR
$CMPX_N$	$Y_{0...17}$

SPECIAL PURPOSE COMPARES

ALGEBRAIC COMPARE WITH LIMITS

CWL Y

COMPARE MAGNITUDE

CMG Y

COMPARE MASKED

CMK Y

BOOLEAN OPERATIONS

TRUTH TABLE

	ANDING	ELIMINATES BITS
ANA	ANSA	0 · 0 = 0
		0 · 1 = 0
ANQ	ANSQ	1 · 0 = 0
		1 · 1 = 1
ANAQ		
ANXN	ANSXN	

	ORING	INSERTS BITS
ORA	ORSA	0 + 0 = 0
		0 + 1 = 1
ORQ	ORSQ	1 + 0 = 1
		1 + 1 = 1
ORAQ		
ORXN	ORSXN	

	EXCLUSIVE OR	BINARY ADD (NO CARRY)
ERA	ERSA	0 ≠ 0 = 0
		0 ≠ 1 = 1
ERQ	ERSQ	1 ≠ 0 = 1
		1 ≠ 1 = 0
ERAQ		
ERXN	ERSXN	

EXAMPLE OF ANDING

```
ANA    MASK
ANA    =020401020402
```

AREG	BEFORE	0	7	0	6	0	5	0	4	0	3	0	2
AREG	BEFORE	000	111	000	110	000	101	000	100	000	011	000	010
MASK		000	010	000	100	000	001	000	010	000	100	000	010
AREG	AFTER	000	010	000	100	000	001	000	000	000	000	000	010
AREG	AFTER	0	2	0	4	0	1	0	0	0	0	0	2

EXAMPLE OF ORING

ORA MASK
ORA =020401020402

AREG	BEFORE	0	7	0	6	0	5	0	4	0	3	0	2
AREG	BEFORE	000	111	000	110	000	101	000	100	000	011	000	010
MASK		000	010	000	100	000	001	000	010	000	100	000	010
AREG	AFTER	000	111	000	110	000	101	000	110	000	111	000	010
AREG	AFTER	0	7	0	6	0	5	0	6	0	7	0	2

EXAMPLE OF THE EXCLUSIVE OR

ERA MASK
ERA =020401020402

AREG BEFORE	0	7	0	6	0	5	0	4	0	3	0	2
AREG BEFORE	000	111	000	110	000	101	000	100	000	011	000	010
MASK	000	010	000	100	000	001	000	010	000	100	000	010
AREG AFTER	000	101	000	010	000	100	000	110	000	111	000	000
AREG AFTER	0	5	0	2	0	4	0	6	0	7	0	0

BOOLEAN COMPARE OPERATIONS

COMPARATIVE AND

CANA	Y
CANQ	Y
CANAQ	Y-PAIR
$CANX_N$	Y

COMPARATIVE NOT AND

CNAA	Y
CNAQ	Y
CNAAQ	Y-PAIR
$CNAX_N$	Y

COMPARATIVE AND

CANA MASK
TNZ BITON

MASK	0010
AREG	1111
Z=	0010

CANA	COMPARATIVE AND TO AREG
CANQ	COMPARATIVE AND TO QREG
CANAQ	COMPARATIVE AND TO AQREG
CANX$_N$	COMPARATIVE AND TO XREG

COMPARATIVE NOT AND

CNAA MASK
TNZ BITON

MASK	1101
1^S COMPLEMENT	0010
AREG	1111
Z=	0010

CNAA	COMPARATIVE NOT AND TO AREG
CNAQ	COMPARATIVE NOT AND TO QREG
CNAAQ	COMPARATIVE NOT AND TO AQREG
$CNAX_N$	COMPARATIVE NOT AND TO XREG

EXAMPLE OF BOOLEAN OPERATIONS AND BOOLEAN COMPARES

```
CHECK   LDA     SWITCH
        ANA     =Ø241000,DU
        CANA    =Ø200000,DU
        TNZ     BCD
        CANA    =Ø040000,DU
        TNZ     ASCII
        CNAA    =Ø776000,DU
        TNZ     BINARY
ERROR   LDA     =Ø400000,DU
        ORSA    SWITCH
        TRA     COMRET

BCD     CANA    =Ø041000,DU
        TNZ     ERROR
          .
          .
OKBCD   LDA     =Ø200040,DU
        ERSA    SWITCH
        TRA     COMRET
NOKBCD  LDA     =Ø200004,DU
        ERSA    SWITCH
COMRET  TRA     XX
```

INDIRECT THEN TALLY (IT) MODIFICATION

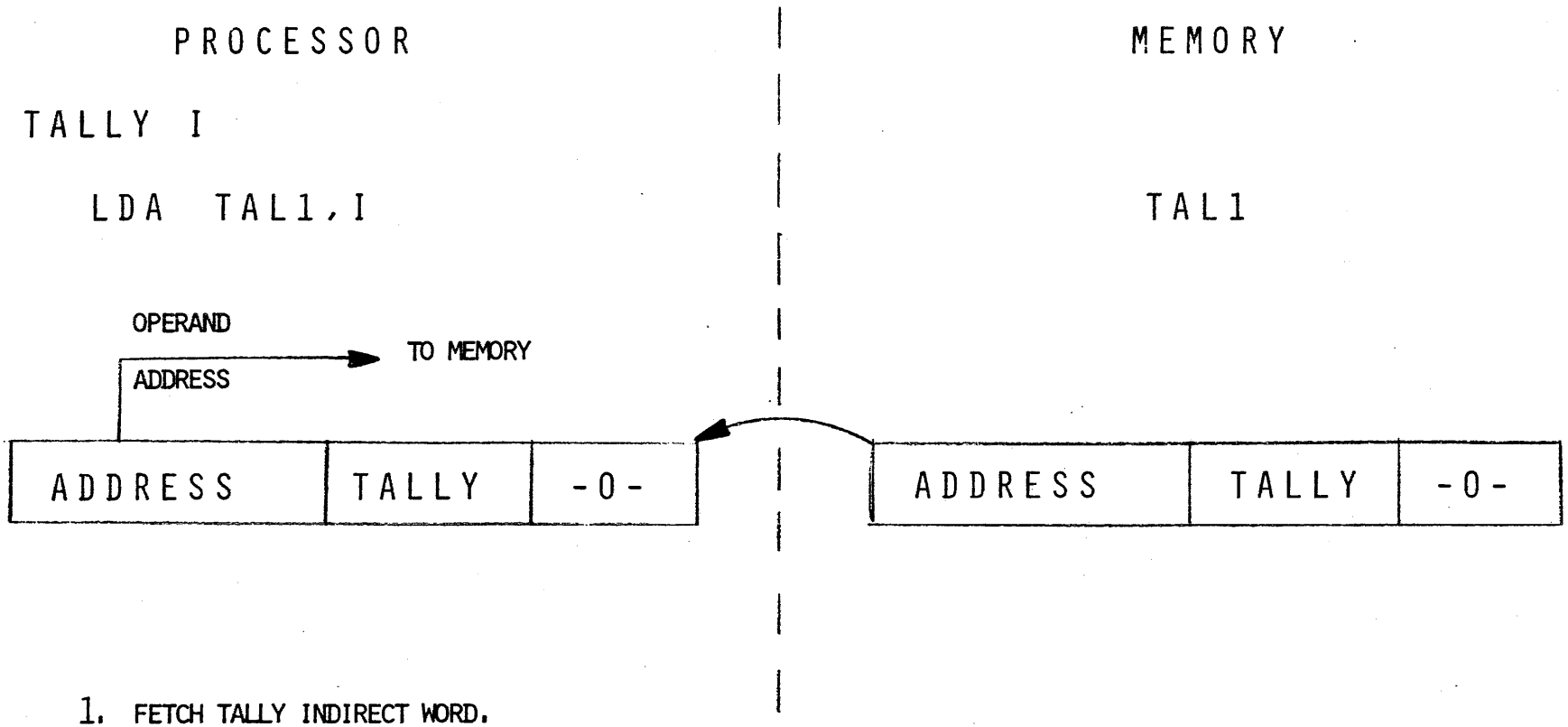
VARIATIONS

I	INDIRECT
ID	INCREMENT ADDRESS - DECREMENT TALLY
DI	DECREMENT ADDRESS - INCREMENT TALLY
AD	ADD DELTA
SD	SUBTRACT DELTA
*CI	CHARACTER FROM INDIRECT
*SC	SEQUENCE CHARACTER
*SCR	SEQUENCE CHARACTER REVERSE
F	FAULT
IDC	INCREMENT ADDRESS - DECREMENT TALLY AND CONTINUE
DIC	DECREMENT ADDRESS - INCREMENT TALLY AND CONTINUE

*CHARACTER HANDLING

INDIRECT -I-
EXAMPLE

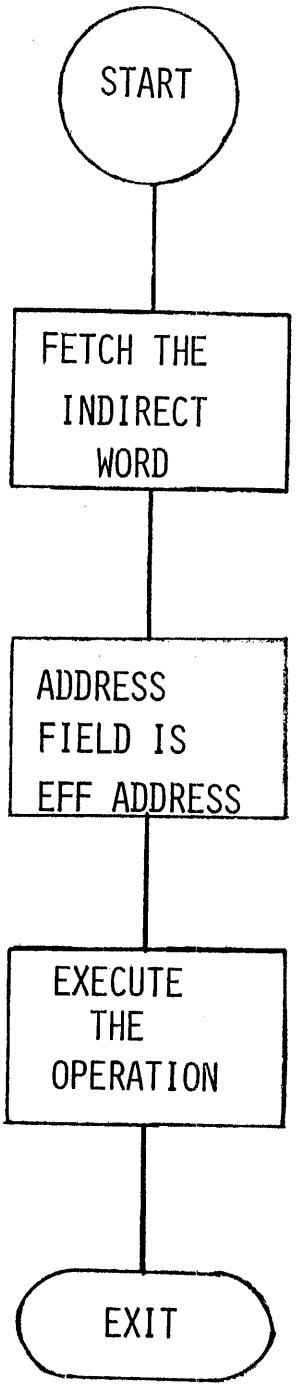
```
                        1
1       8               6
        TSX1    SUBR
          .
          .
          .
SUBR    STX1    ERLKG
          .
          .
          .
        TRA     ERLKG,I
          .
          .
          .
          .
ERLKG   BSS     1
```



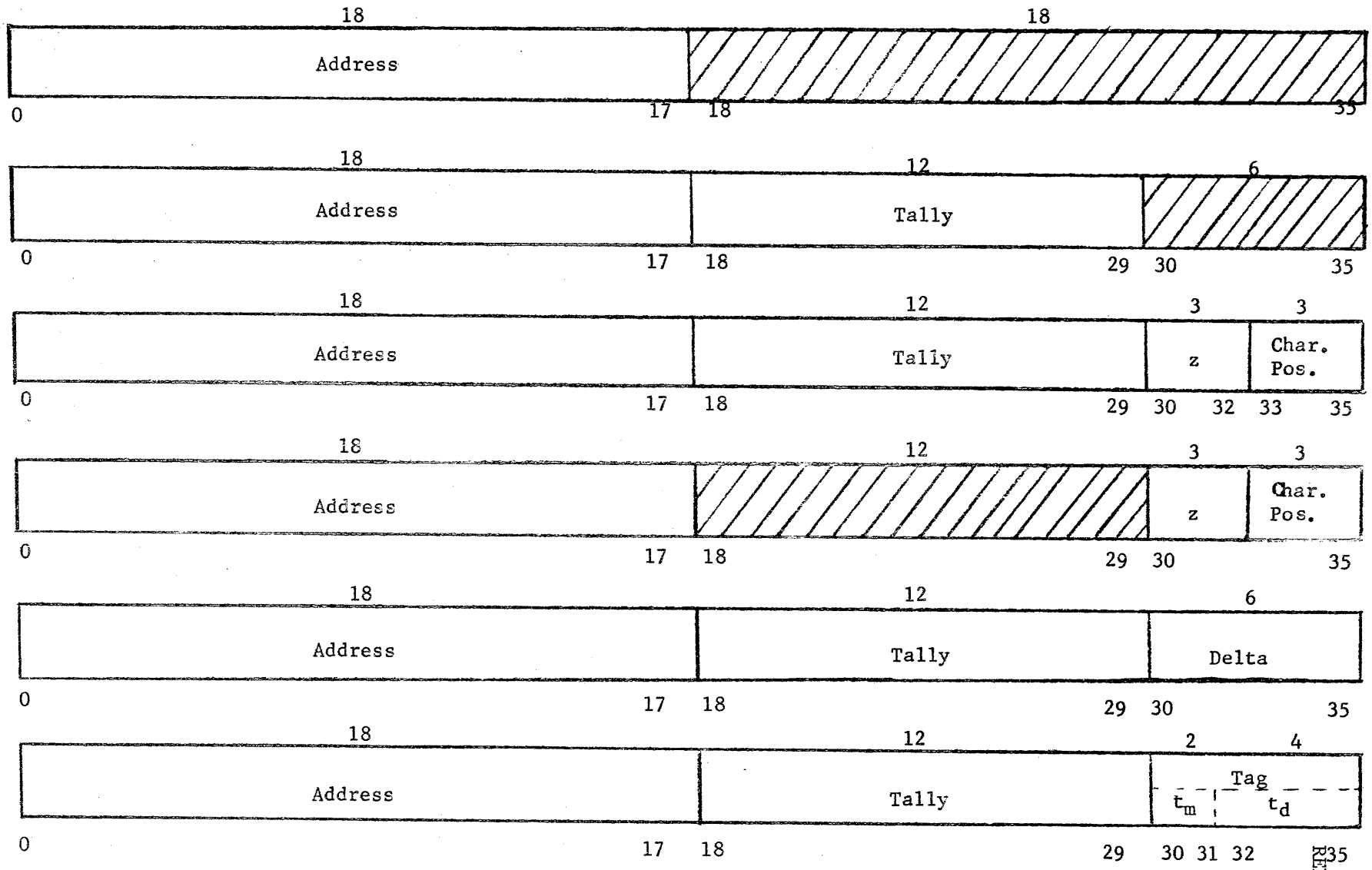
1. FETCH TALLY INDIRECT WORD.
2. USE OPERAND ADDRESS FOR INSTRUCTION EXECUTION.
3. TALLY INDIRECT WORD IS NOT ALTERED IN MEMORY.

INDIRECT VARIATION (I)

FLOWCHART



INDIRECT WORD Formats for IT Modification



ADDRESS TALLY PSEUDO-OPERATIONS

TALLY A,T,C,(TALLY)

USED FOR I, ID, DI, SC AND CI TYPES OF TALLY MODIFICATION, WHERE SC AND CI ARE FOR SIX (6) BIT CHARACTERS.

TALLYB A,T,B,(TALLY BYTE)

USED FOR SC AND CI TYPES OF TALLY MODIFICATION, WHERE NINE (9) BIT BYTES (CHARACTERS) ARE DESIRED.

TALLYD A,T,D,(TALLY AND DELTA)

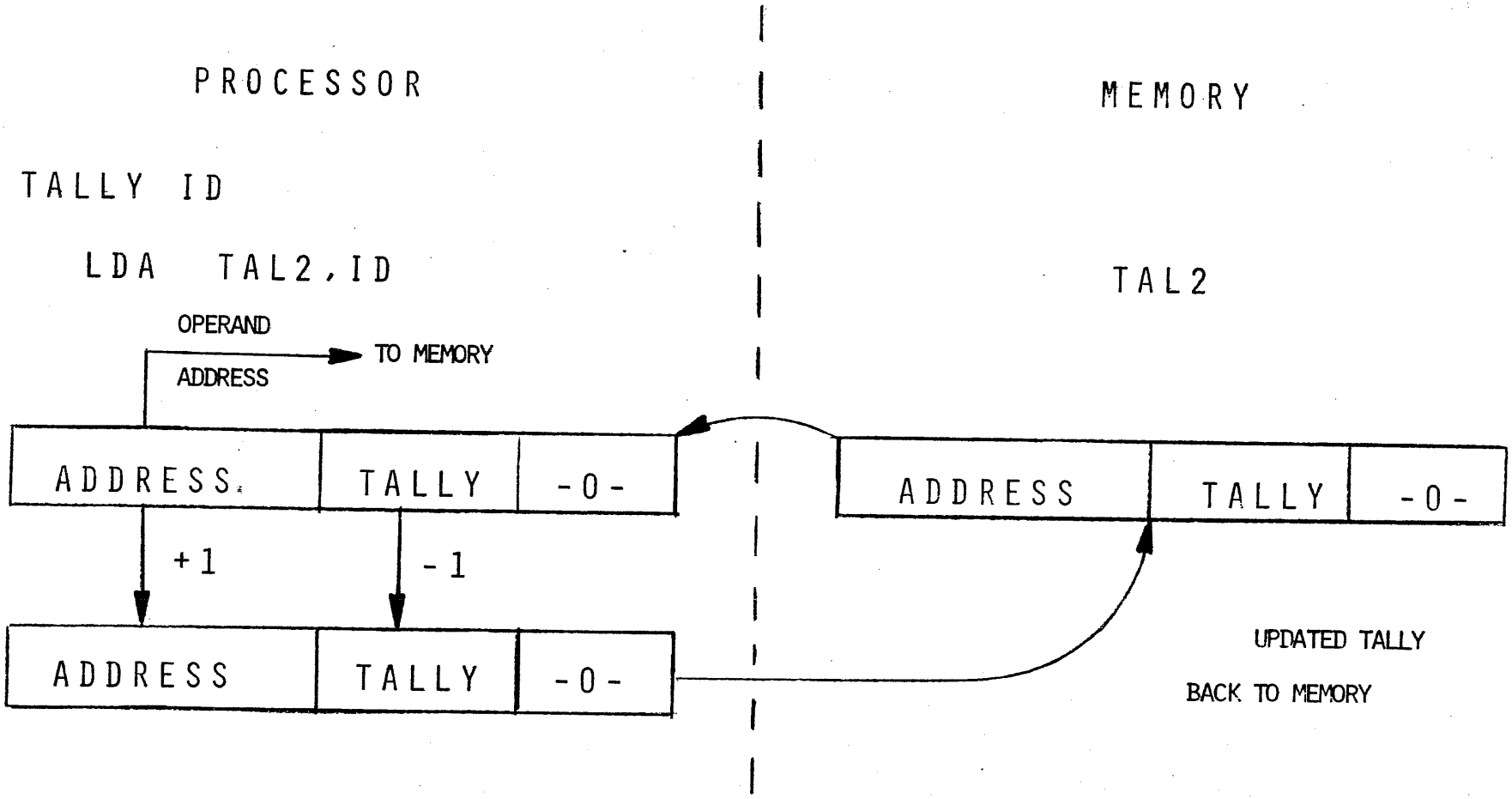
USED FOR AD AND SD TYPES OF TALLY MODIFICATION.

TALLYC A,T,MOD,(TALLY AND CONTINUE)

USED FOR IDC AND DIC TYPES OF TALLY MODIFICATION.

ABBREVIATIONS:

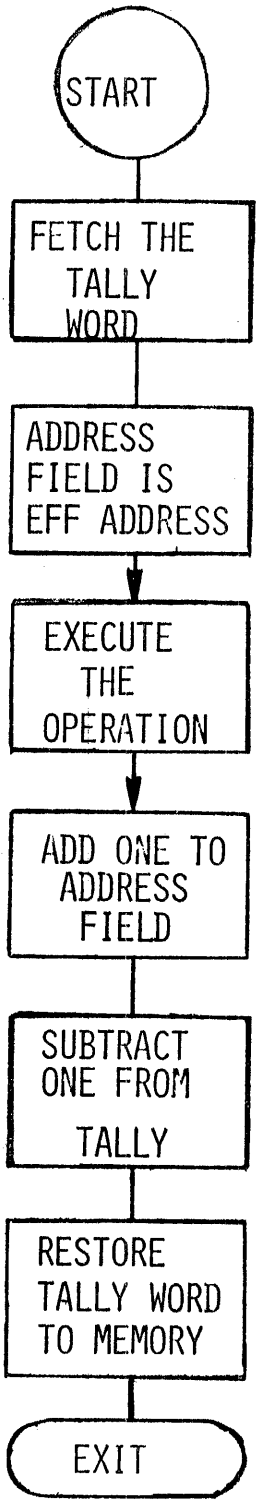
CODE	MEANING
A	ADDRESS
T	TALLY
C	CHARACTER POSITION
B	BYTE
D	DELTA (Δ)
MOD	MODIFIER



1. FETCH TALLY INDIRECT WORD
2. USE OPERAND ADDRESS FOR INSTRUCTION EXECUTION.
3. ADD ONE (1) TO OPERAND ADDRESS AND RESTORE IN TALLY WORD.
4. SUBTRACT ONE (1) FROM TALLY AND RESTORE IN TALLY WORD.
5. RESTORE UPDATED TALLY INDIRECT WORD TO MEMORY.

INCREMENT ADDRESS-DECREMENT TALLY (ID)

FLOWCHART



INDEXING EXAMPLE

PROBLEM: MOVE 50 WORDS OF DATA FROM HERE TO THERE.

```
                        1
1       8               6
        LDX7    50,DU
        LDX6    0,DU
GO      LDA     HERE,6
        STA     THERE,6
        ADX6    1,DU
        SBX7    1,DU
        TNZ     GO
```

INDIRECT THEN TALLY -ID- EXAMPLE

```
        LDA     TAL1,ID
        STA     TAL2,ID
        TTF     *-2
         .
         .
         .
TAL1    TALLY   HERE,50
TAL2    TALLY   THERE,50
```

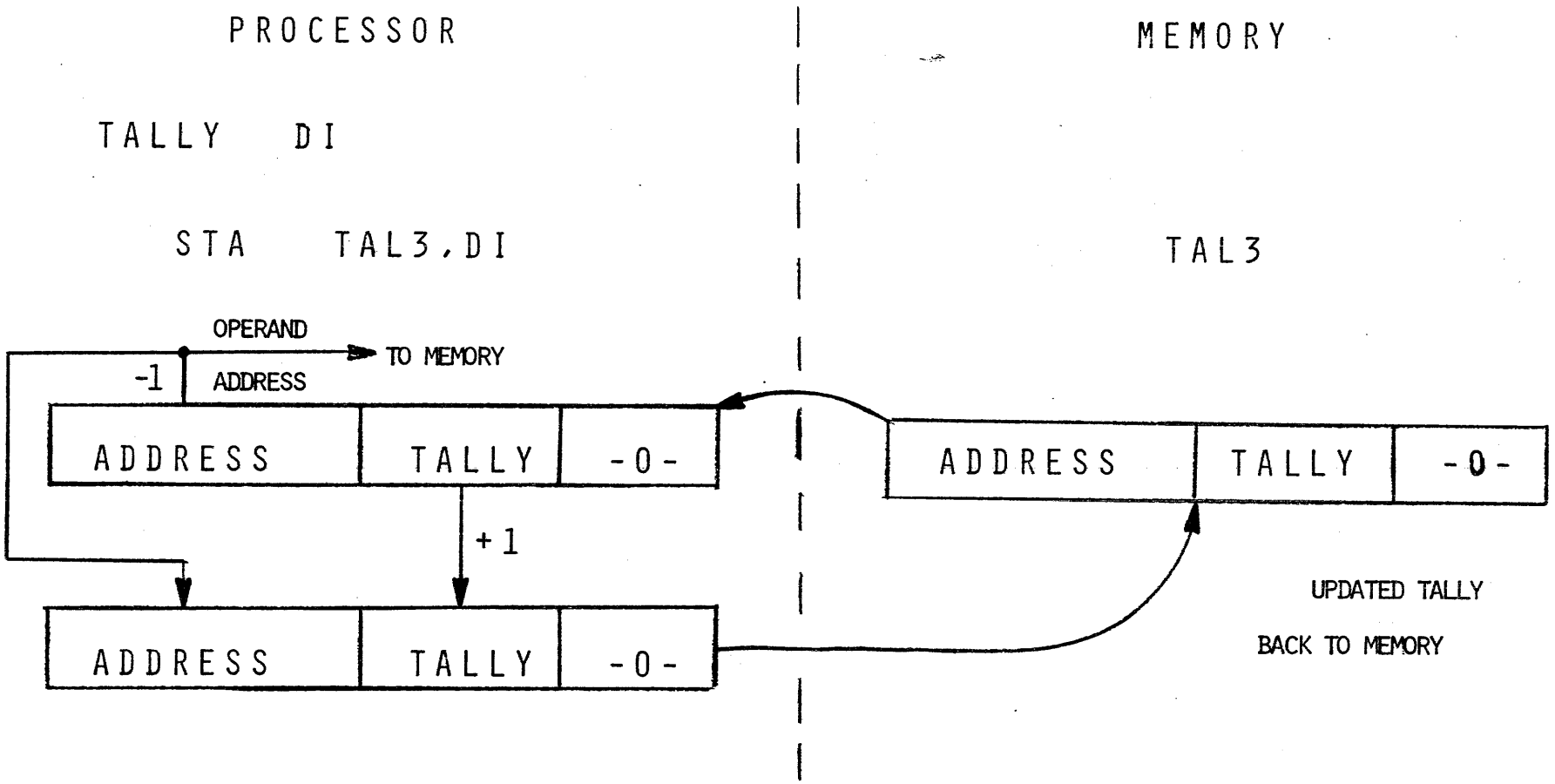
I AND ID EXAMPLE

PROBLEM: SEARCH A 100 WORD TABLE AND COMPLEMENT ALL NEGATIVES FOUND THEREIN.

```
                        1
1       8               6

START   LDA     I       TAL1,I
        TPL             *+2
        NEG
        STA             TAL1,ID
        TTF             START
          .
          .
          .
TAL1    TALLY           TABLE,100
```

TALLY REFRESHING

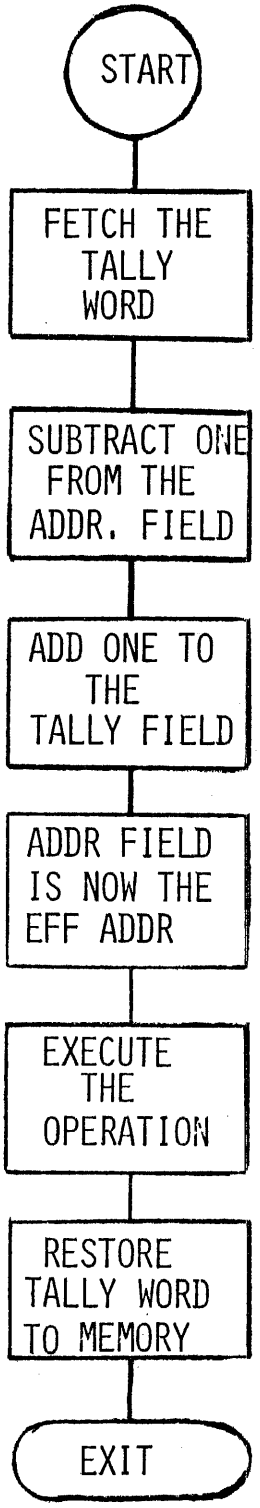
```
                        1
1       8               6

        LDA     TALREF
        STA     TAL1
          .
          .
          .
TALREF  TALLY   TABLE,100
TAL1    BSS     1
```



1. FETCH TALLY INDIRECT WORD
2. SUBTRACT ONE (1) FROM OPERAND ADDRESS AND USE.
3. RESTORE UPDATED ADDRESS IN TALLY WORD.
4. ADD ONE (1) TO TALLY AND RESTORE IN TALLY WORD.
5. RESTORE UPDATED TALLY INDIRECT WORD TO MEMORY.

DECREMENT ADDRESS-INCREMENT TALLY (DI)

FLOWCHART



DECREMENT ADDRESS INCREMENT TALLY -DI- EXAMPLE

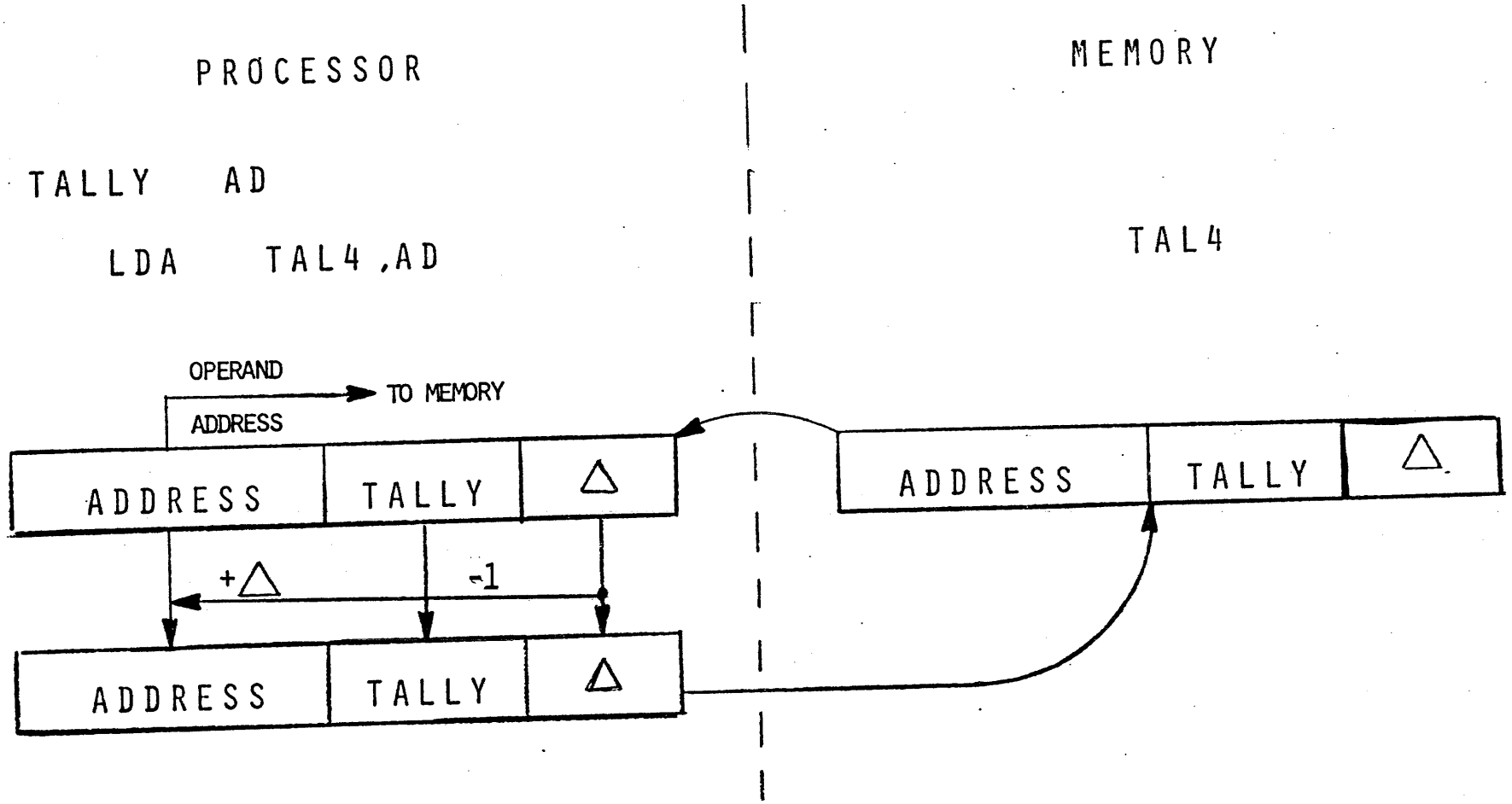
PROBLEM: TRANSFER 500 WORDS OF DATA FROM ONE TABLE TO ANOTHER IN A REVERSE ORDER.

```
                          1
1         8               6

          LDAQ    TALR
          STAQ    TAL1
          LDA     TAL1,ID
          STA     TAL2,DI
          TTF     *-2
            .
            .
            .
TALR      ETALLY  FORWRD,500
           TALLY  REVRSE,-500
TAL1      BSS     1
TAL2      BSS     1
FORWRD    BSS     500
REVRSE    BFS     500
```

I, ID AND DI EXAMPLE

PROBLEM: REVERSE THE ORDER OF THE WORDS OF A 500 WORD TABLE WITHIN ITSELF.

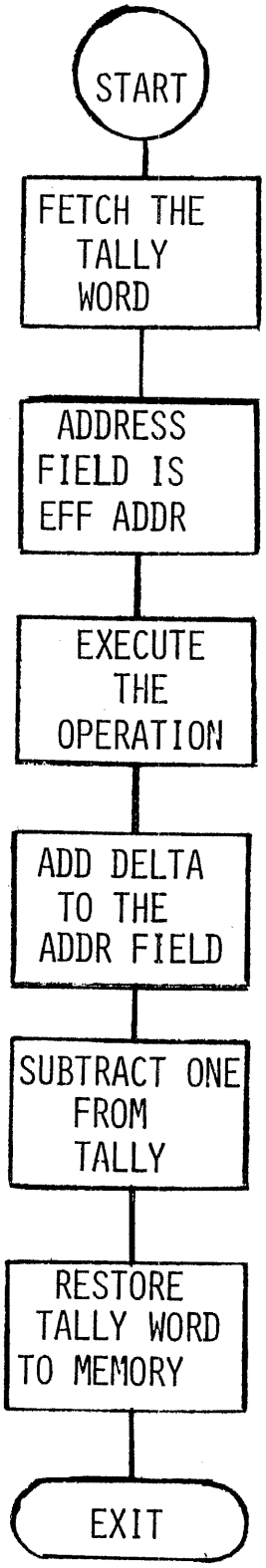
```
                                 1
1                8               6

                 .
                 .
                 .
GOOD             LDA             TAL1,I
                 LDQ             TAL2,DI
                 STA             TAL2,I
                 STQ             TAL1,ID
                 TTF             GOOD
                 .
                 .
                 .
TAL1             TALLY           TABLE,250
TAL2             TALLY           TABLE+500,-250
TABLE            BSS             500
```

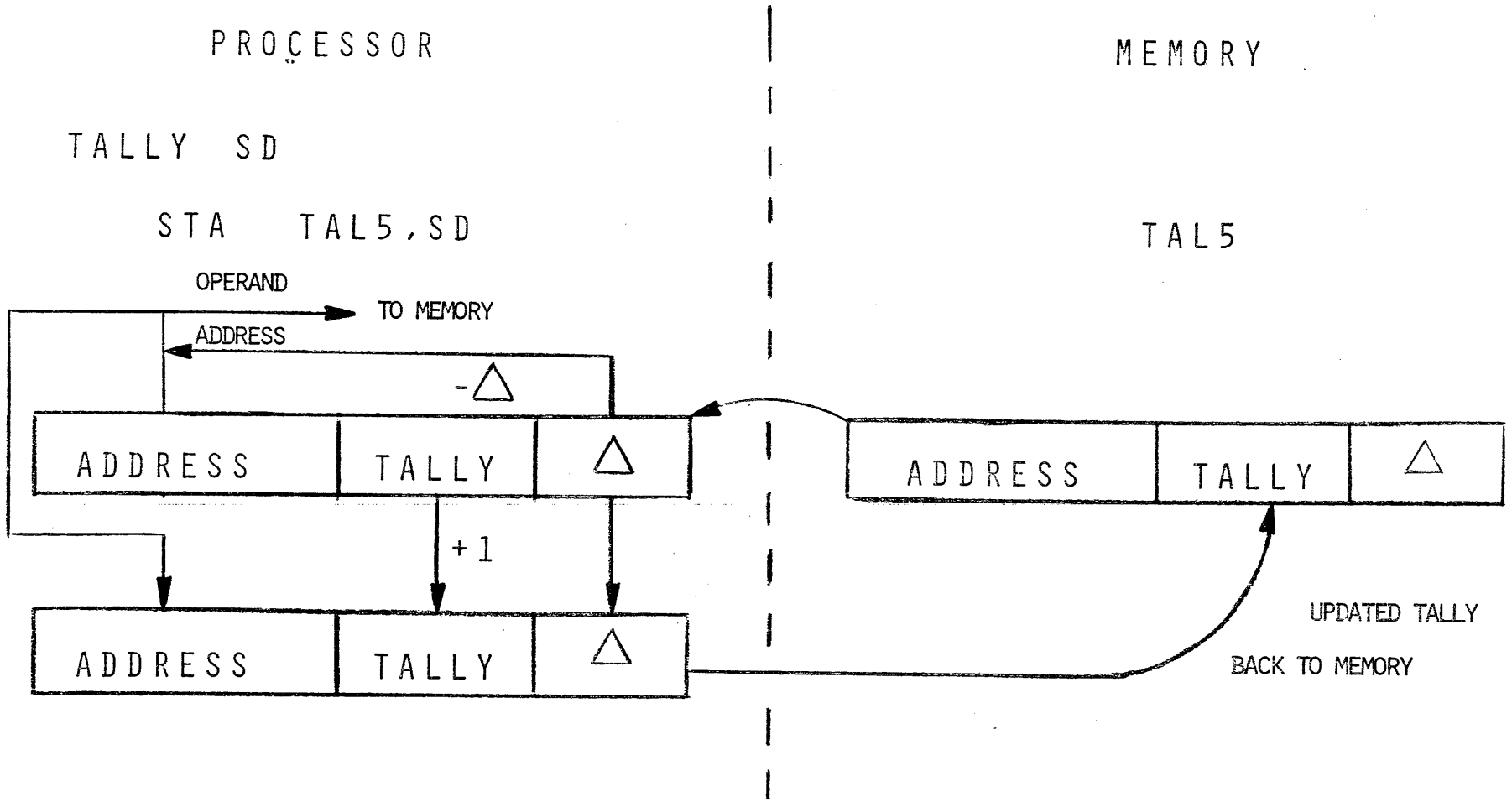


1. FETCH TALLY INDIRECT WORD
2. USE OPERAND ADDRESS FOR INSTRUCTION EXECUTION.
3. ADD DELTA TO OPERAND ADDRESS AND RESTORE IN TALLY WORD.
4. SUBTRACT ONE (1) FROM TALLY AND RESTORE IN TALLY WORD.
5. RESTORE UPDATED TALLY INDIRECT WORD TO MEMORY.

ADD DELTA VARIATION (AD)

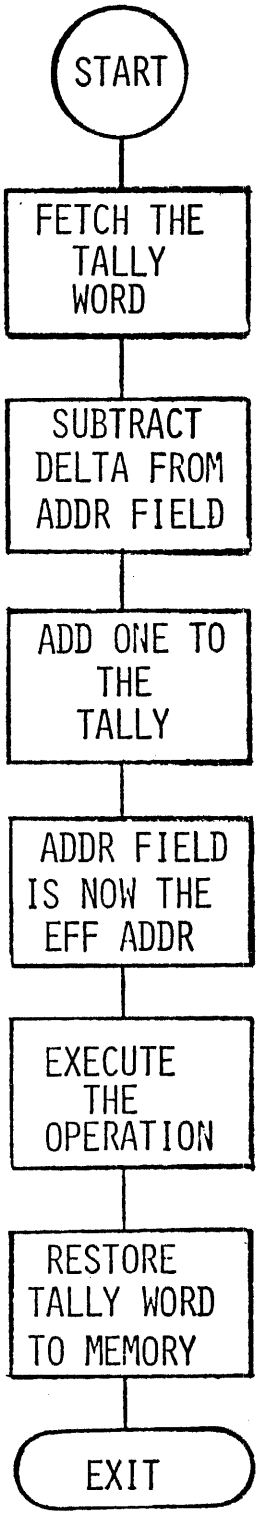
FLOWCHART





1. FETCH TALLY INDIRECT WORD
2. SUBTRACT DELTA FROM OPERAND ADDRESS AND USE.
3. RESTORE UPDATED ADDRESS IN TALLY WORD.
4. ADD ONE (1) TO TALLY AND RESTORE IN TALLY WORD.
5. RESTORE UPDATED TALLY INDIRECT WORD TO MEMORY.

SUBTRACT DELTA VARIATION (SD)
FLOWCHART



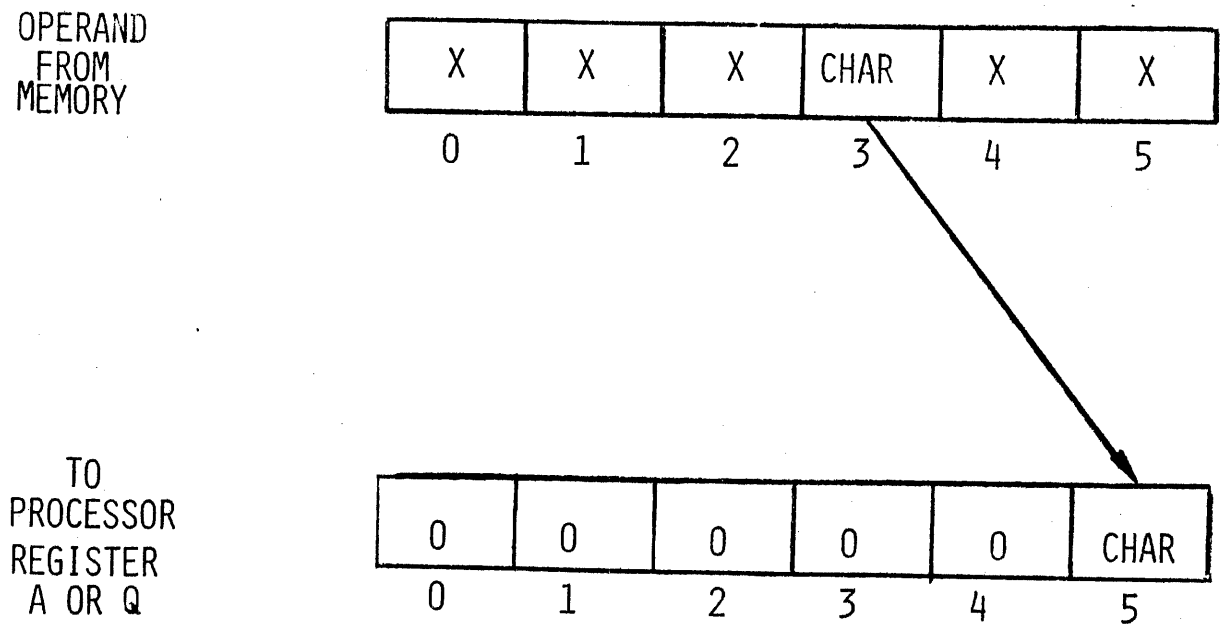
ADD DELTA -AD- AND SUBTRACT DELTA -SD-

PROBLEM: REVERSE THE ORDER OF ONE TABLE INTO ANOTHER USING DOUBLE PRECISION COMMANDS.

```
                              1
1         8                   6

                .
                .
                .
BEGIN     LDAQ      TAL1,AD
          STAQ      TAL2,SD
          TTF       BEGIN
                .
                .
                .
TAL1      TALLYD    TABLEA,4096,2
TAL2      TALLYD    TABLEB,,2
TABLEA    BSS       8192
TABLEB    BFS       8192
```

CHARACTER OPERATIONS

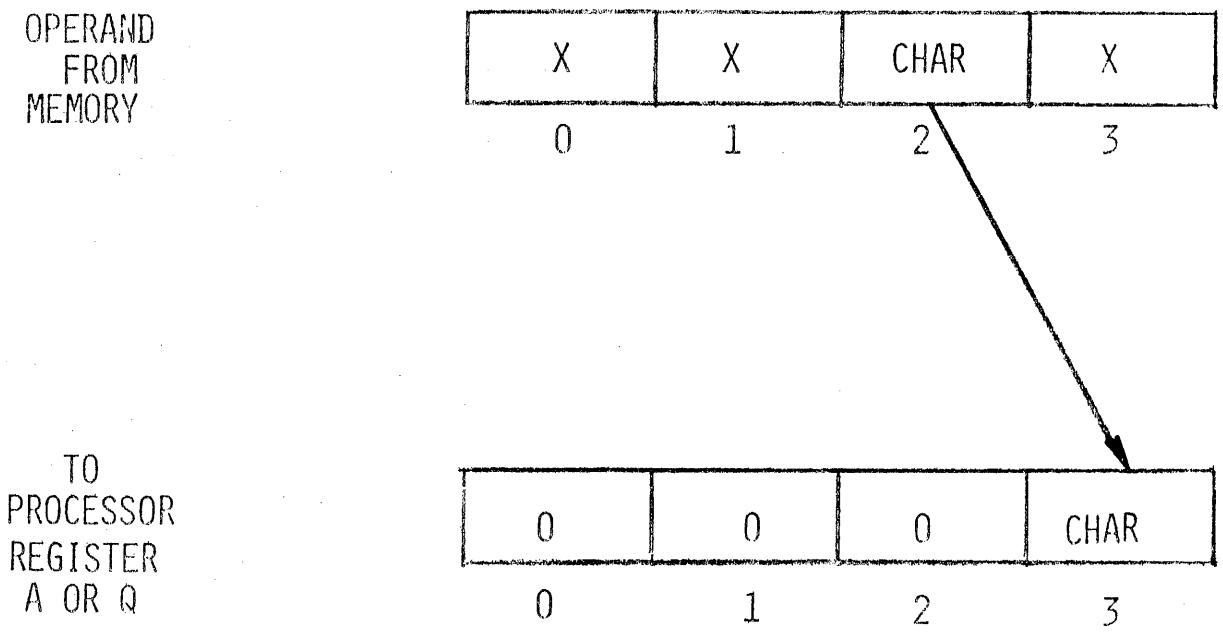
OPERAND FROM MEMORY
SIX-BIT CHARACTER



CHARACTER OPERATIONS

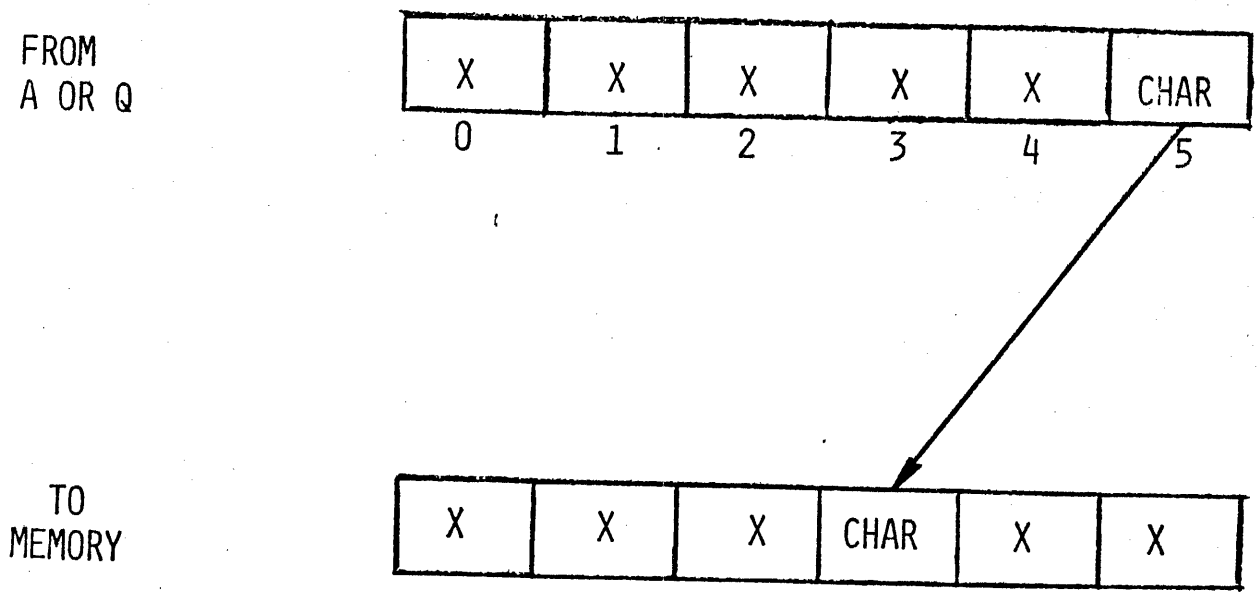
OPERAND FROM MEMORY

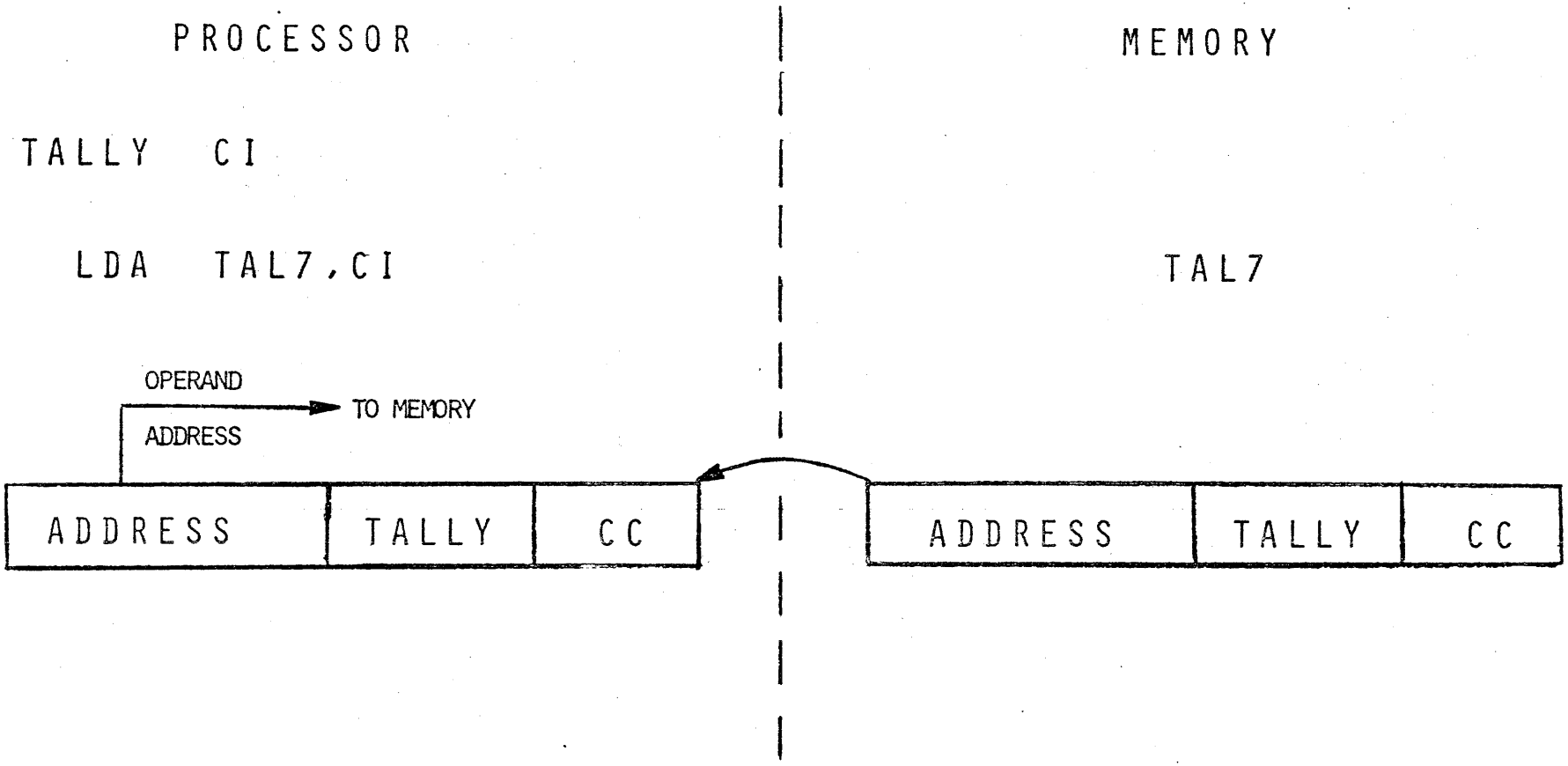
NINE-BIT CHARACTER



CHARACTER OPERATIONS

SIX-BIT CHARACTER



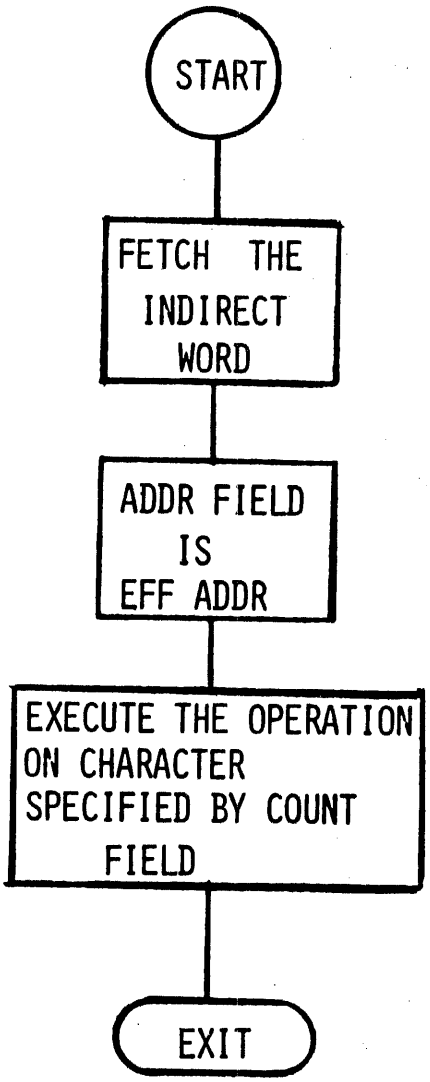


NOTE:

CI TALLY INDIRECT WORD NOT ALTERED IN MEMORY.

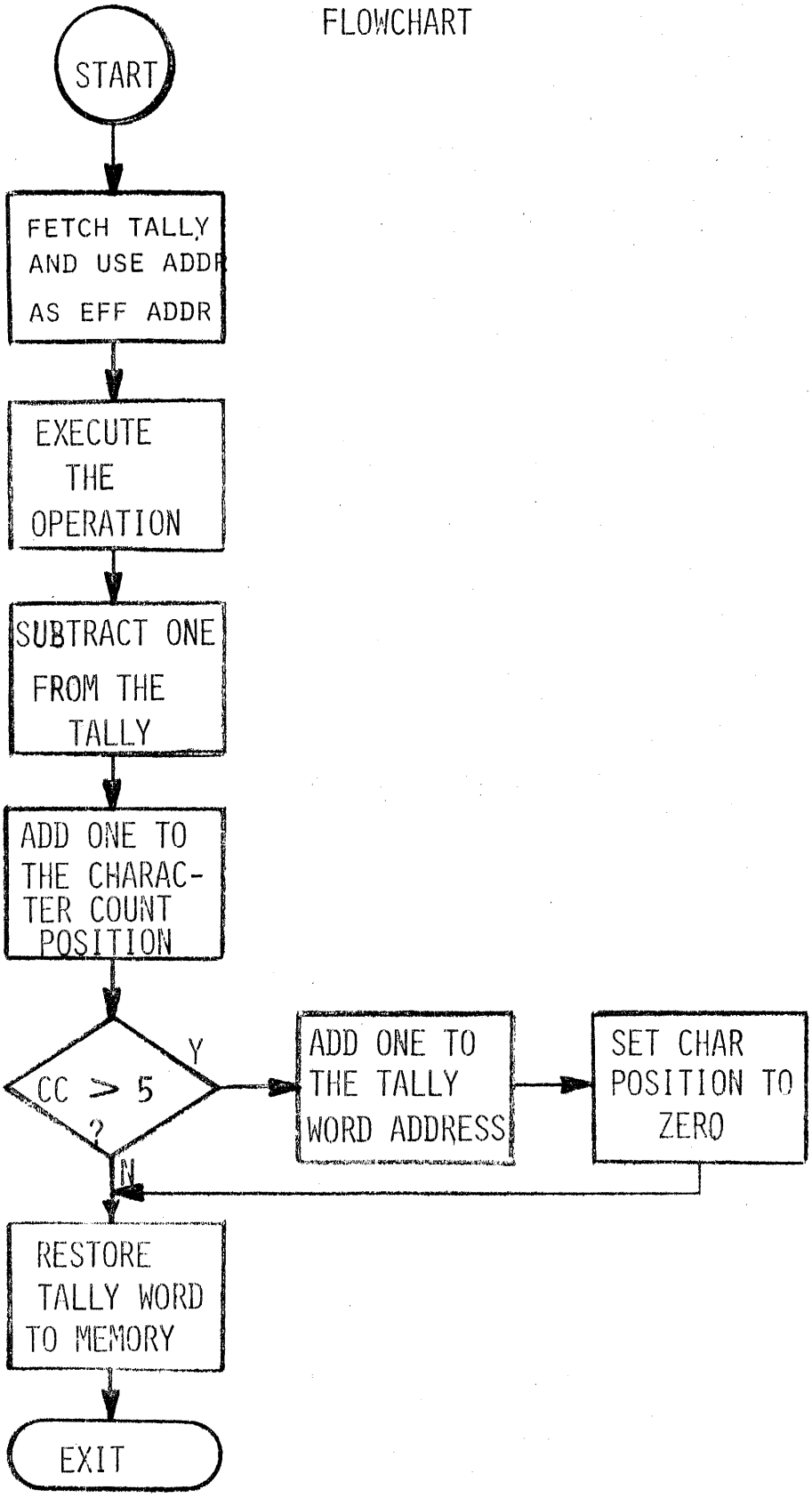
CHARACTER FROM INDIRECT VARIATION (CI)

FLOWCHART

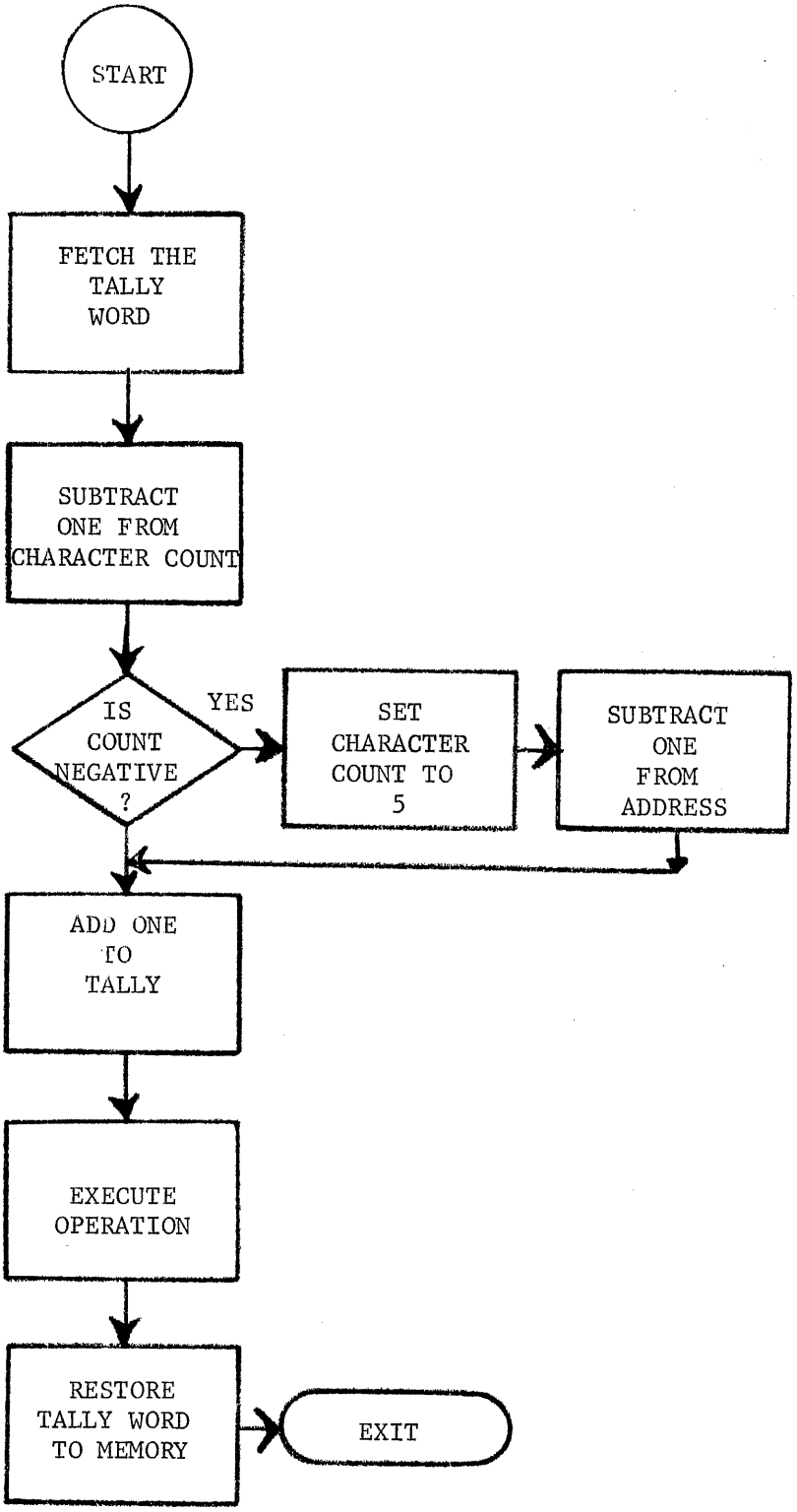


SEQUENCE CHARACTER VARIATION (SC)

FLOWCHART



SEQUENCE CHARACTER REVERSE (SCR) FLOWCHART



REVERSING TABLE WITHIN ITSELF

CHARACTER BY CHARACTER

```
        LDA     TAL1,SCR
        LDQ     TAL2,CI
        STQ     TAL1,CI
        STA     TAL2,SC
        TTF     *-4
         .
         .
         .

TAL1    TALLY   TABLE+200,0,0
TAL2    TALLY   TABLE,600,0
TABLE   BSS     200
```

-CI- AND -SC-

PROBLEM: GIVEN

DOUBLE-WORD XXXXXX XX.XXX

INSERT *'S IN PLACE OF LEADING ZEROS

```
                        1
1       8               6

GO      LDQ             =3HØØ*,DL
        LDA             TALI,CI
        TNZ             *+3
        STQ'            TALI,SC
        TRA             GO+1
          .
          .
TALI    TALLY           WORD,,0
```

PROBLEM:

GIVEN
DOUBLE-WORD ␣XXXXX XXXXXX
INSERT DEC PT AT IMPLIED LOC

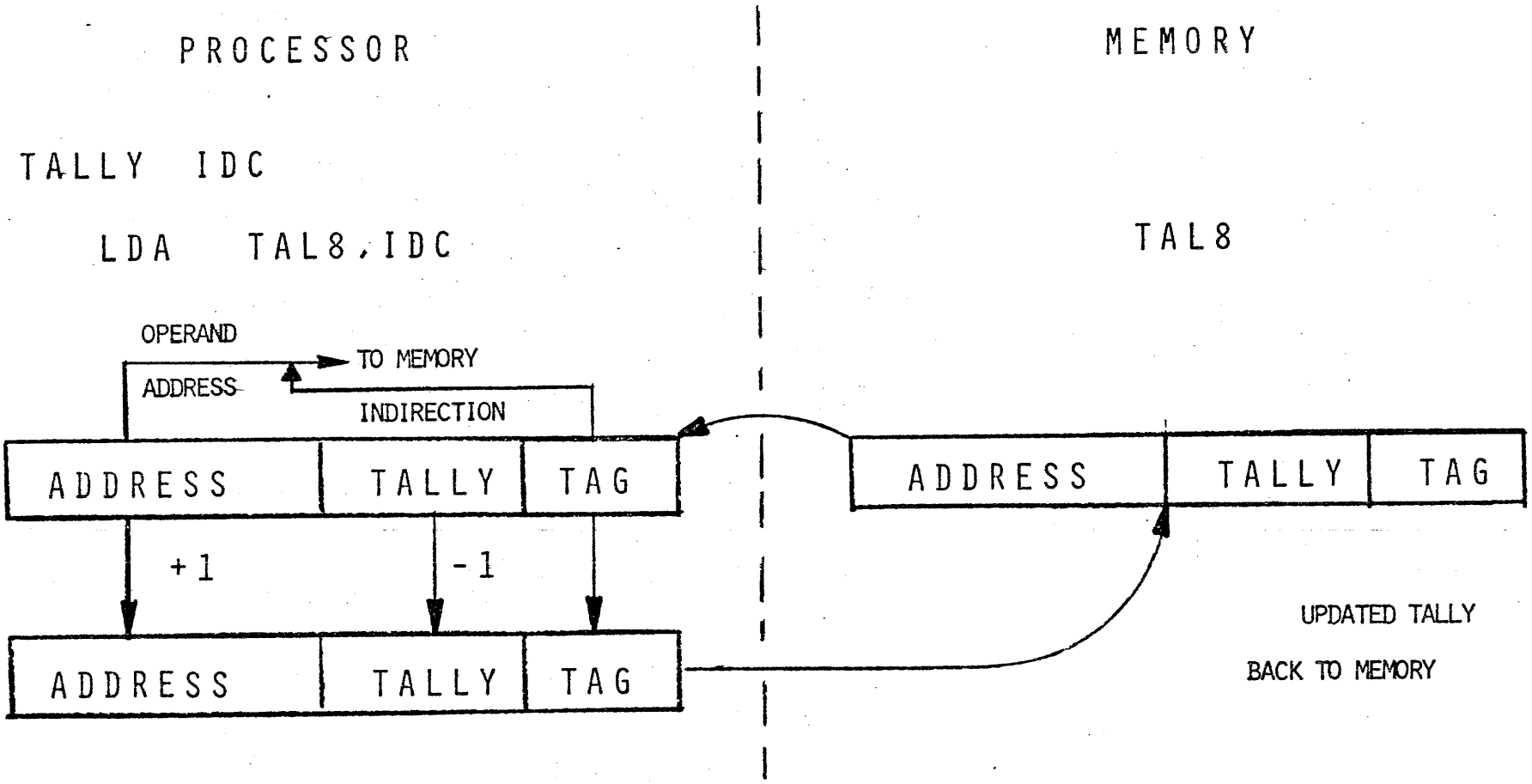
```
          .
          .
          .
        LDQ     =3H␣␣.,DL
        LDA     TAL1,SC
        STA     TAL2,SC
        TTF     *-2
        STQ     TAL2,CI
          .
          .
          .
TAL1    TALLY   WORD,8,1
TAL2    TALLY   WORD,8,0
```

CHECK PROTECTION AND DECIMAL INSERTION

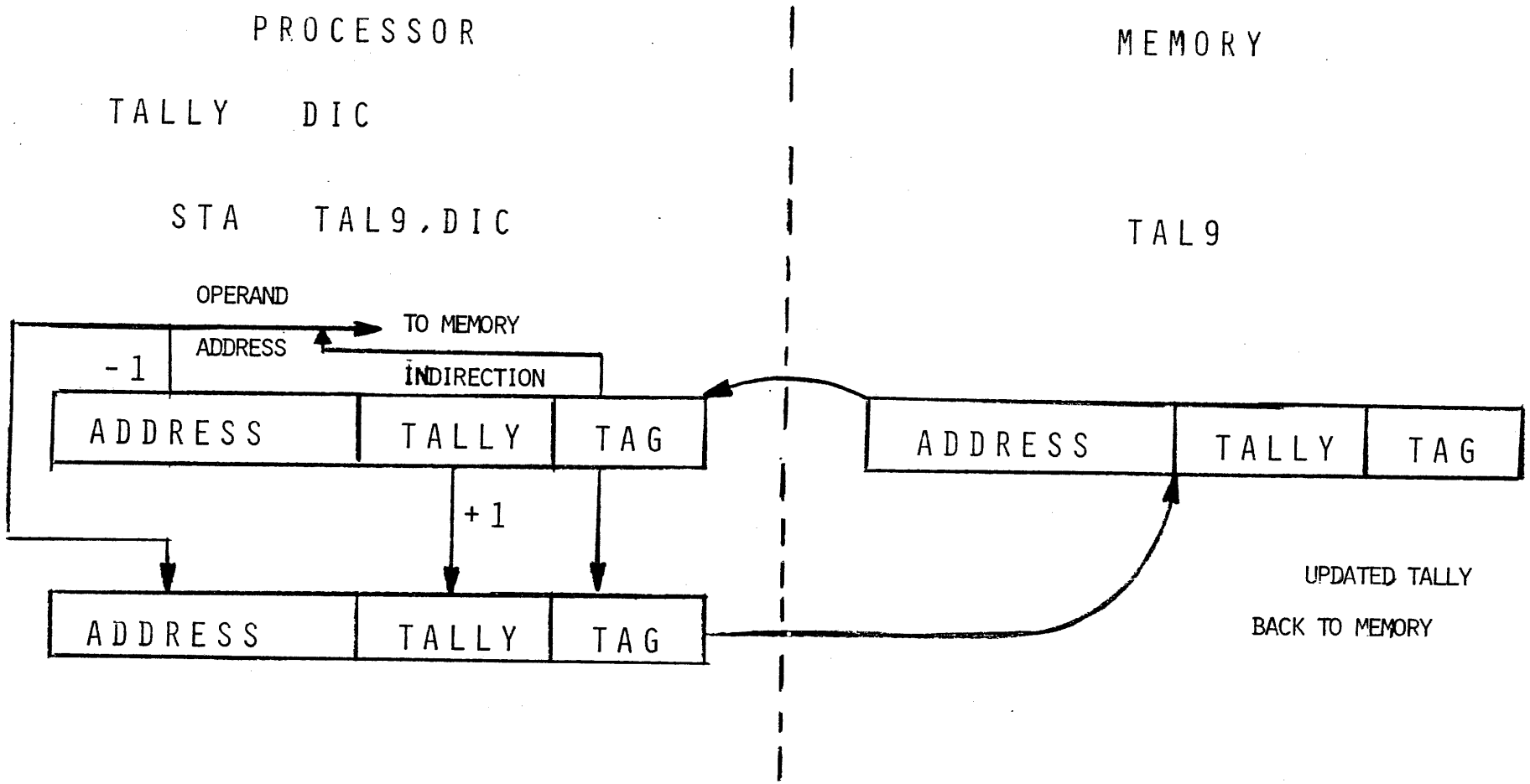
EXAMPLES:

```
BEFORE    ØOOOXX XXXXXX  AFTER ***XXX XX.XXX
BEFORE    ØOOOOO OOOXXX  AFTER ****** **.XXX
```

```
        1       8         16

                LDQ       =3HØ.*,DL
                LDA       TAL1,SC
                TNZ       CHAR
                STQ       TAL2,SC
                TTF       -3,IC
                TRA       INSERT
                LDA       TAL1,SC
        CHAR    STA       TAL2,SC
                TTF       *-2
        INSERT  QRS       6
                STQ       TAL2,CI
                  .
                  .
        TAL1    TALLY     WORD,8,1
        TAL2    TALLY     WORD,8,0
```



1. FETCH TALLY INDIRECT WORD
2. TRAP OPERAND ADDRESS AND MODIFICATION FOR INSTRUCTION EXECUTION.
3. ADD ONE (1) TO OPERAND ADDRESS AND RESTORE IN TALLY WORD.
4. SUBTRACT ONE (1) FROM TALLY AND RESTORE IN TALLY WORD.
5. RESTORE UPDATED TALLY INDIRECT WORD TO MEMORY.
6. USE TRAPPED ADDRESS FOR FURTHER INDIRECTION IF INDICATED BY THE TAG FIELD.



1. FETCH TALLY INDIRECT WORD
2. SUBTRACT ONE (1) AND TRAP OPERAND ADDRESS AND MODIFICATION.
3. RESTORE UPDATED OPERAND ADDRESS IN TALLY WORD.
4. ADD ONE (1) TO TALLY AND RESTORE IN TALLY WORD.
5. RESTORE UPDATED TALLY INDIRECT WORD TO MEMORY.
6. USE TRAPPED ADDRESS FOR FURTHER INDIRECTION IF INDICATED BY THE TAG FIELD.

REPEAT INSTRUCTION

INSTRUCTION FORMAT

TALLY 1-256		EXIT CONDITIONS	OP CODE	I	DELTA 0-63

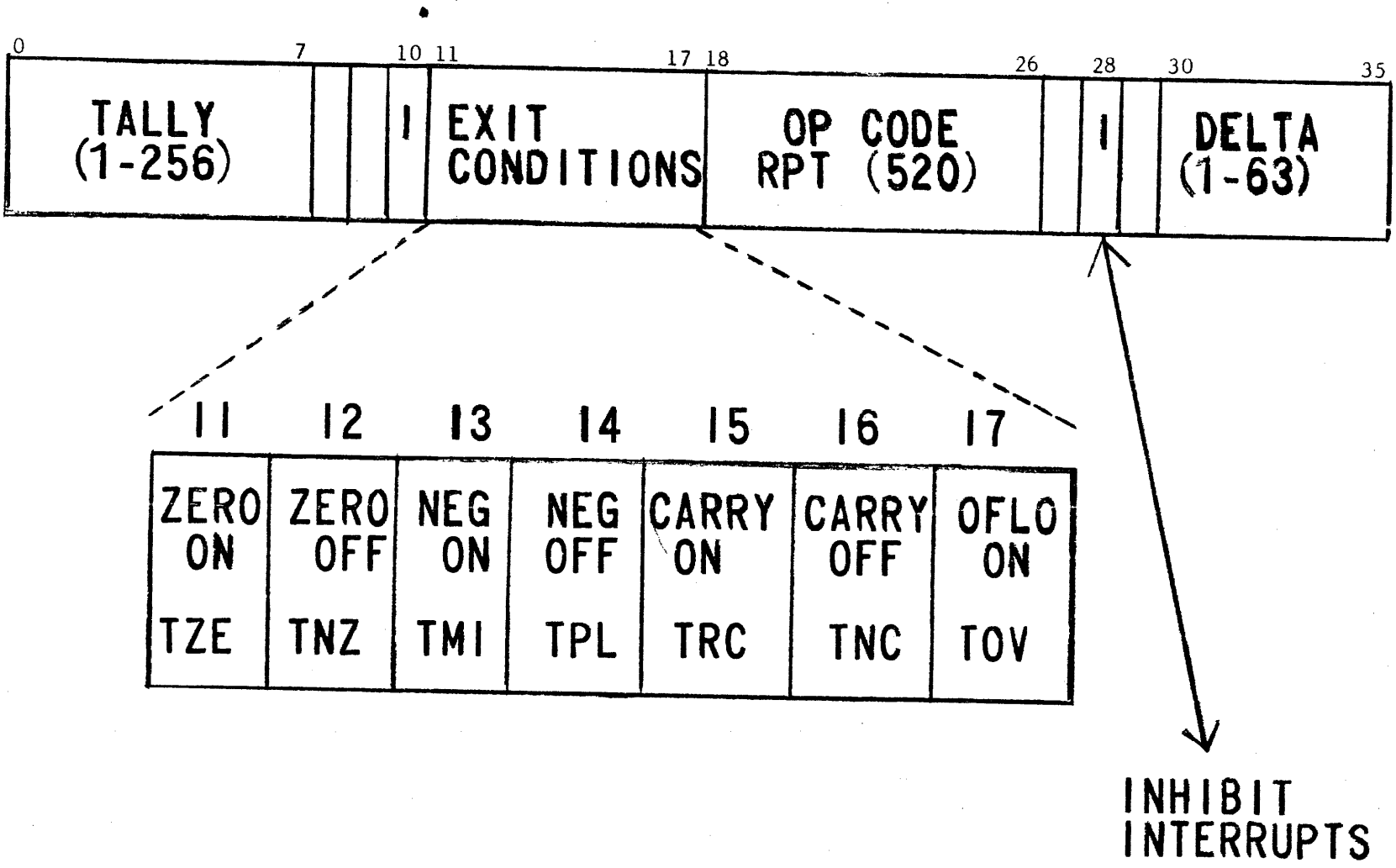
CODING FORMAT

1	8	16
	RPT	TALLY,DELTA,EXIT CONDITIONS

SUMMARY

1. REPEAT NEXT INSTRUCTION N TIMES.
2. INCREMENT OPERAND ADDRESS BY DELTA.
3. EXIT PRIOR TO TALLY RUNOUT IF EXIT CONDITIONS MET.

REPEAT INSTRUCTION FORMAT



REPEAT INSTRUCTION

RULES:

REPEATED INSTRUCTION MUST BE MODIFIED BY INDEX REGISTER.

ONLY X1 THRU X7 MAY BE USED.

TALLY, EXIT CONDITIONS AUTOMATICALLY PLACED IN XO.

TALLY DECREMENTED IN XO.

CODING EXAMPLE:

```
1         8         16
          LDA       KEY
          EAX4      TABLE
          RPT       64,1,TZE
          CMPA      0,4
          TZE       FOUND
            .
            .
            .
TABLE     BSS       64
KEY       BSS       1
```

ADDRESS MODIFICATION
OF
REPEATED INSTRUCTION

FIRST TIME:

EFFECTIVE ADDRESS = TA + $C(X_N)$

$$EA + \Delta \Rightarrow C(X_N)$$

N TIMES:

EFFECTIVE ADDRESS = $C(X_N)$

$$EA + \Delta \Rightarrow C(X_N)$$

TA = TENTATIVE ADDRESS

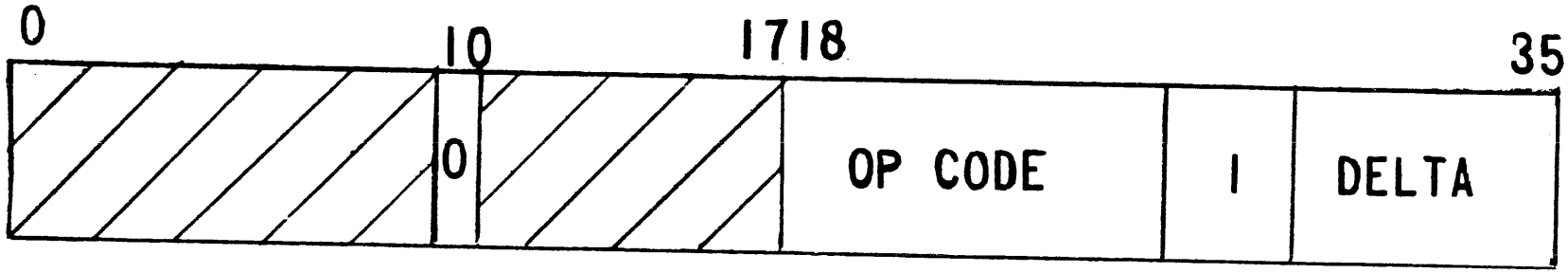
EA = EFFECTIVE ADDRESS

CODING EXAMPLE
USING
REPEAT INSTRUCTION

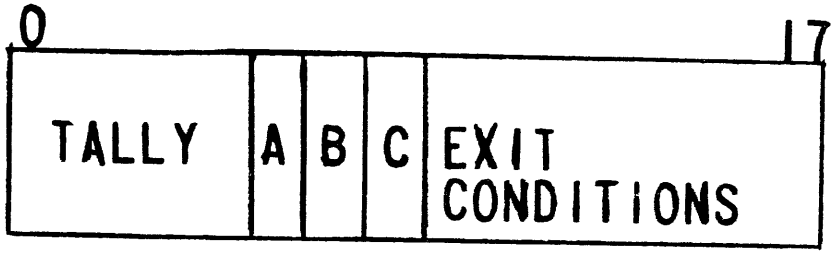
PROBLEM: SEARCH A TABLE OF 64 WORDS FOR NEGATIVE VALUES. STORE ZEROS IN EVERY NEGATIVE WORD.

```
GO      EAX7    TABLE
        RPT     64,1,TMI
        LDA     0,7
        TPL     DØNE
        STZ     -1,7
        TRA     GO
DONE    XXX
        .
        .
TABLE   BSS     64
```

RPTX - RPDX



PROGRAMMER LOADS XO:



RPTX DELTA RPDX ,DELTA

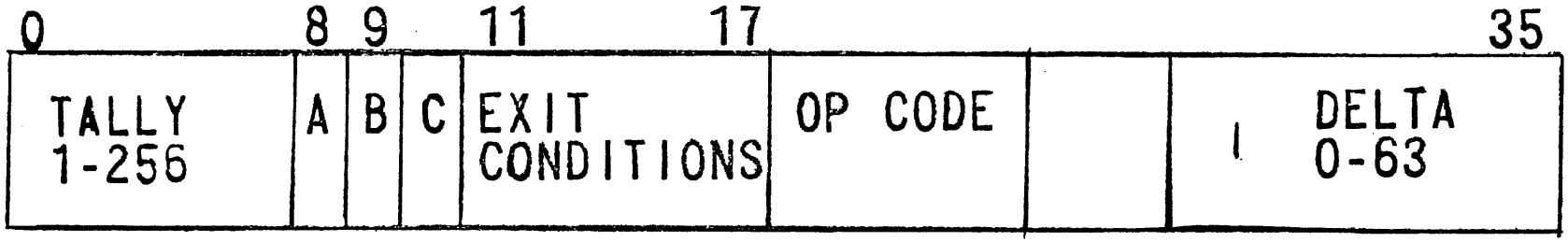
CODING EXAMPLES

```
GO      EAX7    TABLE
        RPT     64,1,TMI
        LDA     0,7
        TPL     DONE
        STZ     -1,7
        TRA     GO
```

```
        EAX7    TABLE
        RPT     64,1,TMI
        LDA     ,7
        TPL     DONE
BACK    STZ     -1,7
        CANXO   =Ø776000,DU
        TZE     DONE
        RPTX    ,1
        LDA     ,7
        TMI     BACK
DONE      .
          .
```

```
        EAX7    TABLE
        LDXO    GO
GO      RPTX    64,1,TMI
        LDA     0,7
        TPL     DONE
        STZ     -1,7
        CANXO   =Ø776000,DU
        TNZ     GO
```

REPEAT DOUBLE FORMAT



RPD RPDA RPDB

RPDA ADD DELTA ONLY TO FIRST REPEATED INSTRUCTION

RPDB ADD DELTA ONLY TO SECOND REPEATED INSTRUCTION

REPEAT DOUBLE EXAMPLES

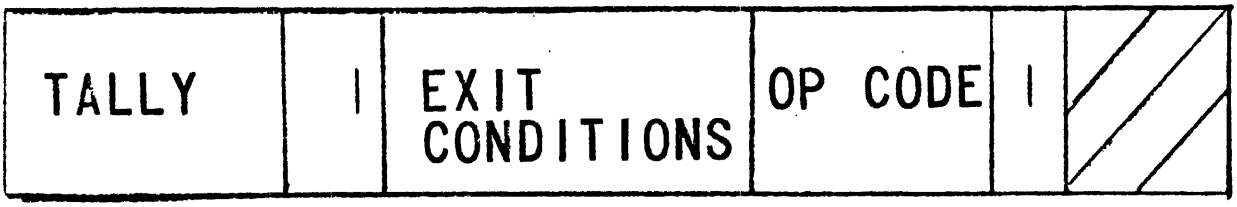
```
        EAX6    FROM
        EAX7    TO
        RPD     100,2
        LDAQ    0,6
        STAQ    0,7
          .
          .
          .
FROM    BSS     200
TO      BSS     200
```

SAME INDEX REGISTER MAY NOT BE USED FOR BOTH INSTRUCTIONS

```
        LDI     ,DL
        LDA     CARDIN
        EAX2    CARDIN+2
        EAX3    0
        RPDA    22,1
        ADLA    0,2
        AWCA    =0,3
        CMPA    CARDIN+1
        TNZ     ERROR
```

REPEAT LINK (RPL)

INSTRUCTION FORMAT:



CODING FORMAT:

RPL TALLY,EXIT CONDITIONS

ADDRESS MODIFICATION:

FIRST TIME

EFFECTIVE ADDRESS = TA + C(Xn)

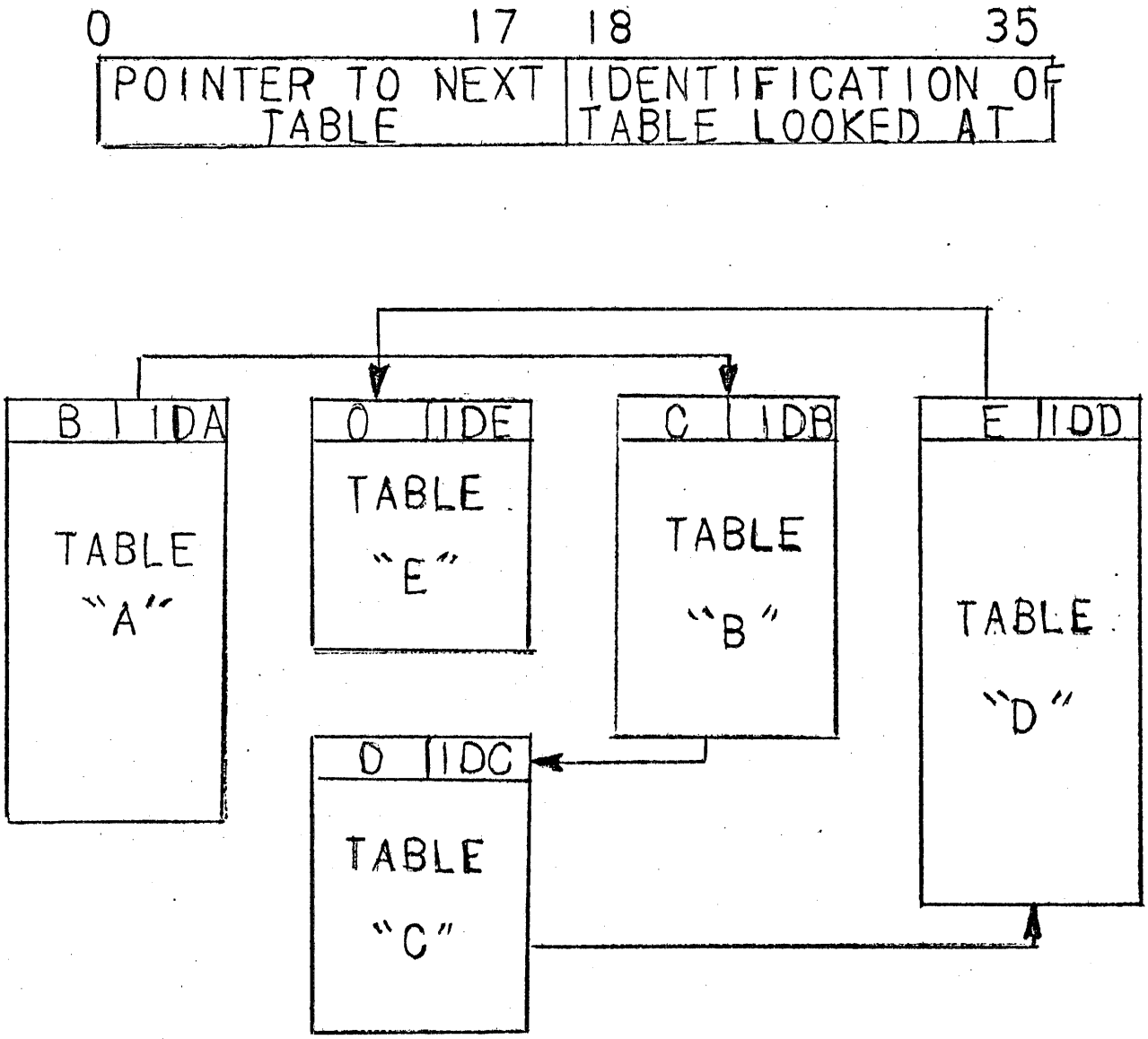
EA ⇒ C(Xn)

N TIMES

NEW EFFECTIVE ADDRESS = $C(EA_{0...17})$

$C(NEA_{0...17}) \Rightarrow C(Xn)$

KEYWORD

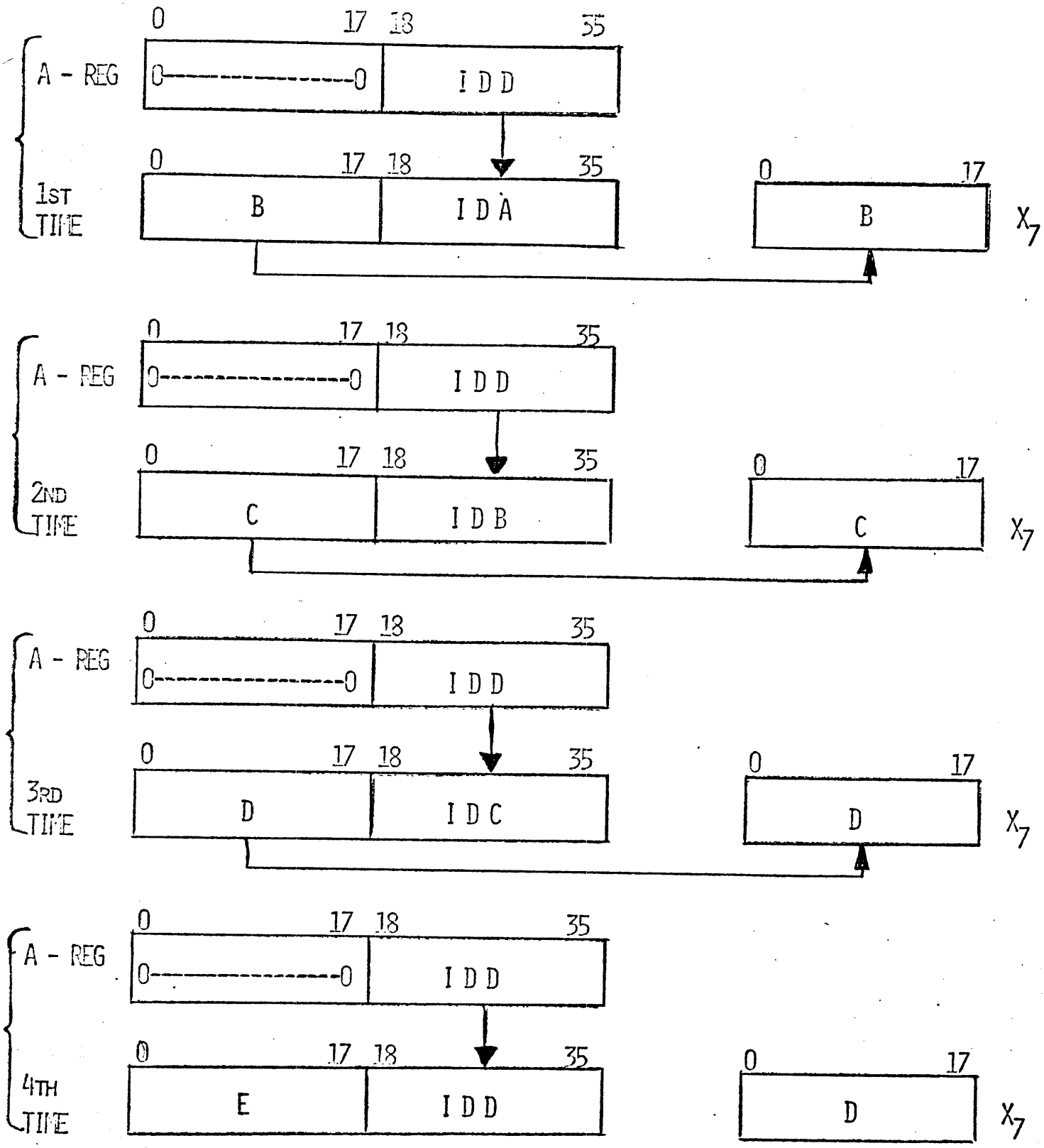

0
17
18
35
POINTER TO NEXT TABLE
IDENTIFICATION OF TABLE LOOKED AT
B
IDA
TABLE "A"
0
IDE
TABLE "E"
C
IDB
TABLE "B"
E
IDD
TABLE "D"
0
IDC
TABLE "C"

RPL CODING

EXAMPLE

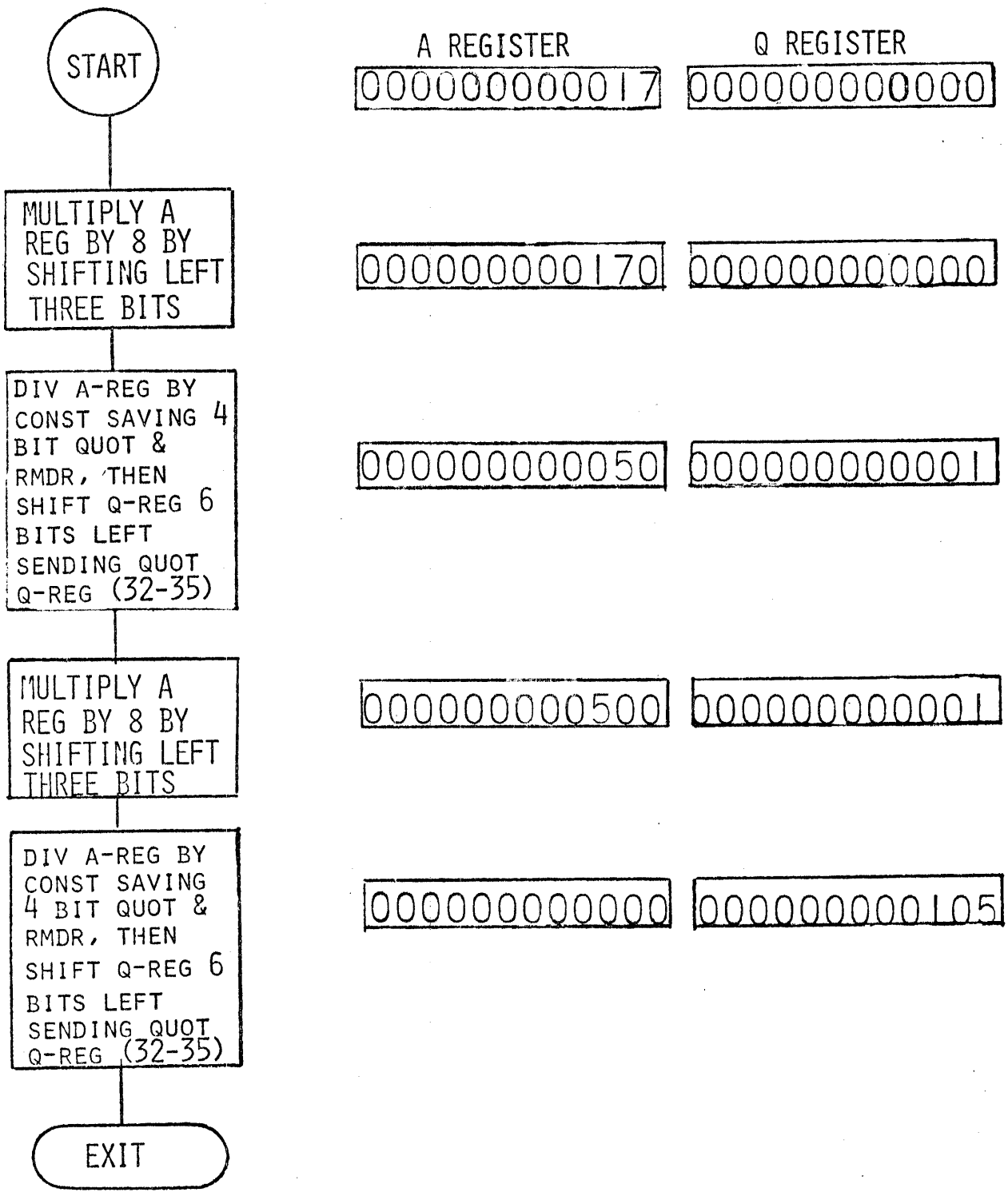
```
EAX7    A
LCQ     =1,DU
LDA     =3HIDD,DL
RPL     5,TZE
CMK     0,7
TNZ     ERROR
  .
  .
  .
```

EXECUTION OF RPL



BCD INSTRUCTION FLOWCHART

```
LDA   =Ø17
EAQ   0
BCD   =80
BCD   =64
```

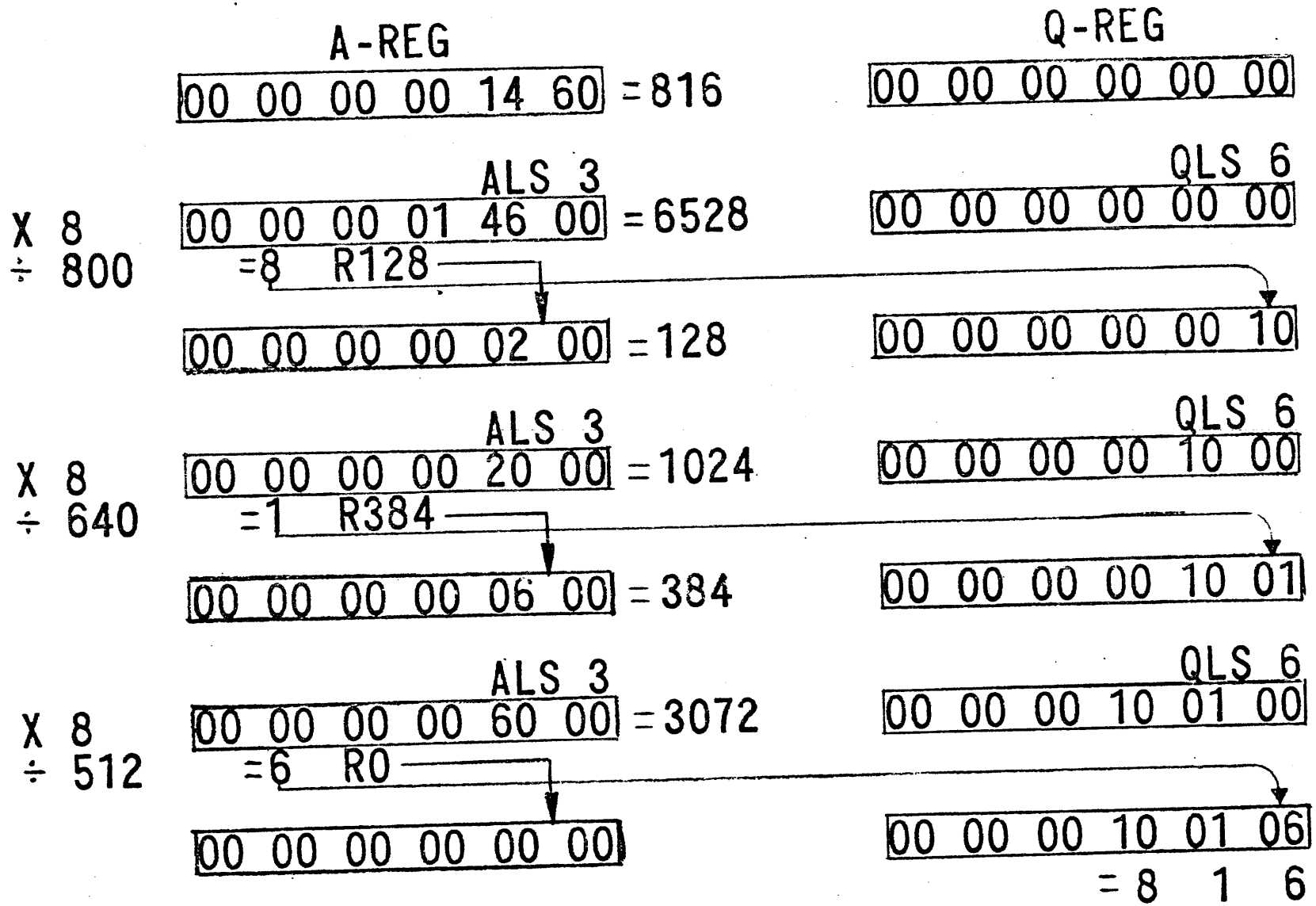


BINARY TO BCD CONVERSION CONSTANTS

CONVERSION STEP	STARTING RANGE OF C(AR) $-10^{10}+1$ $10^{10}-1$	$-10^{9}+1$ $10^{9}-1$	$-10^{8}+1$ $10^{8}-1$	$-10^{7}+1$ $10^{7}-1$	$-10^{6}+1$ $10^{6}-1$	$-10^{5}+1$ $10^{5}-1$	$-10^{4}+1$ $10^{4}-1$	$-10^{3}+1$ $10^{3}-1$	$-10^{2}+1$ $10^{2}-1$	$-10^{1}+1$ $10^{1}-1$
1	$8^1 \times 10^9$	8×10^8	8×10^7	8×10^6	8×10^5	8×10^4	8×10^3	8×10^2	8×10^1	8
2	$8^2 \times 10^8$	$8^2 \times 10^7$	$8^2 \times 10^6$	$8^2 \times 10^5$	$8^2 \times 10^4$	$8^2 \times 10^3$	$8^2 \times 10^2$	$8^2 \times 10^1$	8^2	
3	$8^3 \times 10^7$	$8^3 \times 10^6$	$8^3 \times 10^5$	$8^3 \times 10^4$	$8^3 \times 10^3$	$8^3 \times 10^2$	$8^3 \times 10^1$	8^3		
4	$8^4 \times 10^6$	$8^4 \times 10^5$	$8^4 \times 10^4$	$8^4 \times 10^3$	$8^4 \times 10^2$	$8^4 \times 10^1$	8^4			
5	$8^5 \times 10^5$	$8^5 \times 10^4$	$8^5 \times 10^3$	$8^5 \times 10^2$	$8^5 \times 10^1$	8^5				
6	$8^6 \times 10^4$	$8^6 \times 10^3$	$8^6 \times 10^2$	$8^6 \times 10^1$	8^6					
7	$8^7 \times 10^3$	$8^7 \times 10^2$	$8^7 \times 10^1$	8^7						
8	$8^8 \times 10^2$	$8^8 \times 10^1$	8^8							
9	$8^9 \times 10^1$	8^9								
10	8^{10}									

BCD INSTRUCTION

CONVERT 1460 OCTAL TO 816 DECIMAL



BCD INSTRUCTION

CONVERTING VALUES TO 999999

```
        EAX2    0           PLACE ZEROS IN X2
        LDA     X           LOAD ACCUMULATOR WITH VALUE TO BE CONVERTED
        RPT     6,1         REPEAT 6 TIMES, INCREMENT BY 1
        BCD     TAB,2       DIVIDE BY TAB, TAB+1, ETC.
        STQ     Y           STORE CONVERTED NUMBER IN Y
                .
                .
                .
TAB     DEC     300000, 640000, 512000, 409600, 327680
        DEC     262144
```

BCD EXAMPLE
CONVERTING MAXIMUM VALUES

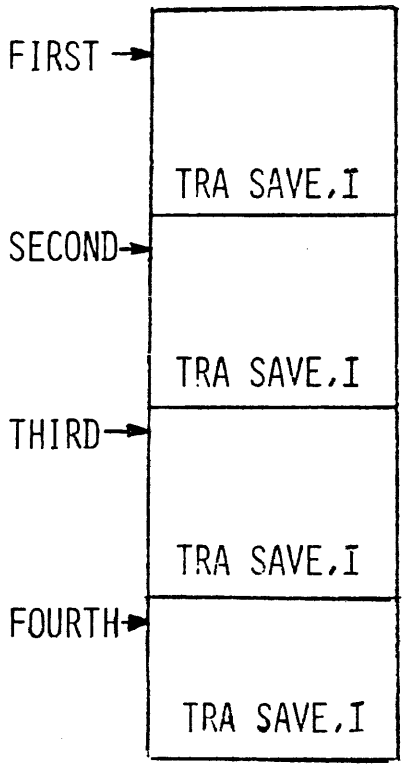
```
          EAX7    TABLE          PLACE ADDRESS OF TABLE IN X7
          EAQ     0              ZERO OUT QR
          LDA     VALUE          BINARY VALUE TO BE CONVERTED
          RPT     4,1            CONVERT FIRST 4 DIGITS
          BCD     0,7            STEP THRU TABLE
          STQ     RESULT         STORE Q IN MEMORY
* MEMORY  NOW  CONTAINS  00XXXX
          RPT     6,1            CONVERT REMAINING VALUE
          BCD     0,7            X7 CONTAINS ADDRESS OF TABLE+4
          STQ     RESULT+1
                   .
                   .
TABLE     DEC     8E9B35,64E8B35,512E7B35
          DEC     4096E6B35,3276800000
          DEC     2621440000,2097152000
          DEC     1677721600,1342177280
          DEC     1073741824
```

XED CODING EXAMPLE

EXAMPLE: A PROGRAM IS MADE UP OF FOUR (4) DIVISIONS WHICH MAY BE ACCESSED AT DIFFERENT TIMES.

X7 INITIALIZED TO DETERMINE ENTRY.

```
            XED     LOC,7
RE-ENTRY--> XXX
            ~
LOC     ESTC1   SAVE
        TRA     FIRST
        STC1    SAVE
        TRA     SECOND
        STC1    SAVE
        TRA     THIRD
        STC1    SAVE
        TRA     FOURTH
```



...ALSO LITERALS AND MACROS.

PREFACE

PROGRAM BREAK	2206
COMMON LENGTH	24
¥ COUNT BITS	5

PRIMARY SYMDEF	ENTRY
.SMDEF	2001
.SAVE	2130

SECONDARY SYMDEF	ENTRY
.EVN	2012

	BLOCK	LENGTH
1	LABCOM	20
2	TALLYS	4

SYMREF

3 .SMREF

...ALSO LITERALS AND MACROS.

```
                                1         TTL     EXAMPLES OF PSUEDO OPERATIONS**
                                2         TTLS    ...ALSO LITERALS AND MACROS.
                                3 **********************************************************************
                                4 **********************************************************************
                                5 **                                                                  **
                                6 **                        THE FOLLOWING PROGRAM LISTING             **
                                7 **                                                                  **
                                8 **                        WILL SHOW MOST OF THE COMMONLY            **
                                9 **                                                                  **
                               10 **                             USED PSEUDO OPS IN A                 **
                               11 **                                                                  **
                               12 **                             RELOCATABLE ASSEMBLY.                **
                               13 **                                                                  **
                               14 **********************************************************************
                               15 **********************************************************************
                               16 *
                               17 * THIS FIRST PSEUDO OP IS NECESSARY FOR ALL RELOCATABLE PROGRAMS IN ORDER
                               18 * TO PROVIDE AN ENTRY POINT FOR GELOAD. IF MORE THAN ONE SYMBOL APPEARS IN
                               19 * THE VARIABLE FIELD THE FIRST SYMBOL WILL BE TAKEN AS THE ENTRY POINT.
                               20 *
                               21         SYMDEF  .SMDEF          PROVIDES AN ENTRY POINT INTO THE PROGRAM.
                               22         SYMDEF  -.EVN           A MINUS SIGN DENOTES A SECONDARY SYMDEF.
                               23 *
                               24 * THE PCC WILL ALLOW THE PRINTING CF ALL PSEUDO OPS FOR YOUR CONVENIENCE.
                               25 *
                               26         PCC     ON
                               27 *
                               28 * THE ORG PSEUDO OP WILL START THE PROGRAM OFF AT LOCATION 2000 OCTAL.
                               29 *
                     002000    30         ORG     1024
002000  002000 2350 00  010    31 .ORG.   LDA     .ORG.
002001  002145 7550 00  010    32 .SMDEF  STA     .OPSYN
002002  002065 2270 00  010    33 START   LDX7    .ZERO.
                               34 * THE EQU PSEUDO OP WILL EQUATE THE SYMBOLIC ADDRES OF .EQU WITH 512 DECIMAL.
                               35 *
                     001000    36 .EQU.   EQU     512
002003  001000 2350 00  000    37         LDA     .EQU.
                               38 *
                               39 * IN THIS EXAMPLE OF THE EQU PSEUDO OP, WE ARE EQUATING ONE SYMBOL.....
                               40 * TO A PREVIOUSLY DEFINED SYMBOL.  HOWEVER, THIS SYMBOL MUST NOT BE A.....
                               41 * SYMDEF OR SYMREF.
                               42 *
                     002002    43 STOP    EQU     START
002004  002002 2350 00  010    44         LDA     START
002005  002002 2350 00  010    45         LDA     STOP
                               46 *
                               47 * THE BOOL PSEUDO OP WILL EQUATE .BOOL. WITH 1200 OCTAL.
                               48 *
                     001200    49 .BOOL.  BOOL    1200
002006  001200 7550 16  000    50         STA     .BOOL.,6
```

...ALSO LITERALS AND MACROS.

```
                                                 51 * THE NULL PSEUDO OP CAUSES THE LABEL TO ASSUME THE NEXT LOCATION COUNTER VALUE.
                                                 52 *
                          002007                 53 .NULL. NULL
002007  002063 2350 00  010                      54         LDA     .ARG.
002010  002204 2350 00  010                      55         LDA     .E.L..          .E.L.. IS THE ERROR LINKAGE SYMBOL.
                                                 56 *
                                                 57 * THIS EVEN PSEUDO OP FORCES THE LOCATION COUNTER TO AN EVEN COUNT
                                                 58 *
002011  000000011007    000
                          002012                 59         EVEN
002012  002035 2350 00  010                      60 .EVN    LDA     .VFD.
002013  002015 7550 00  010                      61         STA     .ODD
                                                 62 *
                                                 63 * THE ODD PSEUDO OP FORCES THE LOCATION COUNTER TO AN ODD COUNT.
                                                 64 *
002014  000000011007    000
                          002015                 65         ODD
002015  002202 2350 00  010                      66 .ODD    LDA     .USE.
002016  002063 7550 00  010                      67         STA     .BFS.
                                                 68 *
                                                 69 * THE EIGHT PSEUDO OP FORCES THE LOCATION COUNTER TO AN EVEN MULTIPLE OF EIGHT.
                                                 70 * NOTICE BITS 15 THRU 17, 6TH OCTAL CHAR, OF THE LOCATION COUNTER IS ALL ZEROS
                                                 71 *
002017  000001710004    000
                          002020                 72         EIGHT
002020  002051 2350 00  010                      73 .EIGHT  LDA     .BSS.
002021  002036 0750 00  010                      74         ADA     .ETC.
002022  002020 7550 00  010                      75         STA     .EIGHT
                                                 76 *
                                                 77 * I HOPE YOU NOTICED THE NOP THAT THE ASSEMBLER INSERTED IN  THE
                                                 78 * INSTRUCTION SEQUENCE WHEN ENCOUNTERING THOSE LAST THREE PSEUDO OPS.
                                                 79 *
                                                 80 *
                                                 81 * WHILE WE ARE AT IT, DID YOU NOTICE THAT THESE REMARKS IN NO WAY EFFECTED
                                                 82 * THE LOCATION COUNTER,OR THE INSTRUCTIONS.
                                                 83 *
                                                 84 *
                                                 85 *
                                                 86 * THE OCT PSEUDO OP USED HERE WILL CREATE 3 OCTAL WORDS STARTING AT LOC. .OCT.
                                                 87 *
002023  000000000300    000                      88 .OCT.   OCT     300,777777525252,-4
002024  777777525252    000
002025  400000000004    000
                                                 89 * THIS DEC PSEUDO OP WILL CREATE THREE WORDS STARTING AT LOCATION .DEC. AFTER
                                                 90 * FIRST CONVERTING FROM DECIMAL TO OCTAL.
                                                 91 *
002026  777777777777    000                      92 .DEC.   DEC     -1,1.2E1832,6.
002027  000000000140    000
002030  006600000000    000
                                                 93 *
```

...ALSO LITERALS AND MACROS.

```
                                      94 * THE BCI PSEUDO OP WILL STORE A FOUR WORD PHRASE STARTING AT LOC. .BCI.
                                      95 *
002031  312043314225    000           96 .BCI.  BCI     4,I LIKE THIS PSEUDO OP.
002032  206330316220    000
002033  476225642446    000
002034  204647332020    000
                                      97 * THE VFD PSEUDO OP WILL CREATE A WORD THAT IS BIT ORIENTED IN ITS EXPRESSION.
                                      98 *
002035  002031775225    010           99 .VFD.  VFD     18/.BCI.,06/77,H6/-,3/2,3/5
                                     100 *
                                     101 * HERE IS A REAL BEAUT.  A VFD USING  ALGEBRAIC AND BOOLEAN OPERATORS.
                                     102 *
                                     103 .ETC.  VFD     4/16/2,         4 BITS OF 16 DIVIDED BY 2
                                     104        ETC     3/2*3,          3 BITS OF 2 TIMES 3
                                     105        ETC     5/15+3,         5 BITS OF 15 PLUS 3
                                     106        ETC     6/17-5,         6 BITS OF 17 MINUS 5
                                     107        ETC     06//16,         BOOLEAN 6 BITS OF 16 NOT
                                     108        ETC     03/2*3,         BOOLEAN 3 BITS OF 2 AND 3
                                     109        ETC     04/15+3,        BOOLEAN 4 BITS OF 15 ORED TO 3
002036  432214612752    000          110        ETC     05/17-5         BOOLEAN 5 BITS OF 17 EX ORED TO 5
                                     111 *
                                     112 * BY THE WAY, THAT ALSO SHOWED THE USE OF THE ETC PSEUDO OP.
                                     113 *
                                     114 ********************************************************************************
                                     115 *
                                     116 * THE FOLLOWING SHOWS A USE OF THE SET AND DUP PSEUDO OPS.  IT WILL PROVIDE.....
                                     117 * CODING TO MOVE 5 WORDS FROM X TO Y.
                                     118 *
                  000000             119 .SET.  SET     0
                  002037             120        DUP     3,5             DUP THE NEXT 3 CARDS, 5 TIMES.
002037  000005 2350 00  020          121        LDA     X+.SET.
002040  000012 7550 00  020          122        STA     Y+.SET.
                  000001             123 .SET.  SET     .SET.+1
002041  000006 2350 00  020                   LDA     X+.SET.
002042  000013 7550 00  020                   STA     Y+.SET.
                  000002             .SET.  SET     .SET.+1
002043  000007 2350 00  020                   LDA     X+.SET.
002044  000014 7550 00  020                   STA     Y+.SET.
                  000003             .SET.  SET     .SET.+1
002045  000010 2350 00  020                   LDA     X+.SET.
002046  000015 7550 00  020                   STA     Y+.SET.
                  000004             .SET.  SET     .SET.+1
002047  000011 2350 00  020                   LDA     X+.SET.
002050  000016 7550 00  020                   STA     Y+.SET.
                  000005             .SET.  SET     .SET.+1
                                     124 *
                                     125 * DID YOU NOTICE THAT THE SET PSEUDO OP ALLOWED THE SAME SYMBOL IN THE LOCATION
                                     126 * FIELD TO BE RE DEFINED WITHOUT ERROR.
```

...ALSO LITERALS AND MACROS.

```
                                    127         EJECT
                                    128 * THE BSS PSEUDO OP WILL RESERVE 5 WORDS STARTING AT LOCATION .BSS.
                                    129 *
                          002051    130 .BSS.   BSS     5
                                    131 *
                                    132 * THIS PSEUDO OP WILL RESERVE 5 WORDS PRECEEDING LOCATION .BSF.
                                    133 *
                          002063    134 .BFS.   BFS     5
                                    135 *
                                    136 * THAT LAST PSEUDO OP IS NICE IF YOU WISH TO USE REVERSE SCANNING ADDRESS
                                    137 * MODIFIERS SUCH AS DI TALLY OR SD TALLY.
                                    138 *
                                    139 * THIS PSEUDO OP WILL CREATE AN INSTRUCTION WORD WITH
                                    140 * NO OP CODE.  IT IS HANDY FOR CREATING ADDRESSES FOR IR OR RI MODIFICATION.
                                    141 *
                          002063    142 .ARG.   NULL
      002063  002007 0000 17  010   143         ARG     .NULL.,7
      002064  002073 0000 11  010   144         ARG     .DLIT.,1
                                    145 *
                                    146 * OR WE COULD CREATE TWO 18 BIT ADDRESSES IN ONE WORD BY USING THE
                                    147 * FOLLOWING PSEUDO OP.
                                    148 *
      002065  002023 002026    011  149 .ZERO. ZERO    .OCT.,.DEC.
                                    150 *
                                    151 * BUT ON THE OTHER HAND IF YOU WOULD RATHER USE ABSOLUTE ADDRESSING RATHER
                                    152 * THAN SYMBOLIC ADDRESSING, YOU CAN CODE IT THIS WAY.
                                    153 *
      002066  002000 001000    000  154         ZERO    1024,512
                                    155 *
                                    156 * OR IF YOU CANT MAKE UP YOUR MIND, YOU CAN USE BOTH SYMBOLIC AND ABSOLUTE.
                                    157 *
      002067  002000 002000    001  158         ZERO    1024,.ORG.      HA, HA. THEY'RE BOTH THE SAME  ADDRESS.
                                    159 ************************************************************************
                                    160 *
                                    161 * HERE IS AN INSTRUCTIONAL LITERAL
                                    162 *
      002070  002120 2350 00  010   163 .MLIT. LDA     =MLDX1 5,DU
      002120  000005 2210 03  000
                                    164 ************************************************************************
                                    165 *
                                    166 * HERE ARE A COUPLE EXAMPLES OF VARIABLE FIELD LITERALS.
      002071  002121 2350 00  010   167 .VLIT. LDA     =V18/START,09/235,3/0,06/17
      002121  002002235017    010
      002072  002122 2350 00  010   168         LDA     =V10/512,02/2,H18/B2X,6/0
      002122  400222026700    000
                                    169 ************************************************************************
                                    170 *
                          002073    171 .REF.   NULL
                                    172         REF     LNRSM           LIST NONREFERENCED SYMBOLS (EG: .REF.)
                                    173 * NOTICE: THIS WILL CAUSE A COMPLETE REFERENCE LISTING OF ALL THE MME'S.
```

...ALSO LITERALS AND MACROS.

```
                                    174 * HERE ARE SOME EXAMPLES OF DECIMAL LITERALS.
                                    175 *
002073  002104 2350 00  010         176 .DLIT. LDA     =10             DECIMAL INTEGER
002074  002105 2350 00  010         177        LDA     =-15            NEGATIVE DECIMAL INTEGER
002075  002106 2350 00  010         178        LDA     =6.0            SINGLE PRECISION FLOATING POINT
002076  002107 2350 00  010         179        LDA     =1.55E1         SINGLE PRECISION FLOATING POINT
002077  002110 2370 00  010         180        LDAQ    =5.12D2         DOUBLE PRECISION FLOATING POINT
002100  002112 2350 00  010         181        LDA     =5817           SINGLE PRECISION FIXED POINT
002101  002113 2350 00  010         182        LDA     =22.585         SINGLE PRECISION FIXED POINT
002102  002114 2350 00  010         183        LDA     =1.2E1B32       SINGLE PRECISION FIXED POINT
002103  002116 2350 00  010         184        LDA     =1.95D1B37      DOUBLE PRECISION FIXED POINT
                                    185 ********************************************************************************
                  002104           186 .LIT.  NULL
                                    187        LIT                     CAUTION...NOTICE THE LOCATION COUNTER.
002104  000000000012    000
002105  777777777761    000
002106  006600000000    000
002107  010760000000    000
002110  024400000000    000
002111  000000000000    000
002112  000005000000    000
002113  264000000000    000
002114  000000000140    000
002115  000000000000    000
002116  000000000004    000
002117  700000000000    000
                                    188 ********************************************************************************
                                    189 *
                                    190 * IF I USE THE SYMREF PSEUDO OP, I CAN REFERENCE A SYMBOL WHICH DOES NOT.....
                                    191 * APPEAR IN THIS ASSEMBLY, AND WILL NOT CAUSE AN ERROR.
                                    192 *
                                    193        SYMREF  .SMREF
002123  030000 7100 00  030         194        TRA     .SMREF
                                    195 *
                                    196 * OF COURSE IT IS EXPECTED THAT THIS SAME SYMBOL WILL HAVE BEEN DEFINED.....
                                    197 * BY A SYMDEF IN ANOTHER ASSEMBLY TO BE EXECUTED WITH THIS ONE.
                                    198 ********************************************************************************
                                    199 *
                                    200 * IF I NEED TO TURN BIT 28 ON, I WOULD USE THE FOLLOWING PSEUDO OP.
                                    201 *
                                    202        INHIB   ON
002124  002002 2352 00  010         203 .INHIB LDA     START
002125  002124 7552 00  010         204        STA     .INHIB
                                    205 *
                                    206 * TO TURN IT OFF, I MERELY DO THE FOLLOWING.
                                    207 *
                                    208        INHIB   OFF
002126  002002 2350 00  010         209        LDA     START
002127  002124 7550 00  010         210        STA     .INHIB
                                    211 *
```

...ALSO LITERALS AND MACROS.

```
                                212 * THE FOLLOWING IS AN EXAMPLE OF THE USE OF THE CALL, SAVE, AND RETURN PSEUDOS.
                                213 * NOTE:  THE SAVE AND RETURN PSEUDO OPS MUST APPEAR IN THE ASSEMBLY BEFORE....
                                214 * THE CALL PSEUDO OP.
                                215 *
                     002130     216 .SAVE  SAVE                    THE SYMBOL .SAVE WILL APPEAR AS A SYMDEF.
002130  002132710000    010
002131  002204630000    010
002132  002204754000    010
002133  002204741000    010
002134  002202 2350 00  010    217        LDA     .USE.           BODY OF THE SUBROUTINE
002135  002137 7550 00  010    218        STA     .CALL                   DITTO
002136  002131710000    010    219 .RET.  RETURN  .SAVE
                                220 *
                                221 * NOTE ** THE CALL PSEUDO OP MUST FOLLOW THE SAVE AND RETURN PSEUDO OPS.........
                                222 * IF USED IN THE SAME ASSEMBLY.
                                223 *
                                224 *
002137  002130701000    010    225 .CALL  CALL    .SAVE           THE SYMBOL .SAVE WILL APPEAR AS A SYMREF.
002140  002142710000    010
002141  002204000341    010
002142  002104 6350 00  010    226        EAA     .LIT.           RETURN POINT INTO PROGRAM.
                                227 *
                                228 ************************************************************************************
                                229 *
                                230 * HERE ARE TWO EXAMPLES OF FLOATABLE CODE.........
                                231 *
002143  777637 2350 04  000    232 .FLOAT LDA     START-*,IC      AS YOU CAN SEE, FLOATABLE CODE....
002144  777636 2350 04  2002   233        LDA     START,$         MAY BE CODED EITHER WAY.
                                234 *
                                235 * SUPPOSE YOU WOULD PREFER TO USE THE MNEMONICS LDAR INSTEAD OF LDA.....
                                236 * FOR THE LOAD THE A REGISTER INSTRUCTION.  THEN YOU WOULD USE THE FOLLOWING.
                                237 *
                                238 LDAR   OPSYN   LDA
                                239 *
                                240 * NOW LETS SEE WHAT HAPPENS WHEN WE USE IT.
                                241 *
002145  002070 2350 00  010    242 .OPSYN LDAR    .MLIT.
002146  002071 2350 00  010    243        LDA     .VLIT.
002147  002202 7010 00  010    244        TSX1    .USE.
                                245 *
                                246 * WATCH ME CHANGE THE LOCATION COUNTER BY USING THE USE PSEUDO OP.
                                247 *
                     002202     248        USE     NEXT            NEXT IS THE NAME OF MY NEW LOCATION CTR.
002202  002143 2350 00  010    249 .USE.  LDA     .FLOAT
002203  002136 7550 00  010    250        STA     .RET.
                                251 *
                                252 * NOW WATCH ME PICK UP WHERE I LEFT OFF WITH THE ORIGINAL LOCATION COUNTER.
                                253 *
                     002150     254        USE
                                255 *
```

...ALSO LITERALS AND MACROS.

```
                                        256 * SUPPOSE YOU WANT TO SHOW BLANK OR LABELLED COMMON AREAS, BELOW THE CODING.
002150  000000 2350 00  020             257         LDA     .BLOCK
002151  010000 0750 00  030             258         ADA     WW
002152  000005 7550 00  020             259         STA     X
                                        260 *
                                        261 * THE FOLLOWING BSS PSEUDO OPS WILL RESERVE AREAS IN BLANK COMMON.
                                        262 * NOTE***  INSTRUCTIONS OR DATA MAY NOT BE PLACED IN BLANK COMMON.
                                        263 *
                000000                  264         BLOCK                       THIS IS BLANK COMMON OF COURSE.
                000000                  265 .BLOCK  BSS     5
                000005                  266 X       BSS     5
                000012                  267 Y       BSS     5
                000017                  268 Z       BSS     5
                                        269 *
                                        270 *
                000000                  271         BLOCK   LABCOM              AND THIS IS LABELED COMMON.
                000000                  272 WW      BSS     10
000012  000000000143    000             273 XX      DEC     99
000013  000000000077    000             274 YY      OCT     77
000014  633031622031    000             275 ZZ      BCI     4,THIS IS LABELED COMMON.
000015  622043212225    000
000016  432524202346    000
000017  444446453320    000
                                        276 *
                                        277 * NOTICE THAT LABELED COMMON PERMITTED THE USE OF CONSTANTS.....
                                        278 * WITHIN THE LABELED COMMON AREA.
                                        279 *
                002153                  280         USE                         AGAIN I RETURN TO MY ORIGINAL COUNTER
                                        281 *
                                        282 * HERE ARE EXAMPLES OF THE FOUR TALLY PSEUDO OPS.
                                        283 *
                                        284 *
002153  020000 2350 52  030             285         LDA     .TALY.,SC
002154  020001 2360 52  030             286         LDQ     .TALB.,SC           NOTE THE SC MOD IS USED FOR BYTE ALSO.
002155  020002 7550 57  030             287         STA     .TALC.,IDC          IDC ALLOWS FURTHER INDIRECTION.
002156  020003 7560 53  030             288         STQ     .TALD.,AD
                                        289 *
                                        290 *
                000000                  291         BLOCK   TALLYS          LABELED COMMON AGAIN.
000000  010000 0012 02  030             292 .TALY.  TALLY   WW,10,2           TALLY USED FOR I, ID, DI, SC, & CI FORMS.
000001  010012 0012 41  030             293 .TALB.  TALLYB  XX,10,1           NOTICE BIT 30 ON FOR BYTE TALLY.
000002  010013 0012 77  030             294 .TALC.  TALLYC  YY,10,*7          NOTICE IR MODIFICATION IN THE TAG FIELD.
000003  010014 0012 12  030             295 .TALD.  TALLYD  ZZ,10,10
                                        296 *
                002157                  297         USE                         BACK TO THE ORIGINAL LOCATION CTR AGAIN.
                                        298 ********************************************************************************
```

...ALSO LITERALS AND MACROS.

```
                                299         EJECT
                                300 *
                                301         REFMA   ON              CREATES A SEPERATE MACRO SYMBOL TABLE.
                                302 ********************************************************************************
                                303 *
                                304 * THIS NEXT SEQUENCE OF INSTRUCTIONS WILL SERVE TO EXPLAIN THE USE OF MACROS.
                                305 * IT WILL ALSO SHOW THE USE OF THE CONDITIONAL PSEUDO OPS, CREATED SYMBOLS,
                                306 * IORP AND THE PRINT MACRO EXPANSION ON/OFF PSEUDO OP.
                                307 *
                                308         PMC     ON
                                309 *
                                310 *               TURN  PMC  ON   TO SEE EXPANDED CODE
                                311 *
                                312 ********************************************************************************
                                313 *
                                314 HOLY    MACRO
                                315         LDX7    #1,DU           ARG #1 IS NR. OF WORDS TO BE MOVED.
                                316         LDX6    0,DU
                                317 #2      LDA     #3,6            ARG #3 IS DATA ORIGIN.
                                318         STA     #4,6            ARG #4 IS DATA DESTINATION.
                                319         IFG     #1,1,3          IF ARG #1 GREATER THAN ONE, ASSEMBLE AND
                                320 *                               EXECUTE THE NEXT THREE INSTRUCTIONS.
                                321         ADX6    1,DU
                                322         SBX7    1,DU
                                323         TNZ     #2              XFER BACK TO ARG #1 IF NOT ZERO.
                                324         ENDM    HOLY
                                325 *
                                326 * ARGUMENT NUMBER 2 IS IMPLICITLY NULLED TO FORCE CREATED SYMBOLS.
                                327         HOLY    25,#,X,Y
                          002157            LDX7    25,DU
002157  000031 2270 03  000     LDX6    0,DU
002160  000000 2260 03  000 .001. LDA     X,6
002161  000005 2350 16  020     STA     Y,6
002162  000012 7550 16  020     IFG     25,1,3
002163  000001 0660 03  000     ADX6    1,DU
002164  000001 1670 03  000     SBX7    1,DU
002165  002161 6010 00  010     TNZ     .001.
                                328 *
                                329 * HERE WE USED THE SYMBOLIC LOCATION  BIG RATHER THAN CREATED SYMBOLS.
                                330         HOLY    1,BIG,XX,YY
                          002166            LDX7    1,DU
002166  000001 2270 03  000     LDX6    0,DU
002167  000000 2260 03  000 BIG  LDA     XX,6
002170  010012 2350 16  030     STA     YY,6
002171  010013 7550 16  030     IFG     1,1,3
                                331 *
                                332 * LETS SEE THAT LAST ONE WITH PMC OFF THIS TIME.
                                333         PMC     OFF
                          002172 334        HOLY    1,BIGGER,XX,YY
```

...ALSO LITERALS AND MACROS.

```
                                335        EJECT
                                336 *
                                337 *  WATCH CAREFULLY NOW.  HERE IS THE IDRP PSEUDO OP AT WORK.  WITH THIS
                                338 * WE SHALL SEE THE ARG PSEUDO OP AGAIN.
                                339 DOIT   MACRO
                                340        TSX1    #1
                                341        IDRP    #2  REPEAT NEXT INST BY NUMBER OF SUB ARGS IN ARG#2.
                                342        ARG     #2              THIS IS THE ARG PSEUDO OP.
                                343        IDRP                    THIS IDRP IS USED AS A DELIMITER.
                                344        ENDM    DOIT
                                345 *
                                346 * LETS TURN PMC ON AND SEE THIS MACRO EXPANDED.
                                347 *
                                348        PMC     ON
                                349 *
                    002176      350        DOIT    START,(XX,YY,ZZ)
002176  002002 7010 00  010               TSX1    START
                                          IDRP    XX,YY,ZZ
002177  010012 0000 00  030               ARG     XX
002200  010013 0000 00  030               ARG     YY
002201  010014 0000 00  030               ARG     ZZ
                                          IDRP
                                351 ************************************************************************
                                352 *
                                353 * TH-TH-THATS ALL FOLKS, BUT LET US NOT FORGET THE END CARD.
                                354 *
```

ERROR LINKAGE

```
002204  000000000000    000
002205  336244242526    000
                        355        END                     NO SYMBOL IN THE VARIABLE FIELD FOR RELOCABLE.
 2206 IS THE NEXT AVAILABLE LOCATION.   GMAP VERSION  JMPA/060970 JMPB/060970 JMPC/060970
THERE WERE    NO  WARNING FLAGS IN THE ABOVE ASSEMBLY
```

OCTAL SYMBOL REFERENCES BY ALTER NO.

OCTAL	SYMBOL	REFERENCES BY ALTER NO.			
2170	BIG	330	330		
2174	BIGGER	334	334		
10	GEBORT				
22	GECALL				
27	GECHEK				
16	GEENDC				
3	GEFADD				
12	GEFCON				
13	GEFILS				
7	GEFINI				
40	GEFRCE				
41	GEFSYE				
35	GEIDSE				
45	GEINFO				
1	GEINOS				
6	GELAPS				
37	GELBAR				
33	GELOOP				
11	GEMORE				
25	GEMREL				
43	GENEWS				
42	GEPRIO				
17	GERELC				
4	GERELS				
15	GERETS				
2	GEROAD				
31	GEROLL				
30	GEROUT				
24	GERSTR				
23	GESAVE				
14	GESETS				
5	GESNAP				
44	GESNUM				
20	GESPEC				
26	GESYOT				
21	GETIME				
32	GEUSER				
34	GEWAKE				
2161	.001.	327	327		
2063	.ARG.	142	54	142	
2031	.BCI.	96	96	99	
2063	.BFS.	134	67	134	
0	.BLOCK	265	257	265	
1200	.BOOL.	49	49	50	
2051	.BSS.	130	73	130	
2137	.CALL	225	218	225	
2026	.DEC.	92	92	149	
2073	.DLIT.	176	144	176	
2020	.EIGHT	73	73	75	
2204	.E.L..		55	216	225

OCTAL SYMBOL REFERENCES BY ALTER NO.

OCTAL	SYMBOL	REFERENCES BY ALTER NO.									
36	.EMM										
1000	.EQU.	36	36	37							
2036	.ETC.	103	74	103							
2012	.EVN	60	22	60							
2143	.FLOAT	232	232	249							
2124	.INHIB	203	203	204	210						
2104	.LIT.	186	186	226							
2070	.MLIT.	163	163	242							
2007	.NULL.	53	53	143							
2023	.OCT.	88	88	149							
2015	.ODD	66	61	66							
2145	.OPSYN	242	32	242							
2000	.ORG.	31	31	158							
2073	.REF.	171	171								
2136	.RET.	219	219	250							
2130	.SAVE	216	216	219	225						
5	.SET.	123	119	121	122	123					
2001	.SMDEF	32	21	32							
3	.SMREF		193	194							
1	.TALB.	293	286	293							
2	.TALC.	294	287	294							
3	.TALD.	295	288	295							
0	.TALY.	292	285	292							
2202	.USE.	249	66	217	244	249					
2035	.VFD.	99	60	99							
2071	.VLIT.	167	167	243							
2065	.ZERO.	149	33	149							
2002	START	33	33	43	44	167	203	209	232	233	350
2002	STOP	43	43	45							
0	WW	272	258	272	292						
5	X	266	121	123	259	266	327				
12	XX	273	273	293	330	334	350				
12	Y	267	122	123	267	327					
13	YY	274	274	294	330	334	350				
17	Z	268	268								
14	ZZ	275	275	295	350						

```
OCTAL    SYMBOL    REFERENCES BY ALTER NO.

MACRO REF TABLE BY ALTERS

  2176   DOIT              339   350
  2157   HOLY              314   327   330   334
**  17845 WORDS OF MEMORY WERE USED BY GMAP FOR THIS ASSEMBLY.
```

MACROS

MACRO PROTOTYPE:

```
1       8
DOIT    MACRO                       (HEADING)

        LDA     = 5
        ADA     COUNT               (BODY)
        STA     TOTAL

        ENDM    DOIT                (TERMINATOR)
```

MACRO EXPANSION CALL:

```
        DOIT
```

MACRO

WITH

SUBSTITUTABLE ARGUMENTS

MACRO PROTOTYPE:

```
DOIT    .MACRO
        LDA         #1
        ADA         #2
        STA         #3
        ENDM        DOIT
```

MACRO CALL:

```
        DOIT        COUNT,=5,TOTAL
```

REFRESHING TALLY WORDS

WITH MACRO CALL

```
RESTOR    MACRO
          EAX6        #2
          EAX7        #3
          RPD         #1,1
          LDA         0,6
          STA         0,7
          ENDM        RESTOR
           .
           .
           .
          RESTOR      150,TALSAV, TALWRK
```

THIS EXAMPLE WILL RESTORE UP TO
256 TALLY WORDS WITH ONE CALL

LOCATION FIELD SYMBOLS

1	8	16
HOLY	MACRO	
	EAX7	0
GO	LDA	#1,7
	STA	#2,7
	ADX7	1,DU
	CMPX7	#3,DU
	TNZ	GO
	ENDM	HOLY

NESTING OF MACROS

```
PLEASE    MACRO
          TZE     #1
          TRC     #2
          ENDM    PLEASE

AGAIN     MACRO
          CMPA    #1
          PLEASE  #2
          ENDM    AGAIN

DOIT      MACRO
          LDA     #1
          AGAIN   #2
          STA     #3
          ENDM    DOIT
```

MACRO WITH CONDITIONAL PSEUDO OP

MACRO PROTOTYPE:

```
BINBCD      MACRO
            EAX7        0
            LDA         #1
            EAQ         0
            IFE         #1,SEQBIN,3
            RPT         3,1
            BCD         TAB1,7
            STQ         SEQBCD
            INE         #1,SEQBIN,6
            RPT         4,1
            BCD         TAB,7
            STQ         #2
            RPT         6,1
            BCD         0,7
            STQ         #2+1
            ENDM        BINBCD
```

MACRO CALL:

```
            BINBCD      EBIN,EBCD

            BINBCD      FBIN,FBCD

            BINBCD      SEQBIN
```

MACRO WITH

INDEFINITE REPEAT

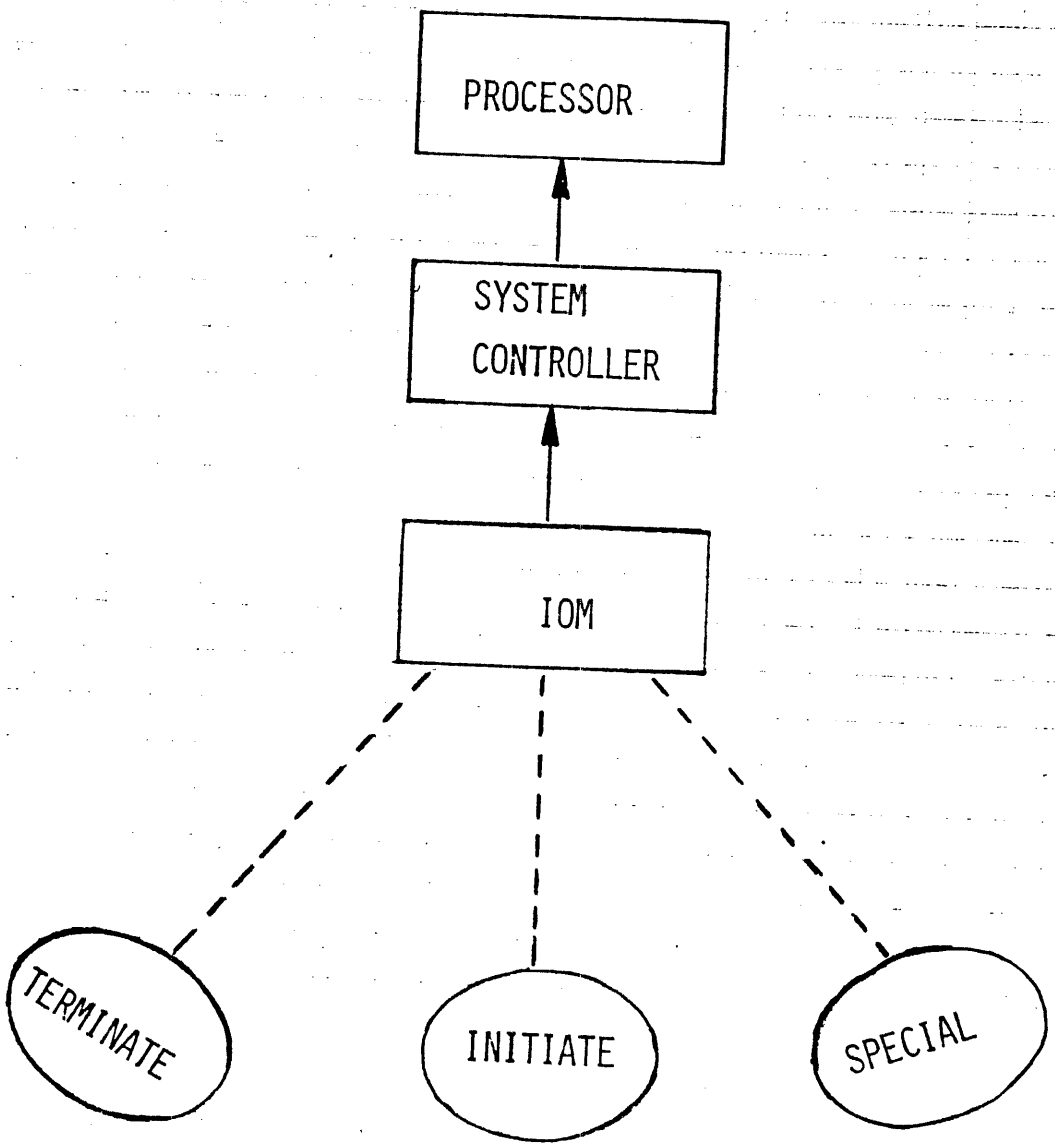
```
1               8               16
DOIT            MACRO
                LDA             #1
                IDRP            #2
                ADA             #2
                IDRP
                STA             #3
                ENDM            DOIT
```

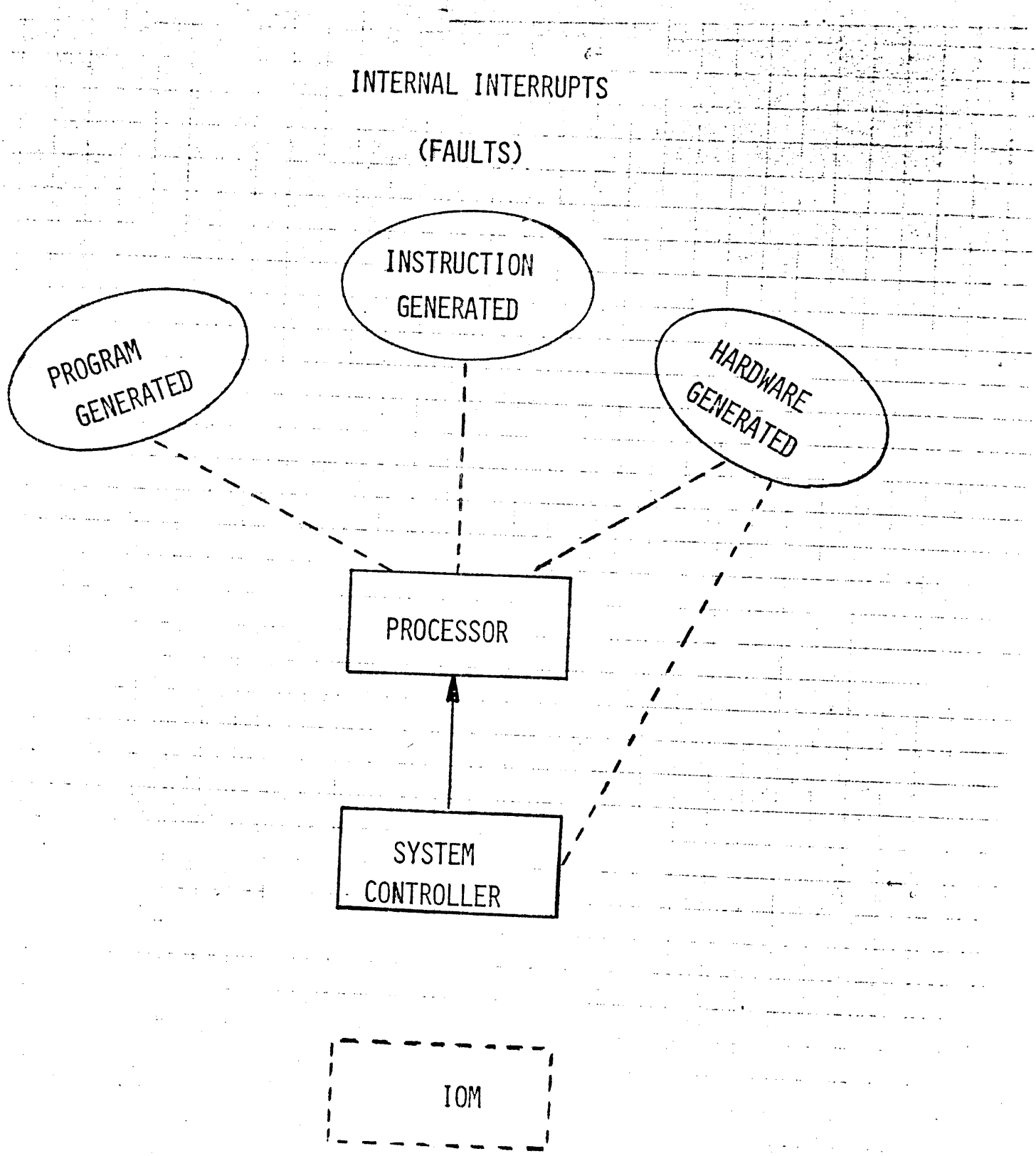
MACRO WITH SET

AND INDEFINITE REPEAT

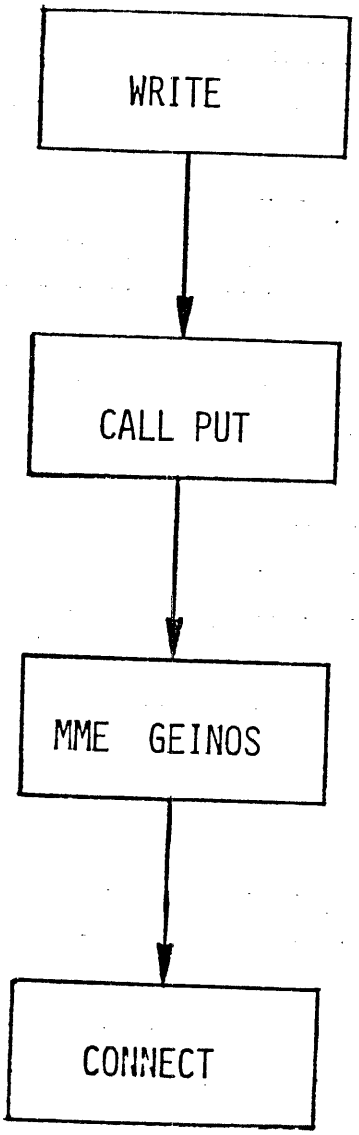
```
1               8               16
XFER            MACRO
K               SET             0
                IDRP            #2
K               SET             K+1
                IDRP
                TSX1            #1
                TRA             *+1+K
                IDRP            #2
                ARG             #2
                IDRP
                ENDM            XFER
```

EXTERNAL INTERRUPTS




INTERNAL INTERRUPTS
(FAULTS)
INSTRUCTION GENERATED
PROGRAM GENERATED
HARDWARE GENERATED
PROCESSOR
SYSTEM CONTROLLER
IOM

LEVELS OF I/O



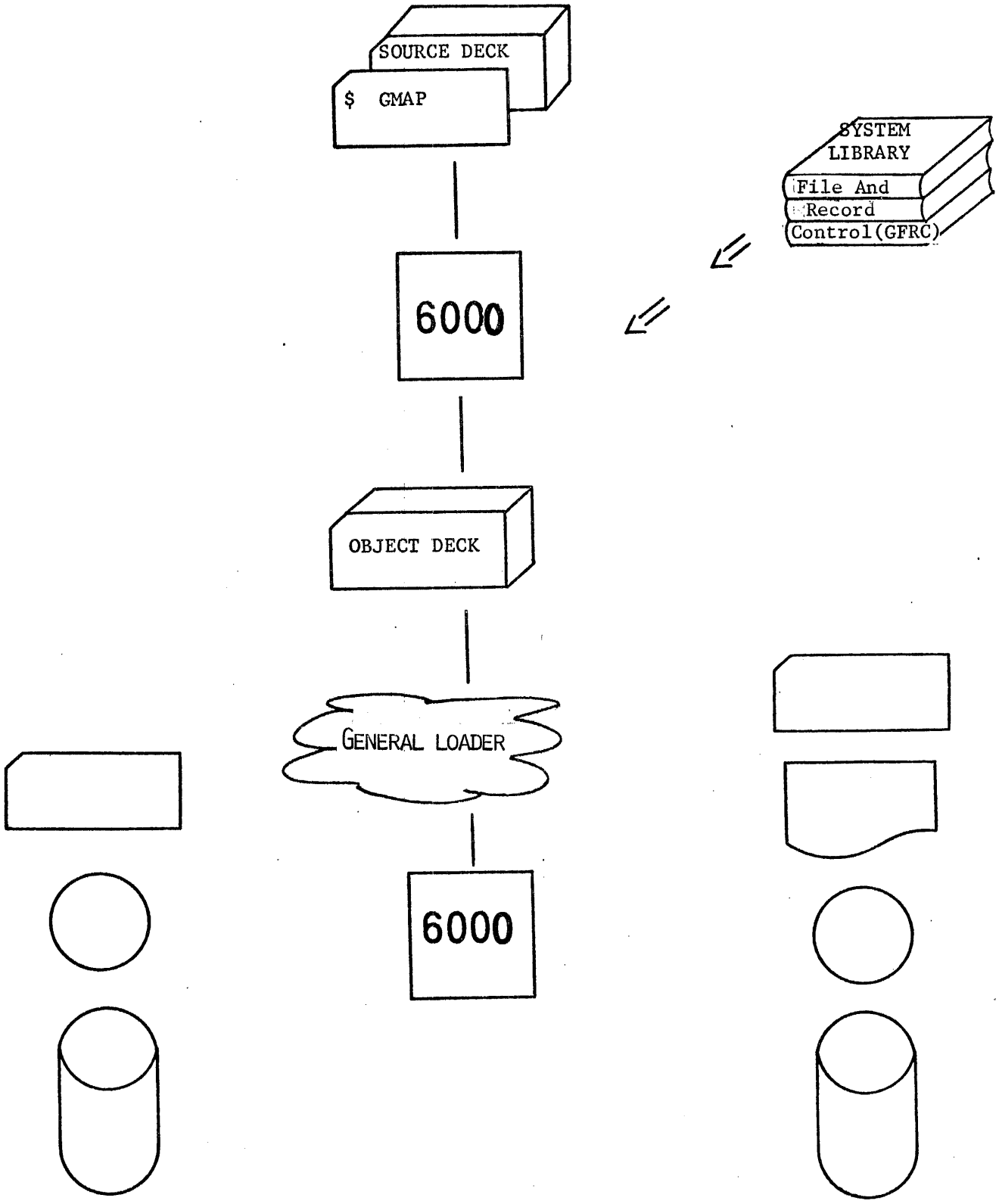
(E.G., COBOL)

FILE AND RECORD CONTROL (GEFRC) SUBROUTINE

(REQUEST FOR I/O)

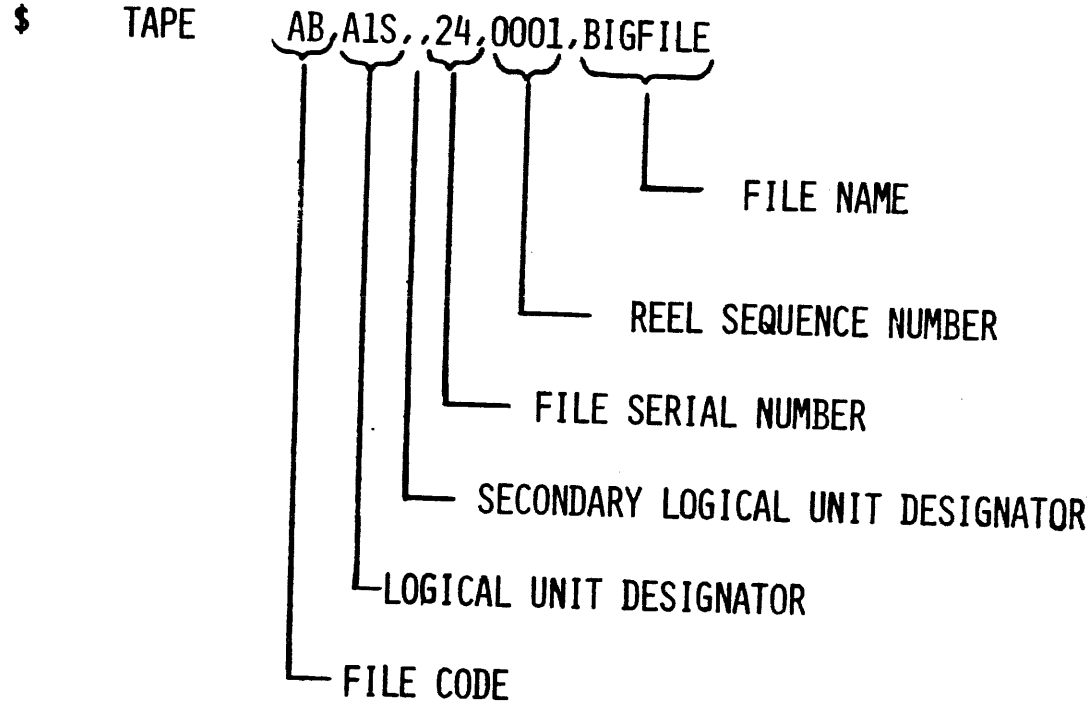
(IOS)

Pages 26-171 through
26-182 have been
deleted from this handbook.


SOURCE DECK
$ GMAP
SYSTEM LIBRARY
File And Record Control(GFRC)
6000
OBJECT DECK
GENERAL LOADER
6000

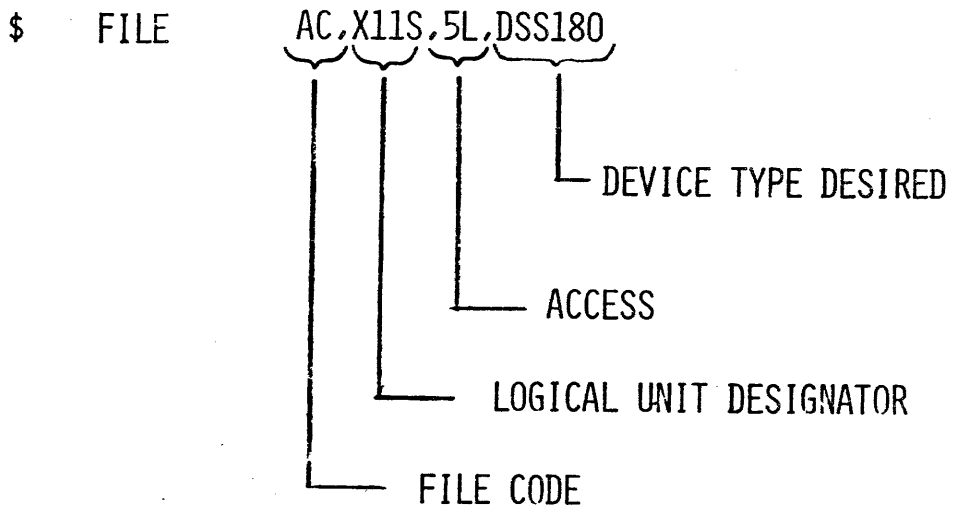
$ TAPE FILE CARD

ASSIGN A TAPE UNIT TO THE PROGRAM.



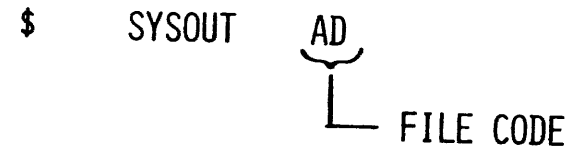
MASS STORAGE FILE CARD

ALLOCATE A TEMPORARY FILE ON SYSTEM MASS STORAGE.



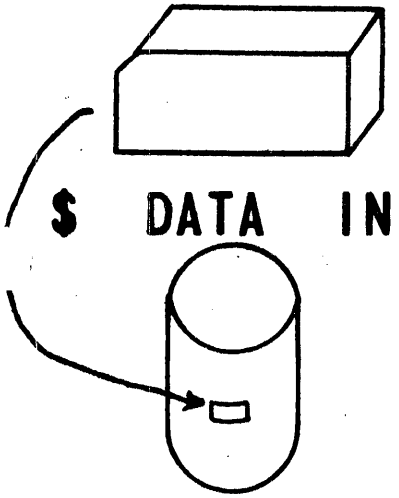
SYSOUT & DATA FILE CARDS

ASSIGN OUTPUT FILES TO SYSTEM OUTPUT MEDIA CONVERSION.

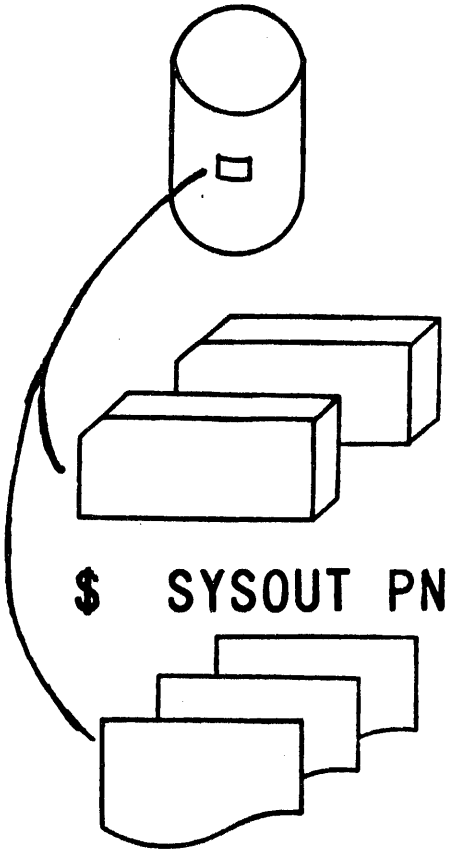


WRITE FILES ONTO TEMPORARY LINKED SYSTEM STORAGE FOR INPUT INTO USER PROGRAM.

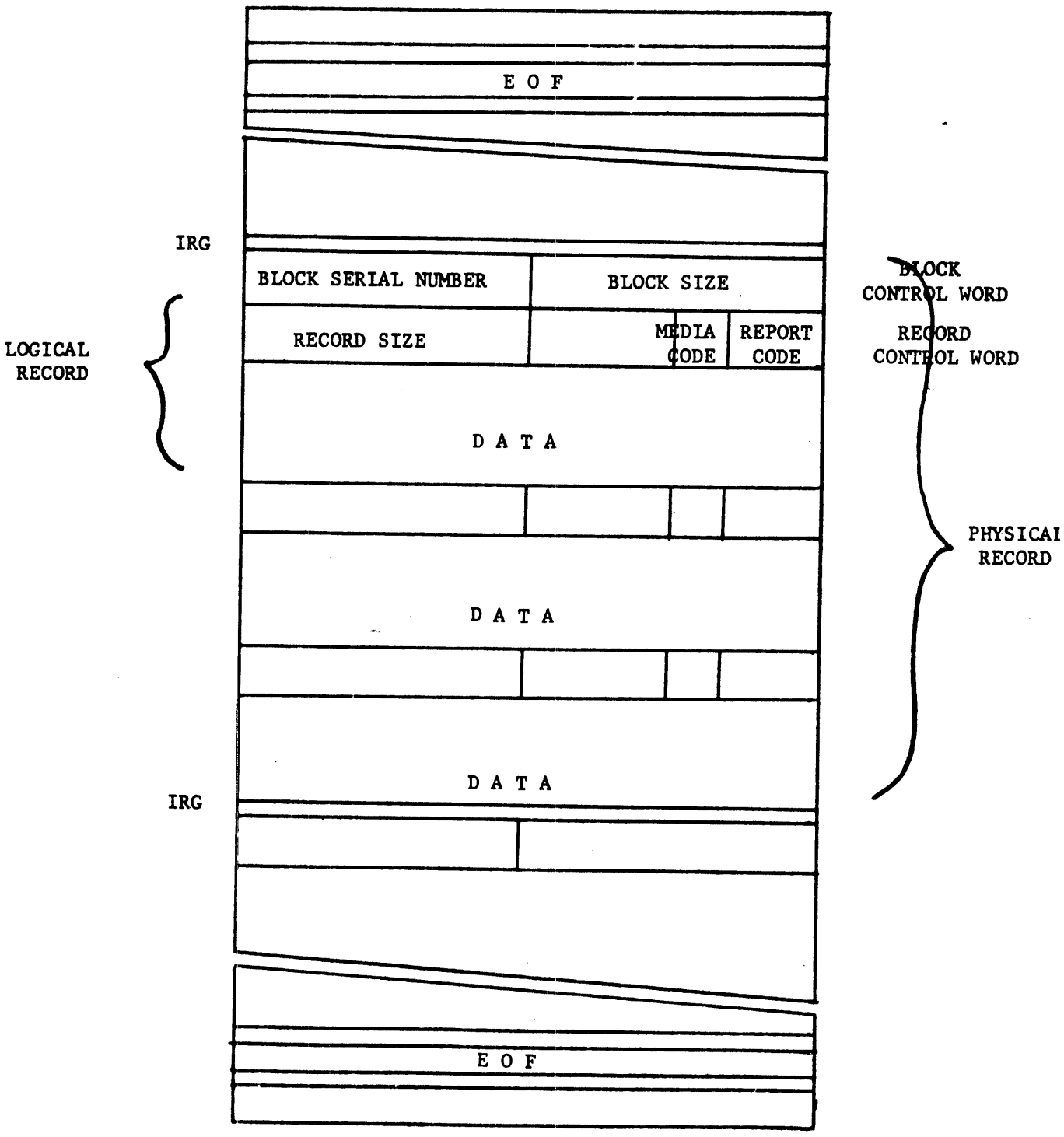
MEDIA CONVERSION



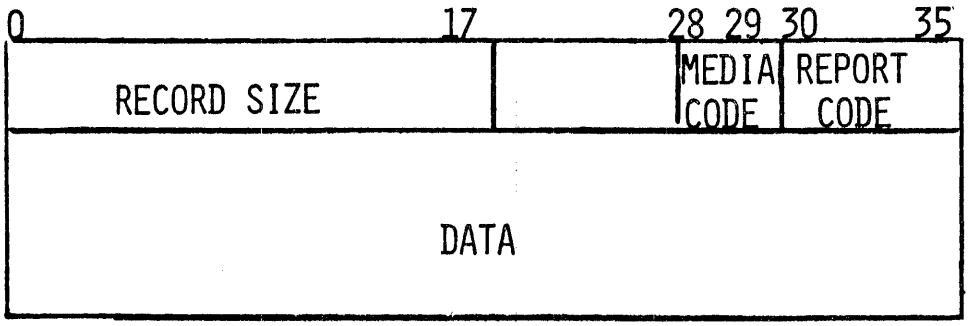

```
$  SNUMB
$  IDENT
$  GMAP
     .
     .
     .
   END
$  EXECUTE
$  DATA  IN
     .
     .
$  SYSOUT PN
$  SYSOUT PR
$  ENDJOB
```



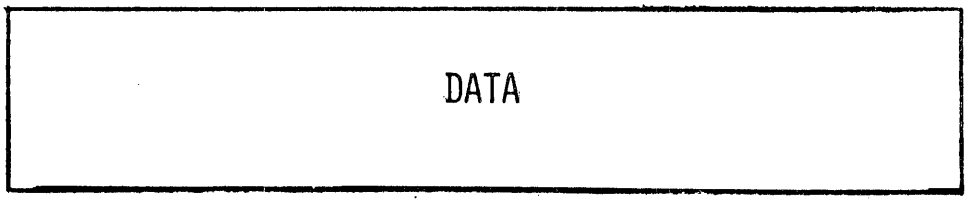
LOGICAL AND PHYSICAL RECORDS



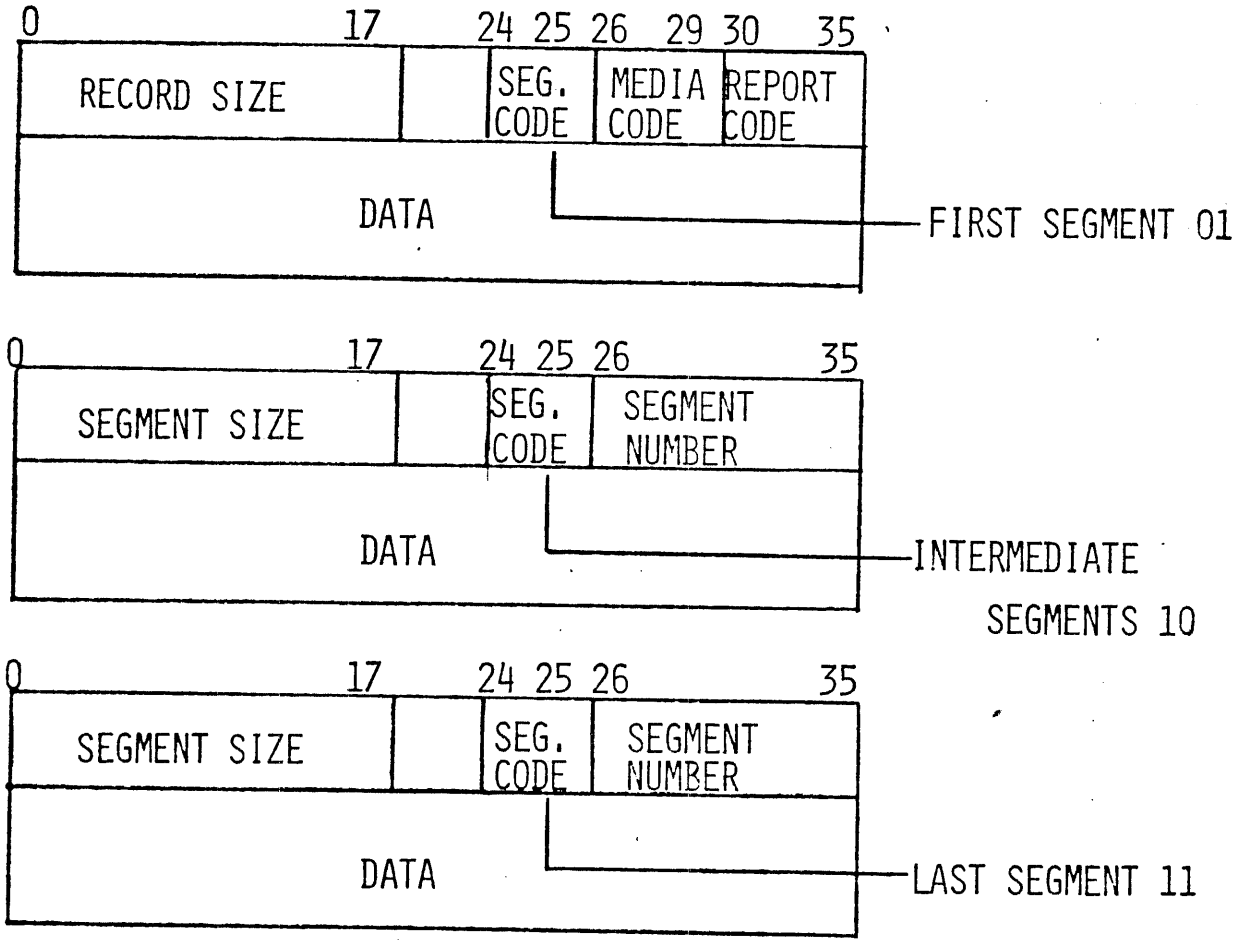
VARIABLE LENGTH RECORDS

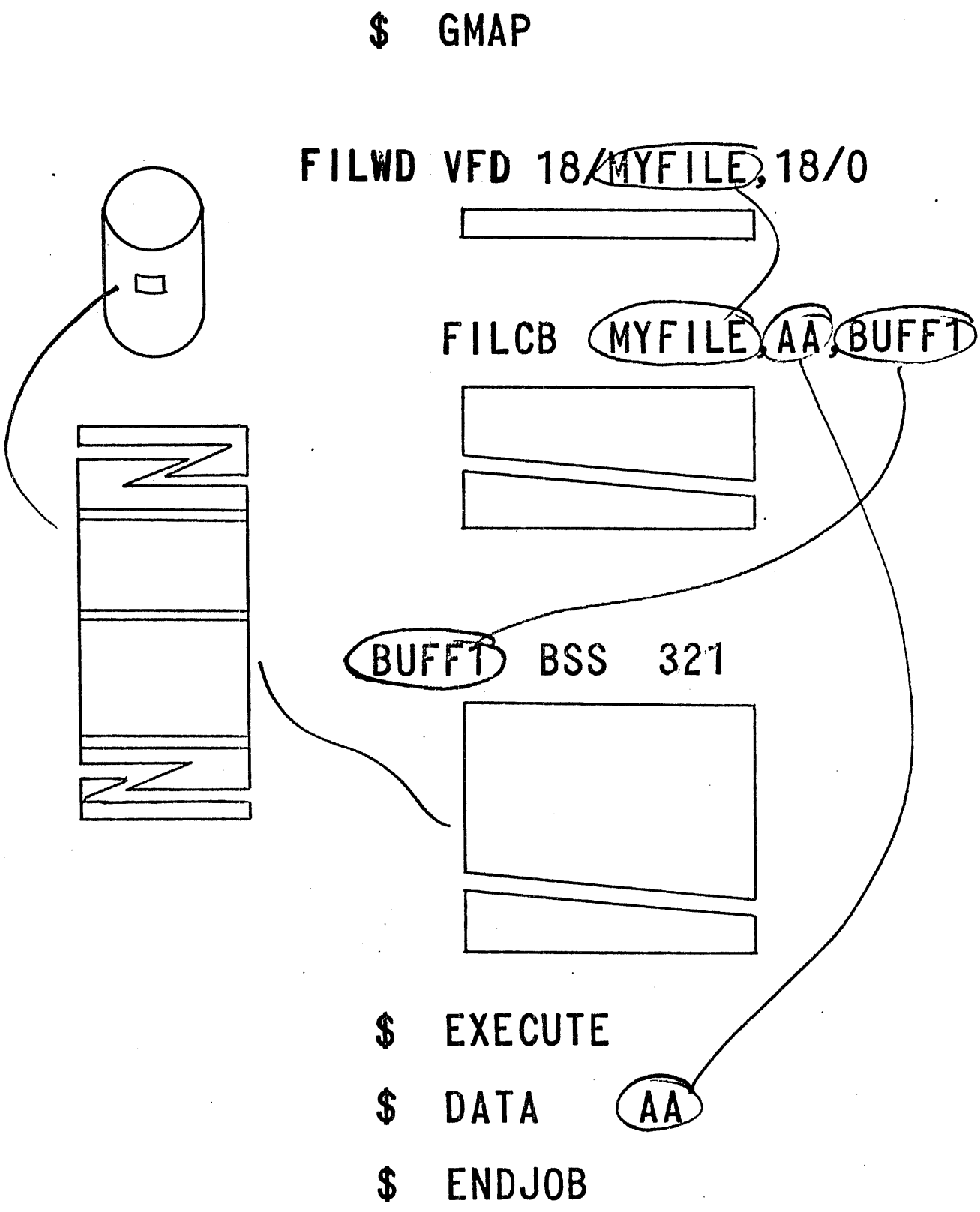


FIXED LENGTH RECORDS



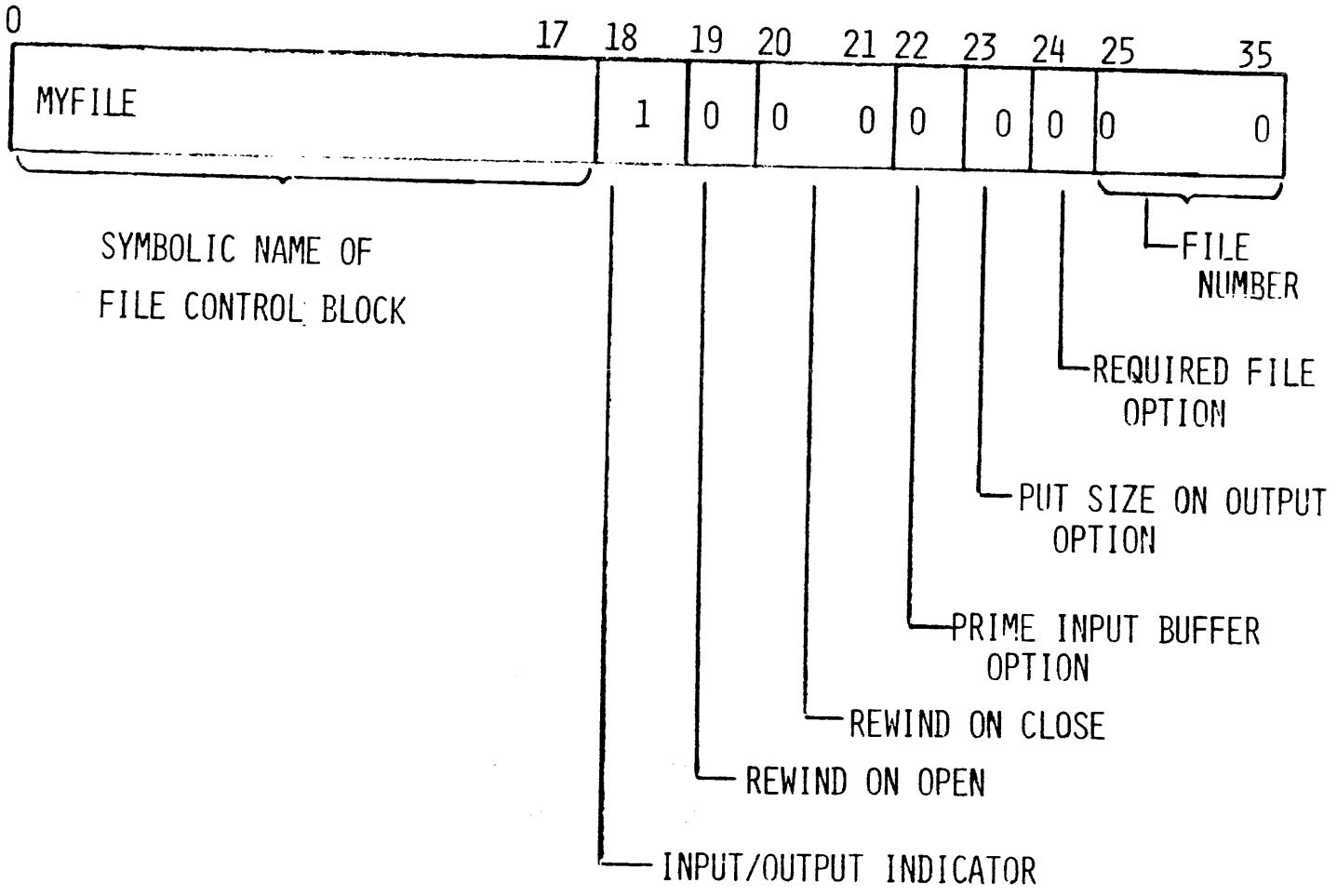
PARTITIONED RECORDS




$ GMAP
FILWD VFD 18/MYFILE,18/0
FILCB MYFILE,AA,BUFFT
BUFFT BSS 321
$ EXECUTE
$ DATA AA
$ ENDJOB

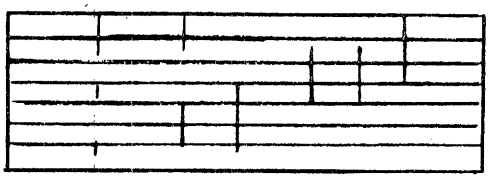
FILE DESIGNATOR WORD

VFD 18/MYFILE,1/1,17/0

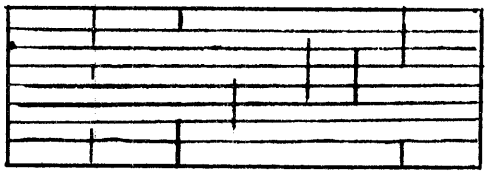


OPEN FILES

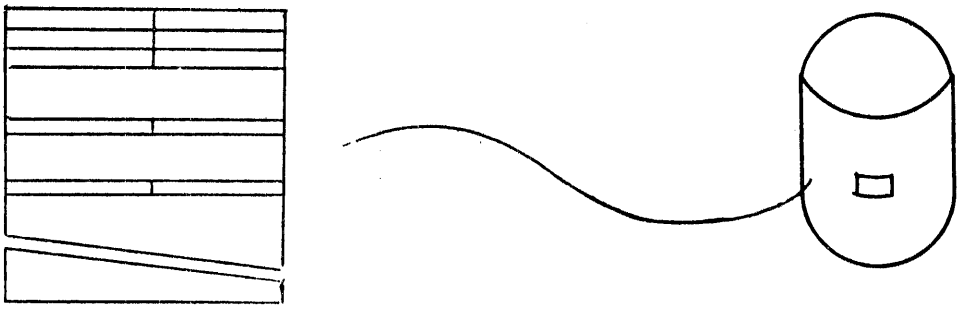
```
          CALL    OPEN(AWD,2)

AWD       VFD     18/FILEAA,18/0
          VFD     18/FILEBB,1/1,17/0

          FILCB   FILEAA,AA,BUFFA1,BUFFA2
```



```
          FILCB   FILEBB,BB,BUFFB1
```



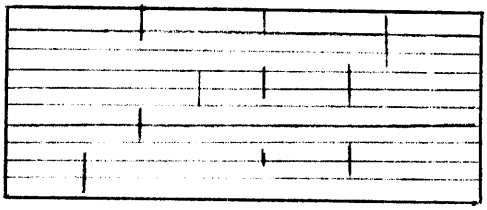
```
BUFFA1    BSS     321
```



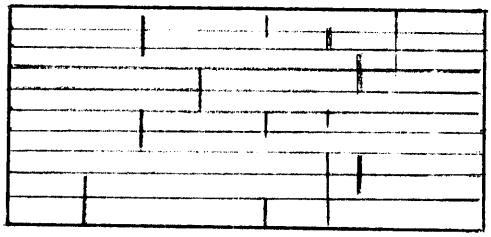
```
$         DATA    AA

$         TAPE    BB
```

CLOSE FILES

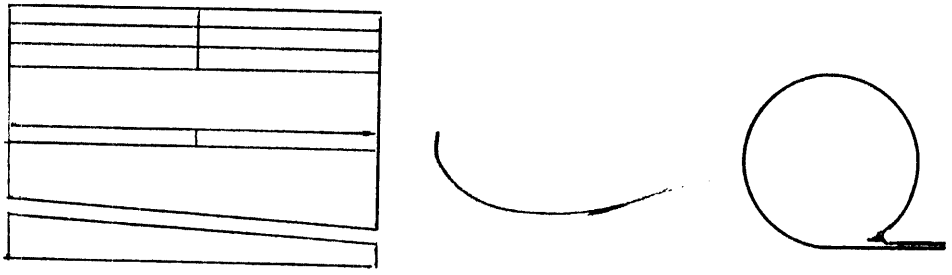
```
          CALL   CLOSE(AWD,2)

AWD       VFD    18/FILEAA,18/0
          VFD    18/FILEBB,1/1,17/0

          FILCB  FILEAA,AA,BUFFA1,BUFFA2
```



```
          FILCB  FILEBB,BB,BUFFB1
```

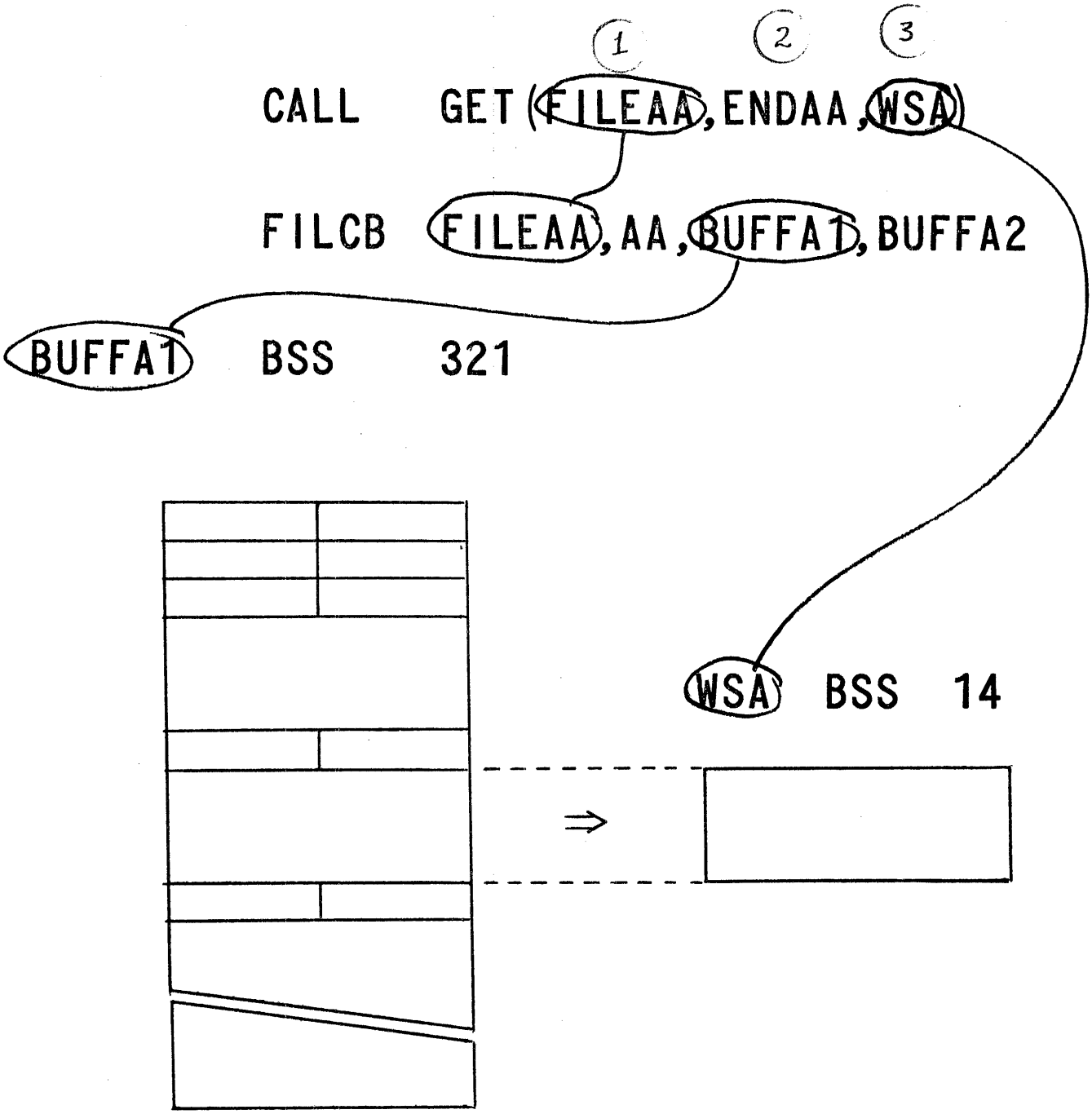


```
BUFFB1    BSS    321
```

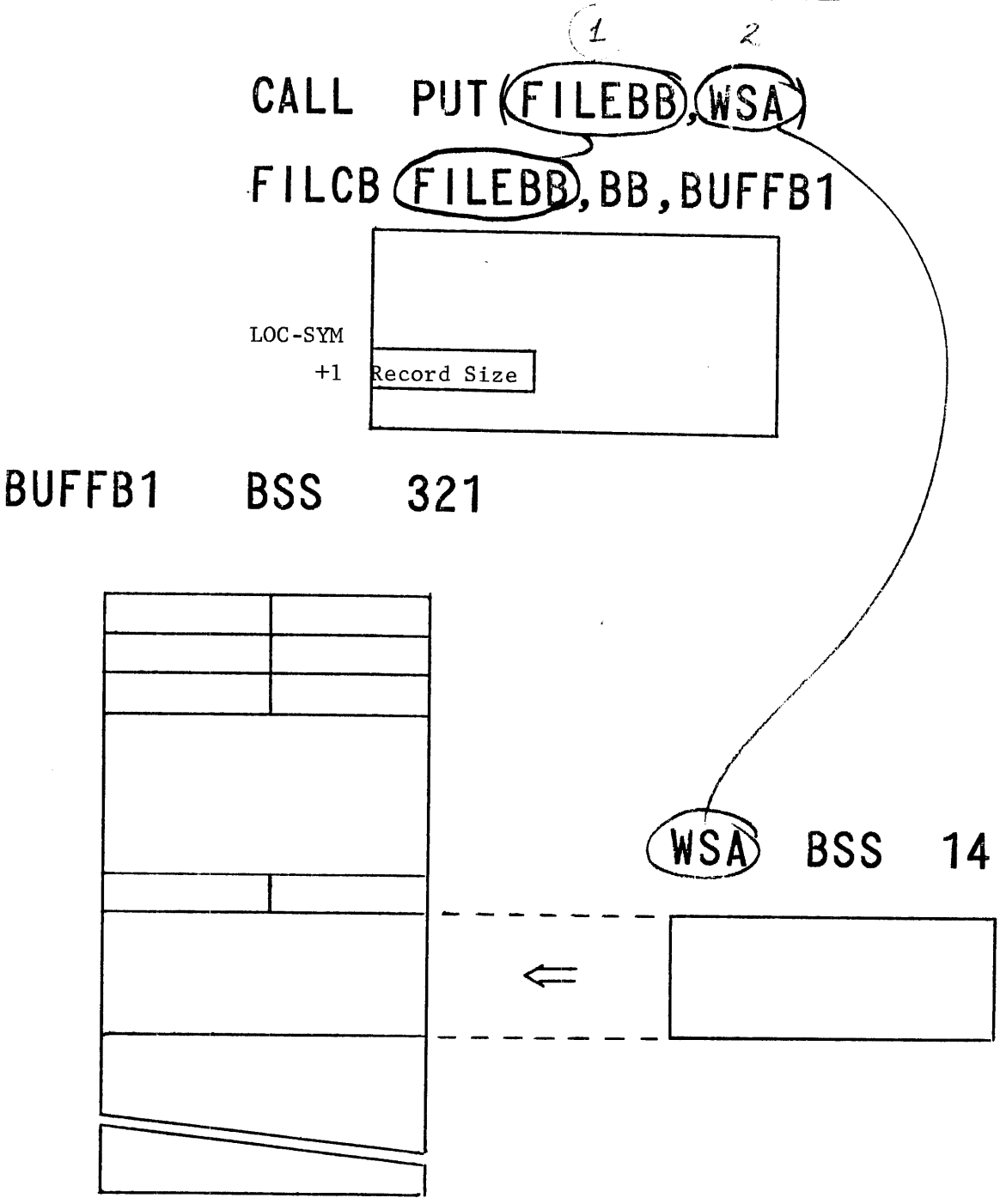


```
$         DATA   AA

$         TAPE   BB
```

GET LOGICAL RECORD

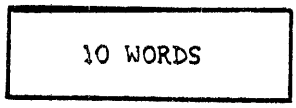


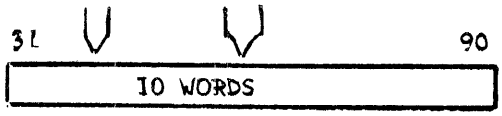
WRITE LOGICAL RECORD



INITIALIZE EDITING PARAMETERS

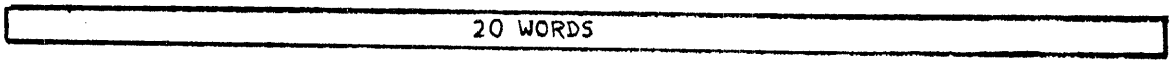
```
        CALL   IOEDIT(LLIST,5)

LLIST   ZERO   HEAD1,1
        ZERO   HEAD2,2
        ZERO   PCHCD,3
        ZERO   RPCD,7
        ZERO   PGNO,5

HEAD1   BCI
```



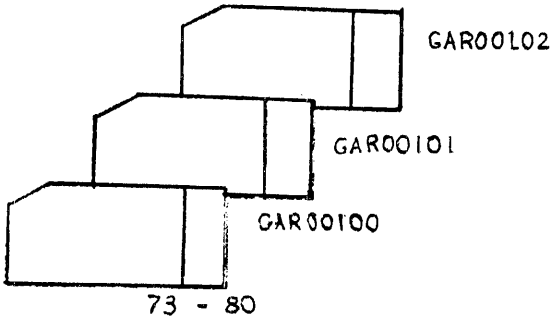


39686 01 03-16-70 19.268

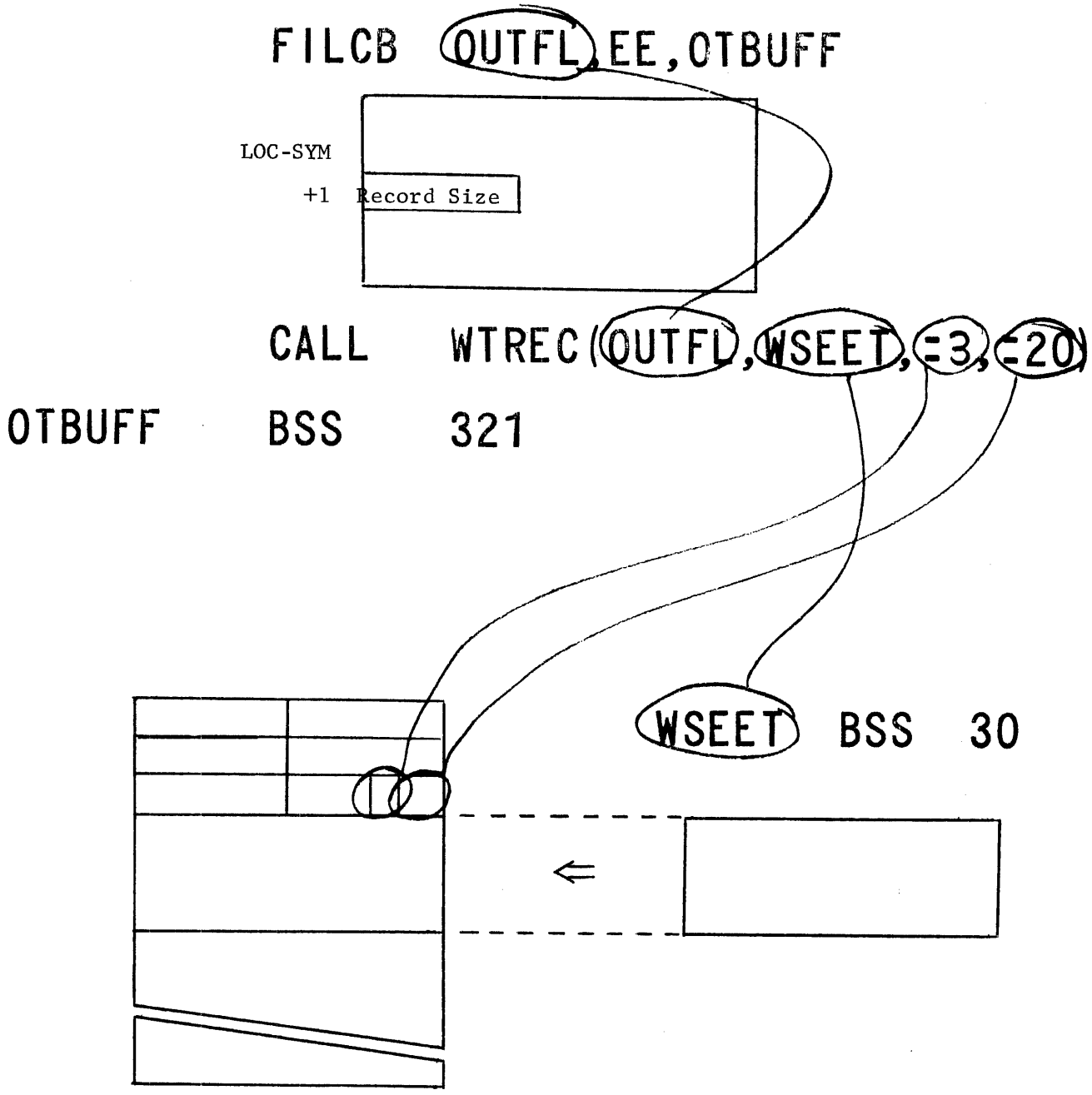
PAGE 00



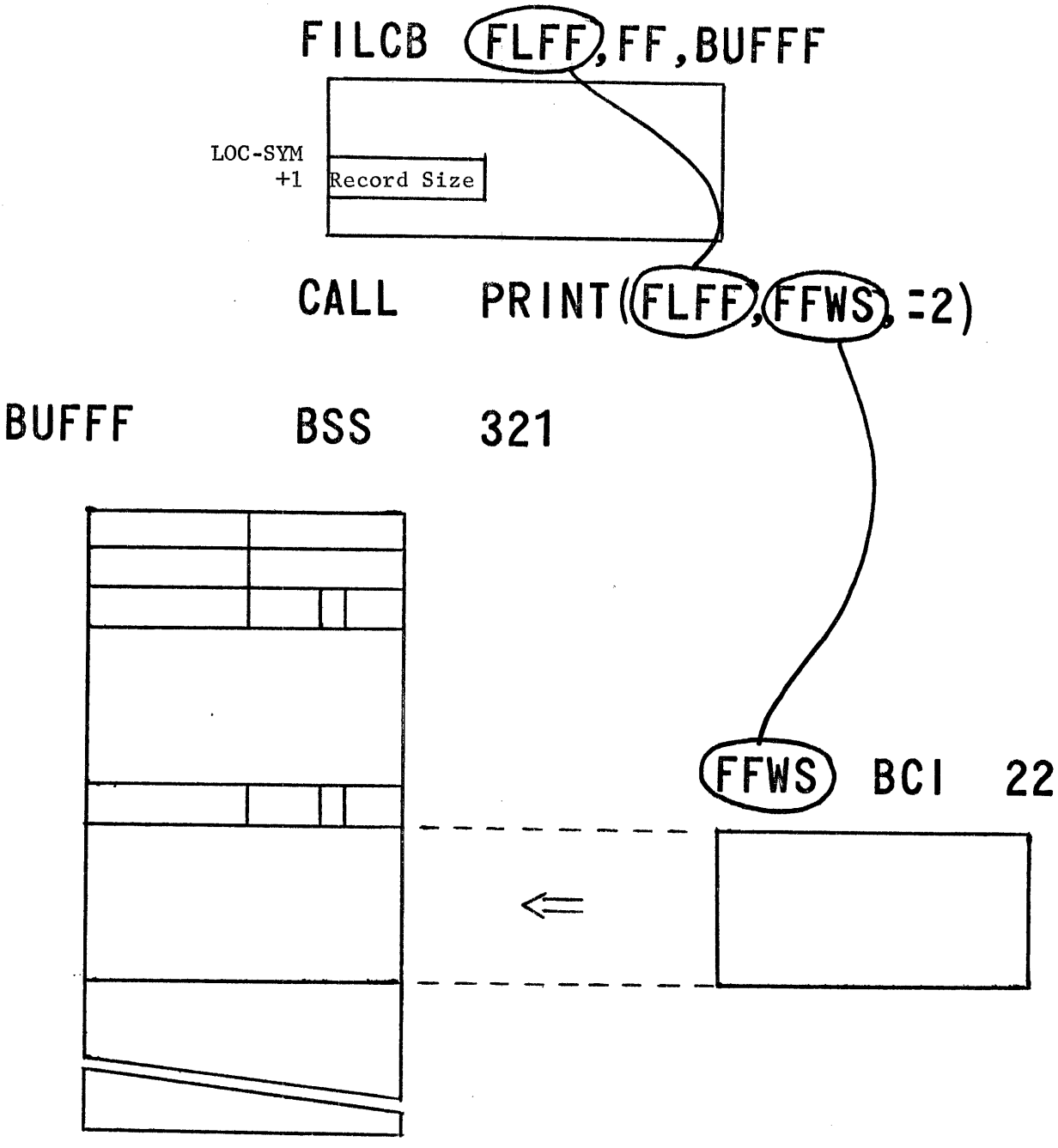

```
HEAD2   BCI
PCHCD   BCI    2,GAR00100
```



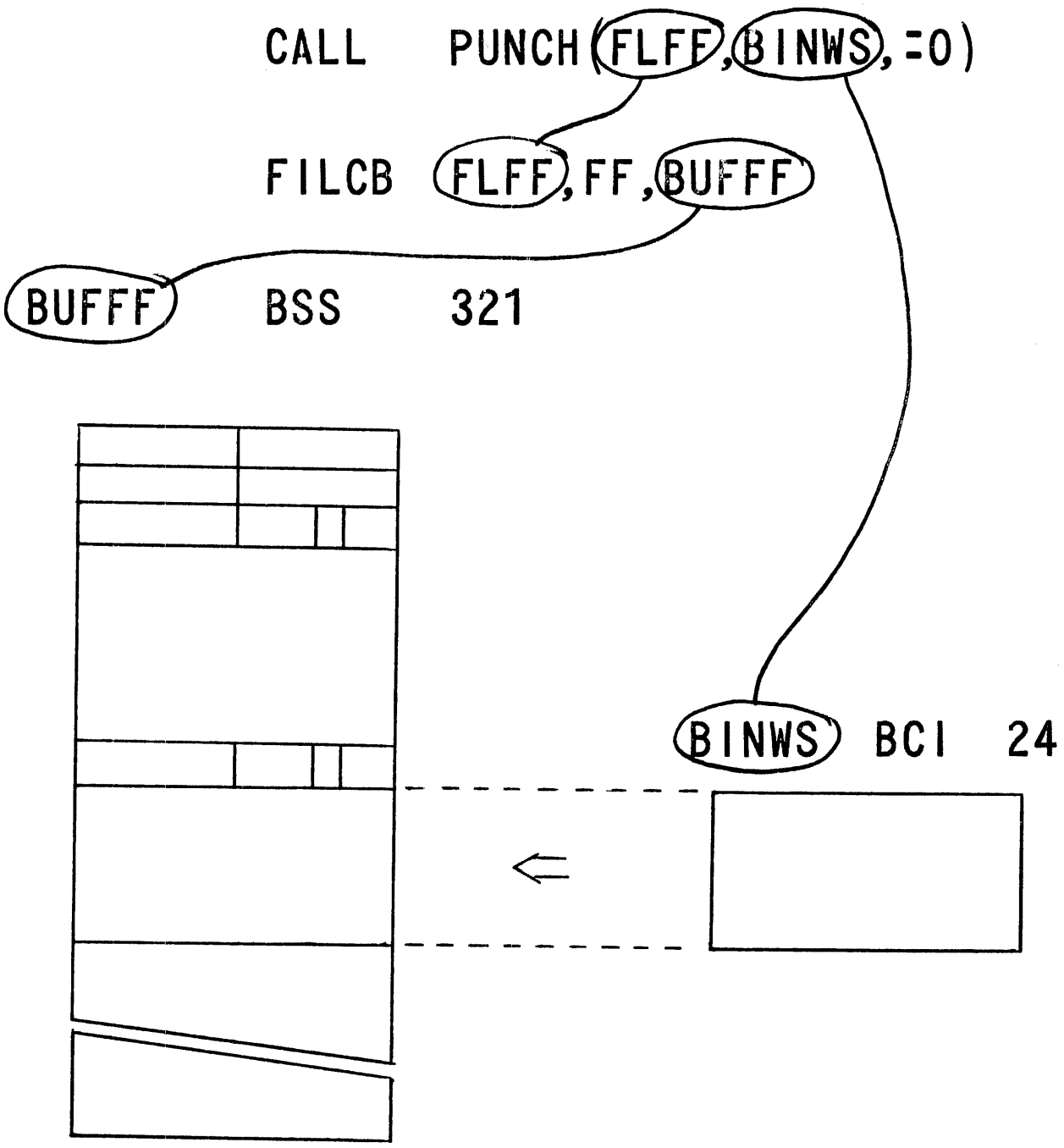
WRITE LOGICAL RECORD



WRITE LOGICAL RECORD

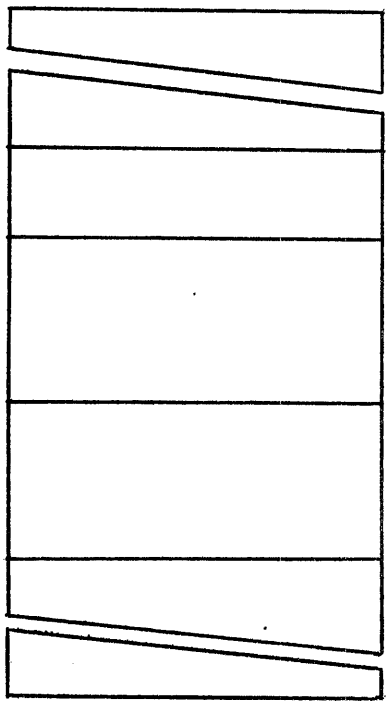


WRITE LOGICAL RECORD

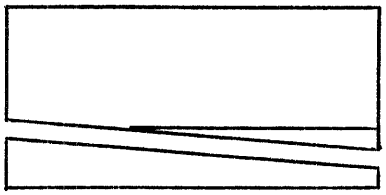


PHYSICAL RECORD PROCESSING
SEQUENTIAL FILES

```
FILCB  PHINS,SI
CALL   READ(PHINS,CNTA,INSCC)
CNTA   IOTP   AREA3,25
       IOTP   AREA1,50
       IOTD   AREA2,50
```



```
AREA1  BSS  100
```

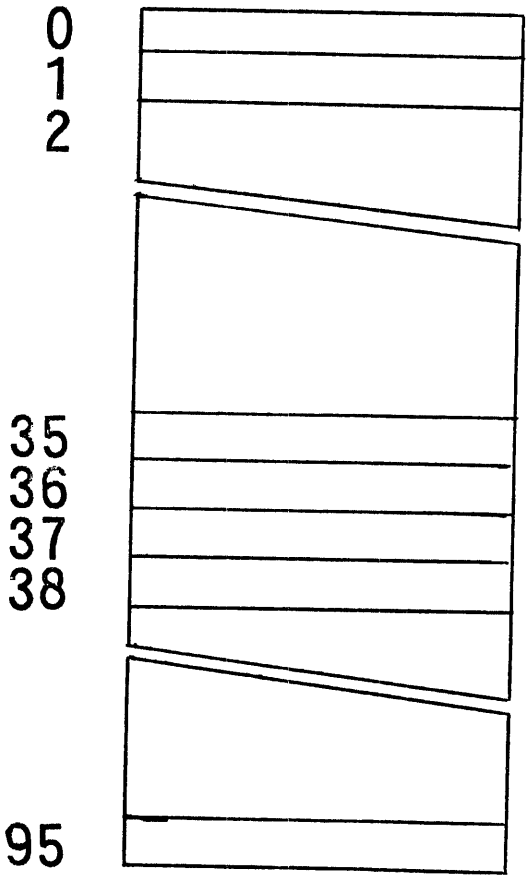


```
AREA2  BSS  50
```

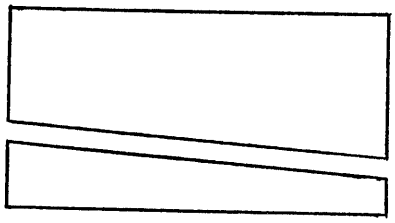
```
AREA3  BSS  25
```

PHYSICAL RECORD PROCESSING
RANDOM FILES

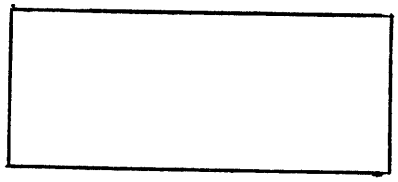
```
        FILCB   PHINR,RI

        CALL    READ(PHINR,CNTC)

CNTC    DEC     35
        IOTP    AREA8,40
        IOTP    AREA7,80
        IOTD    AREA9,40
```



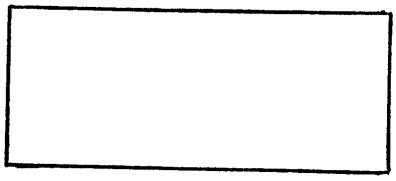

```
AREA7   BSS   80
```



```
AREA8   BSS   40
```



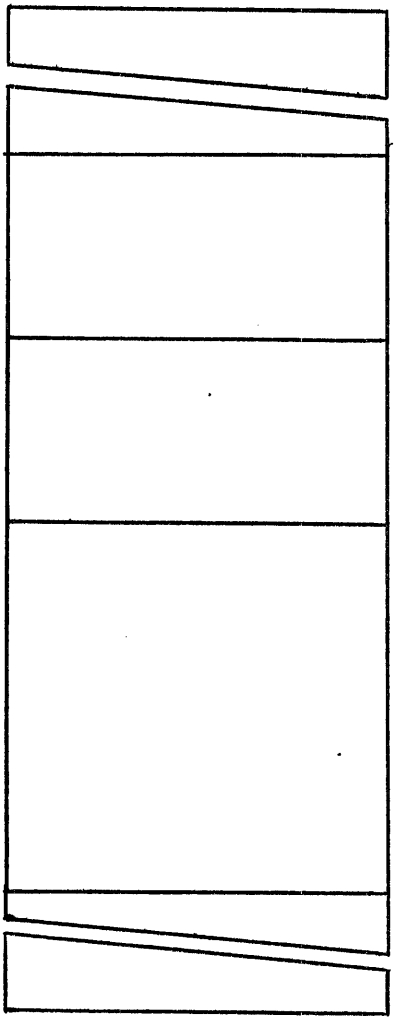
```
AREA9   BSS   40
```



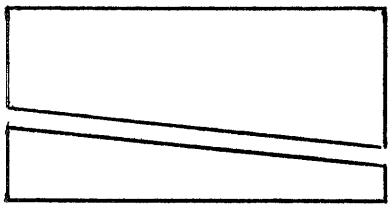
```
$  DISC  RI,X1S,1R
```

PHYSICAL RECORD PROCESSING
SEQUENTIAL FILES

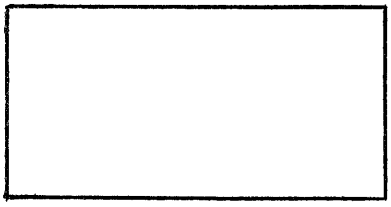
```
FILCB  PHOTS,SO

CALL   WRITE(PHOTS,CNTB)

CNTB   IOTP   AREA5,25
       IOTP   AREA4,50
       IOTD   AREA6,25
```



```
AREA4  BSS  50
```



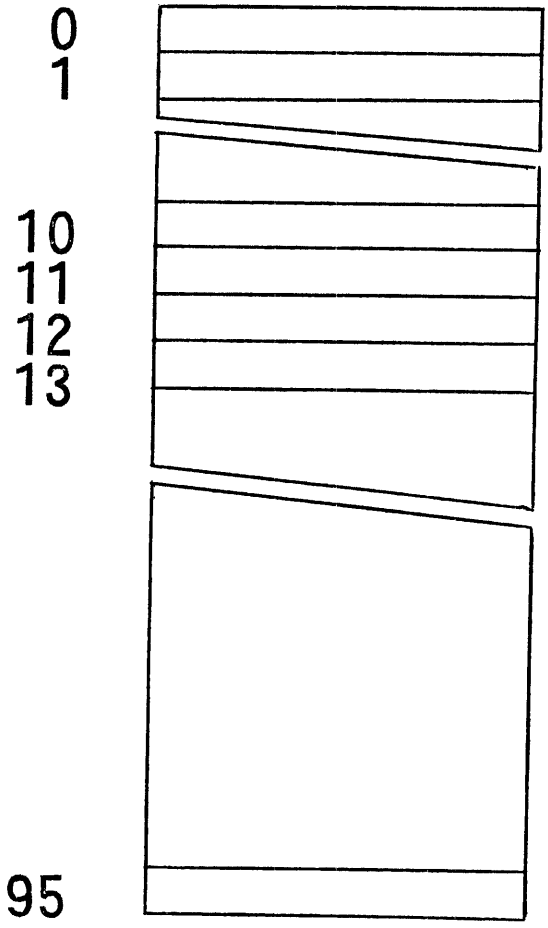
```
AREA5  BSS  25
```



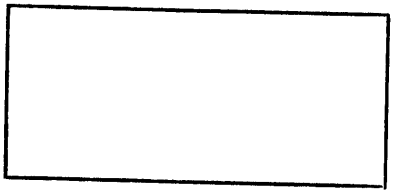
```
AREA6  BSS  25
```

PHYSICAL RECORD PROCESSING
RANDOM FILES

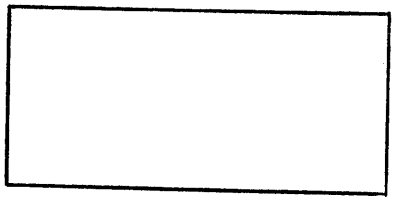
```
FILCB  PHOTR,R0

CALL   WRITE(PHOTR,CNTD)

CNTD   DEC    10
       IOTP   AREA12,80
       IOTP   AREA10,40
       IOTD   AREA11,40
```



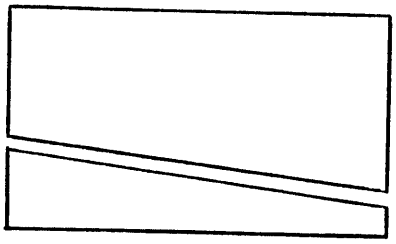

```
AREA10  BSS  40
```



```
AREA11  BSS  40
```



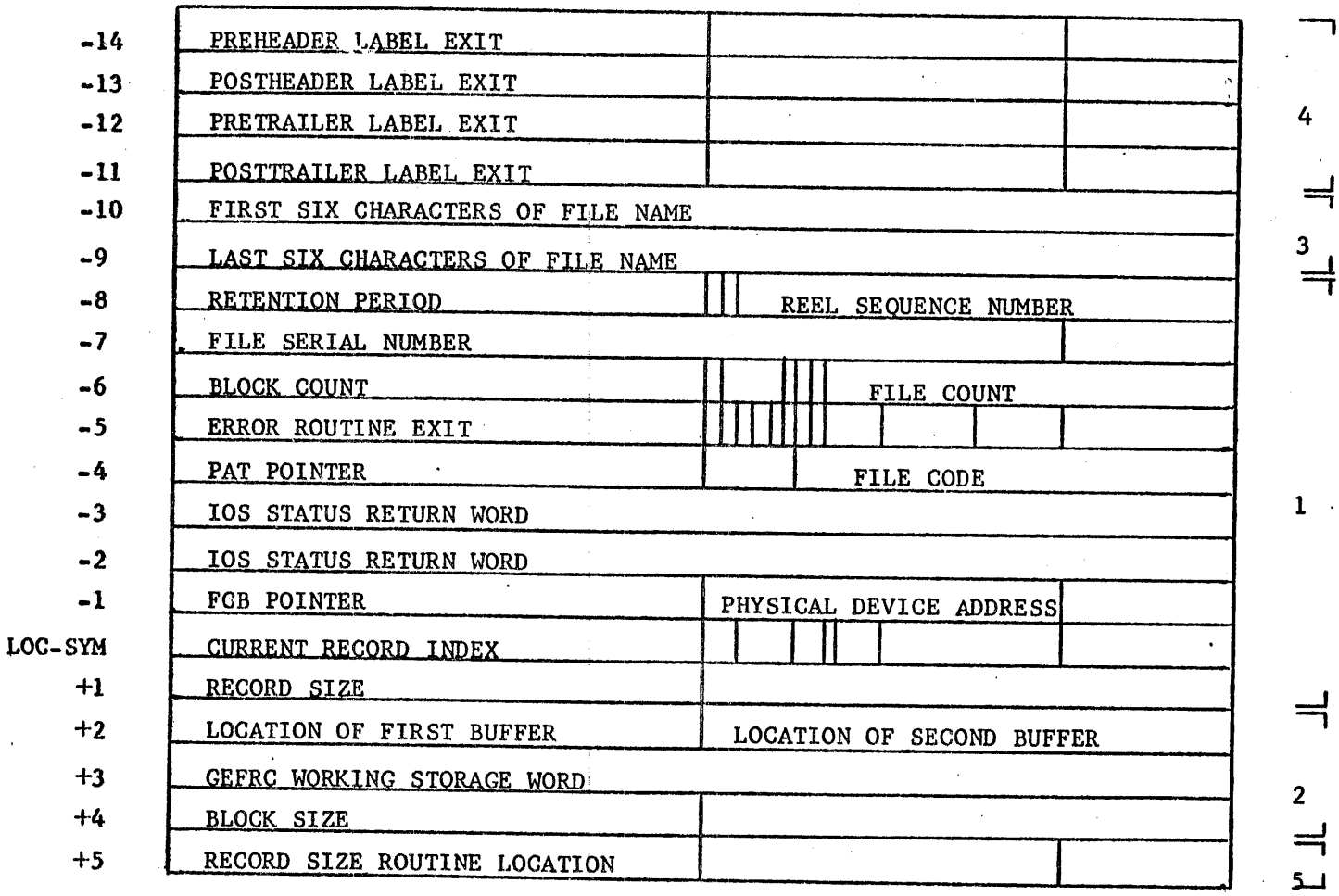
```
AREA12  BSS  80
```



```
$  DISC  R0,X1S,1R
```

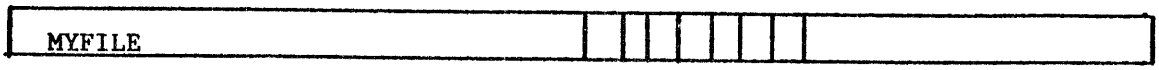
FILE CONTROL BLOCK

FILCB MYFILE,AA,BUFF1



FILE DESIGNATOR WORD

FWD VFD 18/MYFILE,18/0



FILE CONTROL BLOCK MACRO EXAMPLE

```
                                1         TTL     BASIC SA COURSE - GEFRC EXAMPLES
                                2         TTLS    FILE CONTROL BLOCK MACRO EXAMPLE
                                3         SYMDEF  GO
000008  000000 0110 00  000     4 GO      NOP
                                5 *
                                6 *
                                7 *     EXAMPLE OF COMPLETE FILE CONTROL
                                8 *     BLOCK MACRO-INSTRUCTION
                                9 *
                               10 *
000276                         11         JSE     COUNT
000276                         12         FILCB   MYFILE,          SYMBOLIC NAME OF FILE CONTROL BLOCK
000276                         13         ETC     ZZ,              FILE CODE
000276                         14         ETC     BUFF1,           FIRST BUFFER
000276                         15         ETC     BUFF2,           SECOND BUFFER
000276                         16         ETC     320,             MAXIMUM BLOCK SIZE
000276                         17         ETC     0,               RECORD FORM
000276                         18         ETC     ,                RECORD SIZE
000276                         19         ETC     ,                NO BLOCK SERIAL NUMBERS
000276                         20         ETC     ,                ERROR ROUTINE
000276                         21         ETC     ,                NO STANDARD LABLES
000276                         22         ETC     0,               RECORDING MODE
000276                         23         ETC     0,               RECORDING DENSITY
000276                         24         ETC     0,               MULITFILE REEL
000276                         25         ETC     0,               RETENTION PERIOD
000276                         26         ETC     ,                PREHEADER LABEL ROUTINE
000276                         27         ETC     ,                POSTHEADER LABEL ROUTINE
000276                         28         ETC     ,                PRETRAILER LABEL ROUTINE
000276                         29         ETC     ,                POSTTRAILER LABEL ROUTINE
000276                         30         ETC     MYFILE,          FILENAME
000276                         31         ETC     0,               GEPR OVERRIDE
000276                         32         ETC     0                PARTITIONED RECORD
```

FILE DESIGNATOR WORD AND BUFFERS

```
                                          35 *
                                          36 *
                                          37 *     EXAMPLE OF FILE DESIGNATOR WORD
                                          38 *
                                          39 *
                                          40 ZZWD   VFD     18/MYFILE,     SYMBOLIC NAME OF FILE CONTROL BLOCK
                                          41        ETC     1/1,           INPUT/OUTPUT INDICATOR
                                          42        ETC     1/0,           REWIND ON OPEN
                                          43        ETC     2/0,           REWIND ON CLOSE
                                          44        ETC     1/0,           PRIME INPUT BUFFER OPTION
                                          45        ETC     1/0,           PUT SIZE ON OUTPUT OPTION
                                          46        ETC     1/0,           REQUIRED FILE OPTION
000315 000310400000 010                   47        ETC     11/0           FILE POSITIONING VALUE
                                          48 *
                                          49 *
                                          50 *     RESERVE BUFFER SPACE
                                          51 *
                                          52 *     EACH BUFFER MUST BE ONE WORD LARGER THAN
                                          53 *     THE PHYSICAL RECORD FOR A BUFFER CONTROL WORD
                                          54 *     TO CONTROL THE DATA FLOW TO OR FROM THE BUFFER
                                          55 *
                                          56 *
000316                                    57 BUFF1  BSS     321            FIRST BUFFER
001047                                    58 BUFF2  BSS     321            SECOND BUFFER
```

CALL: OPEN, CLOSE, GET, PUT MACRO-INSTRUCTIONS

```
                                        61 *
                                        62 *
                                        63 *    THE FOLLOWING IS A DESCRIPTION OF A FILE CONTROL
                                        64 *    BLOCK FOR A CARD FILE USING A    $   DATA   AA
                                        65 *    FILE CONTROL CARD.
                                        66 *
                                        67 *    BECAUSE OF THE USE OF THE  $   DATA   AA   FILE
                                        68 *    CONTROL CARD THIS FILE WILL BE IN STANDARD
                                        69 *    SYSTEM FORMAT
                                        70 *
                                        71 *
                    001520              72          FILCB   FILEAA,AA,BUFFA2,BUFFA2
                                        73 *
                                        74 *
                                        75 *    FILE CONTROL BLOCK FOR OUTPUT FILE CODE BB
                                        76 *    USING STANDARD SYSTEM FORMAT WITH ONE BUFFER
                                        77 *    THIS IS A TAPE FILE USING A  $   TAPE   BB
                                        78 *    FILE CONTROL CARD
                                        79 *
                                        80 *
                    001535              81          FILCB   FILEBB,BB,BUFFB1
                                        82 *
                                        83 *
                                        84 *    FILE DESIGNATOR WORDS ASSOCIATED WITH FILEAA AND FILEBB
                                        85 *
                                        86 *
001552  001530000000        010         87 AWD:     VFD     18/FILEAA,18/0
001553  001545400000        010         88          VFD     18/FILEBB,1/1,17/0
                                        89 *
                                        90 *
                                        91 *    BUFFERS FOR FILEAA AND FILEBB
                                        92 *
                                        93 *
                    001554              94 BUFFA1   BSS     321             FIRST BUFFER FOR FILEAA
                    002255              95 BUFFA2   BSS     321             SECOND BUFFER FOR FILEAA
                    002756              96 BUFFB1   BSS     321             BUFFER FOR FILEBB
                                        97 *
                                        98 *
                                        99 *    WORKING STORAGE AREA INTO WHICH GEFRC
                                       100 *    WILL MOVE THE LOGICAL RECORD
                                       101 *
                                       102 *
                    003457             103 WSA     BSS     14
                    000001             104         JSE
```

CALL OPEN, CLOSE, GET, PUT MACRO-INSTRUCTIONS

```
                                        107 *
                                        108 *
                                        109 *     INITIALIZE TWO FILES BEGINNING AT A LIST
                                        110 *     OF FILE DESIGNATOR WORDS LOCATED AT AWD
                                        111 *
                                        112 *
000001  040000700800      030           113         CALL OPEN(AWD,2)
000002  080006710800      010
000003  007210000161      010
000004  081552000800      010
000005  080002000800      000
```

CALL OPEN, CLOSE, GET, PUT MACRO-INSTRUCTIONS

```
                                        116 *
                                        117 *
                                        118 *    OBTAIN THE NEXT LOGICAL RECORD FROM THE
                                        119 *    BUFFER OF FILEAA AND MOVE IT TO WORKING
                                        120 *    STORAGE AREA WSA
                                        121 *
                                        122 *
000006  010000701000    030             123        CALL GET(FILEAA,ENDAA,WSA)
000007  000014710000    010
000010  007210000173    010
000011  001530000800    010
000012  000052000000    010
000013  003457000000    010
                                        124 *
                                        125 *
                                        126 *    OBTAIN THE FIRST LOGICAL RECORD IN THE NEXT
                                        127 *    PHYSICAL RECORD FROM FILEAA AND MOVE
                                        128 *    IT TO WORKING STORAGE AREA WSA
                                        129 *
                                        130 *
000014  100000701800    030             131        CALL GETBK(FILEAA,ENDAA,WSA)
000015  000022710000    010
000016  007210000203    010
000017  001530000000    010
000020  000052000800    010
000021  003457000000    010
```

CALL: OPEN, CLOSE, GET, PUT MACRO-INSTRUCTIONS

```
                                    134 *
                                    135 *
                                    136 *    TO INSERT THE NEXT LOGICAL RECORD INTO THE
                                    137 *    NEXT AVAILABLE SPACE IN THE OUTPUT BUFFER
                                    138 *
                                    139 *    THE SIZE OF THE LOGICAL RECORD MUST
                                    140 *    BE PLACED IN WORD PLUS 1 OF THE
                                    141 *    FILE CONTROL BLOCK WHEN CALL PUT MACRO IS USED
                                    142 *
                                    143 *         SEVERAL METHODS CAN BE USED
                                    144 *         TO PLACE A BINARY VALUE IN THE
                                    145 *         RECORD SIZE OF THE FILE CONTROL BLOCK
                                    146 *
                                    147 *      (1,)  USING A LITERAL
                                    148 *
                                    149 *
000022  000016 2260 03  000       150           LDX6    =14,DU
000023  001546 7?60 00  010       151           STX6    FILEBB+1
                                    152 *
                                    153 *
                                    154 *      (2,)  OR USING THE CONTENTS OF ANOTHER FIELD
                                    155 *
                                    156 *
000024  003475 2260 00  010       157           LDX6    FOR
000025  001546 7?60 00  010       158           STX6    FILEBB+1
                                    159 *
                                    160 *
                003475              161           JSE     COUNT
003475  010420202820    000       162 FOR       VFD     18/14,18/0
                000026              163           JSE
                                    164 *
                                    165 *
                                    166 *    INSERT THE NEXT LOGICAL RECORD FROM AREA WSA INTO
                                    167 *    THE NEXT AVAILABLE SPACE OF THE OUTPUT BUFFER
                                    168 *
                                    169 *
000026  020000701000      030     170           CALL PJT(FILEBB,WSA)
000027  000033710000      010
000030  00721000:252      010
000031  001545000000      010
000032  003457000000      010
                                    171 *
                                    172 *
                                    173 *    TO INSERT THE NEXT LOGICAL RECORD AS THE FIRST LOGICAL
                                    174 *    RECORD OF THE NEXT PHYSICAL RECORD FROM AREA WSA
                                    175 *
                                    176 *
000033  130000701000      030     177           CALL PJTBK(FILEBB,WSA)
000034  00004071:000      010
```

```
000036  001545000000    010
000037  003457000000    010
```

CALL OPEN, CLOSE, GET, PUT MACRO-INSTRUCTIONS

```
                                        180 *
                                        181 *
                                        182 *    THIS SAME OPERATION COULD HAVE BEEN
                                        183 *    ACCOMPLISHED EASIER USING THE CALL COPY MACRO
                                        184 *
                                        185 *
000040  010000701000    030     186         CALL GET(FILEAA,ENDAA)
000041  000045710000    010
000042  007210007272    010
000043  001530000000    010
000044  000052000000    010
000045  030000701000    030     187         CALL COPY(FILEBB,FILEAA)
000046  000052710000    010
000047  007210000273    010
000050  001545000000    010
000051  001530000000    010
```

CALL OPEN, CLOSE, GET, PUT MACRO=INSTRUCTIONS

```
                                    190 *
                                    191 *
                                    192 *     THIS IS THE END-OF-FILE ROUTINE WHERE
                                    193 *     CONTROL WILL BE TRANSFERRED WHEN ALL
                                    194 *     OF THE INPUT RECORDS HAVE BEEN PROCESSED
                                    195 *     AND A 17 OCTAL END-OF-FILE MARK IS ENCOUNTERED
                                    196 *
                                    197 *     THE CALL CLOSE MACRO DISCONNECTS THE FILES
                                    198 *     SPECIFIED BY THE FILE DESIGNATOR WORDS
                                    199 *
                                    200 *
000052  070000701000      030       201 ENDAA  CALL CLOSE(AWD,2)
000053  000057710000      010
000054  007210000311      010
000055  001552000000      010
000056  000002000000      000
000057  000007 0010 00    000       202        NAME     GEFINI
```

PROCESSING FIXED LENGTH RECORDS

```
                                    205 *
                                    206 *
                                    207 *    THIS EXAMPLE SHOWS HOW TO PROCESS FIXED
                                    208 *    LENGTH RECORDS AND MOVE LOGICAL RECORDS
                                    209 *    FROM FILECC TO FILEDD
                                    210 *
                                    211 *    FILE CONTROL BLOCK, FILE DESIGNATOR WORD,
                                    212 *    BUFFER, AND WORKING STORAGE AREA FOR
                                    213 *    INPUT FILECC USING A  $   TAPE   CC
                                    214 *    FILE CONTROL CARD
                                    215 *
                                    216 *
                  003476            217         JSE     COUNT
                  003476            218         FILCB   FILECC,CC,BUFFC1,,101,1,20
003513  003506000000    010         219 CWD     VFD     18/FILECC,18/0
                  003514            220 BUFFC1  BSS     102             BUFFER FOR FILEGC
                  003662            221 WSC     BSS     20              WORKING STORAGE AREA
                                    222 *
                                    223 *
                                    224 *    FILE CONTROL BLOCK, FILE DESIGNATOR WORD,
                                    225 *    AND BUFFER FOR OUTPUT FILE DD USING
                                    226 *    A $   TAPE  DD  CONTROL CARD
                                    227 *
                                    228 *
                  003706            229         FILCB   FILEDD,DD,BUFFD1,,201,1,20
003723  003716400000    010         230 DWD     VFD     18/FILEDD,1/1,17/0
                  003724            231 BUFFD1  BSS     202             BUFFER FOR FILEDD
                  000060            232         JSE
```

PROCESSING FIXED LENGTH RECORDS

```
                                                235 *
                                                236 *
                                                237 *     THIS IS AN EXAMPLE OF CODING TO ESTABLISH
                                                238 *     RECORD SIZE OF INPUT AND OUTPUT FILES
                                                239 *     AND THE TRANSFER OF DATA TO AND FROM
                                                240 *     WORKING STORAGE AREA WSC
                                                241 *
                                                242 *
000060  040000701000    030     243             CALL OPEN(CWD,1)
000061  000065710000    010
000062  007210000363    010
000063  003513000000    010
000064  000001000000    000
000065  040000701000    030     244             CALL OPEN(DWD,1)
000066  000072710000    010
000067  007210000364    010
000070  003723000000    010
000071  000001000000    000
000072  000024 2250 03  000     245             LDX5    =20,DU          PLACE RECORD SIZE OF
000073  003507 7450 00  010     246             STX5    FILECC+1        INPUT FILE IN FCB
000074  010000701000    030     247             CALL GET(FILECC,ENDC,WSC)
000075  000102710000    010
000076  007210000367    010
000077  003506000000    010
000100  000111000000    010
000101  003662000000    010
000102  000024 2250 03  000     248             LDX5    =20,DU          PLACE RECORD SIZE OF
000103  003717 7450 00  010     249             STX5    FILEDD+1        OUTPUT FILE IN FCB
000104  020000701000    030     250             CALL PUT(FILEDD,WSC)
000105  000111710000    010
000106  007210000372    010
000107  003716000000    010
000110  003662000000    010
000111  070000701000    030     251 ENDC        CALL CLOSE(CWD,1)
000112  000116710000    010
000113  007210000373    010
000114  003513000000    010
000115  000001000000    000
000116  070000701000    030     252             CALL CLOSE(DWD,1)
000117  000123710000    010
000120  007210000374    010
000121  003723000000    010
000122  000001000000    000
```

CALL WTREC EXAMPLES

```
                                255 *
                                256 *
                                257 *     THE CALL WTREC CAN BE USED TO CREATE A
                                258 *     LOGICAL RECORD AND DIRECT IT TO ANY OUTPUT
                                259 *     MEDIA DEPENDING ON THE DEVICE SPECIFIED
                                260 *     ON THE FILE CONTROL CARD
                                261 *
                                262 *     FILE CONTROL BLOCK, FILE DESIGNATOR WORD,
                                263 *     AND BUFFER FOR OUTPUT FILE OUTFL
                                264 *
                                265 *
                    004236      266           JSE     COUNT
                    004236      267           FILCB   OUTFL,EE,OTBUFF
                                268 *
004253  004266400000    010     269 OTWD      VFD     18/OUTFL,1/1,17/0
                                270 *
                    004254      271 OTBUFF    BSS     321             BUFFER FOR OUTFL
                    000123      272           USE
```

CALL WTREC EXAMPLES

```
                                        275 *
                                        276 *
                                        277 *     THIS IS AN EXAMPLE USING CALL WTREC
                                        278 *     TO CREATE A LOGICAL RECORD AND DIRECT
                                        279 *     IT TO TAPE OR DISC DEPENDING ON THE
                                        280 *     FILE CONTROL CARD
                                        281 *
                                        282 *
000123  000036 2350 03  000           283           LDA     =30,DU          PLACE RECORD SIZE IN
000124  004247 7750 00  010           284           STA     OUTFL+1         FILE CONTROL BLOCK
000125  150000701000    030           285           CALL WTREC(OUTFL,WSEET,=5,=20)
000126  000134710000    010
000127  007210000435    010
000130  004246000000    010
000131  004755000000    010
000132  007212000000    010
000133  007213000000    010
                                        286 *
                                        287 *
                                        288 *     A LOGICAL RECORD WILL BE CREATED
                                        289 *     USING THE CONTENTS OF THIS
                                        290 *     FIELD
                                        291 *
                                        292 *
                004755                  293           JSE     COUNT
                004755                  294 WSEET     BSS     30
                000134                  295           JSE
```

CALL WTREC EXAMPLES

```
                                        298 *
                                        299 *
                                        300 *     THE FOLLOWING EXAMPLES CREATE A LOGICAL RECORD
                                        301 *     USING THE CALL WTREC AND DIRECT THE RECORD
                                        302 *     TO THE PRINTER OR CARD PUNCH THROUGH THE
                                        303 *     USE OF   $   SYSOUT   EE  FILE CONTROL CARD
                                        304 *
                                        305 *     THIS IS AN EXAMPLE USING CALL WTREC
                                        306 *     TO PUNCH A BINARY CARD
                                        307 *
                                        308 *
000134  000033 2350 03   000           309          LDA     =27,DU          PLACE THE RECORD SIZE IN
000135  004247 7550 00   010           310          STA     OUTFL+1         FILE CONTROL BLOCK
000136  150C007C1000     030           311          CALL WTREC(OUTFL,DTSTR,=1,REP22)
000137  000145710000     010
000140  007210009467     010
000141  004246000000     010
000142  005013000000     010
000143  007214000000     010
000144  005046000000     010
                                        312 *
                                        313 *
                 005013                 314          USE     COUNT
005013  202063303162     000           315 DTSTR    BCI     9,  THIS INFORMATION
005014  203145264651     000
005015  442163319645     000
005016  202020202020     000
005017  202020202020     000
005020  202020202020     000
005021  202020202020     000
005022  202020202020     000
005023  202020202020     000
005024  202066314343     000           316          BCI     9,  WILL BE PUNCHED
005025  202225209764     000
005026  452330252420     000
005027  202020202020     000
005030  202020202020     000
005031  202020202020     000
005032  202020202020     000
005033  202020202020     000
005034  202020202020     000
005035  202031452022     000           317          BCI     9,  IN BINARY
005036  314521517020     000
005037  202020202020     000
005040  202020202020     000
005041  202020202020     000
005042  202020202020     000
005043  202020202020     000
005044  202020202020     000
005045  202020202020     000
```

CALL WTREC EXAMPLES

```
                                318 *
                                319 *
                                320 *     SYSOUT WILL SORT TOGETHER ALL RECORDS
                                321 *     HAVING THE SAME REPORT CODE
                                322 *
                                323 *
005046  000000000022    000    324 REP22  DCT      000000000022    REPORT CODE OF 22
                 000165         325        USE
```

CALL WTREC EXAMPLES

```
                                        328 *
                                        329 *
                                        330 *     THIS IS AN EXAMPLE USING CALL WTREC
                                        331 *     TO PUNCH A HOLLERITH CARD
                                        332 *
                                        333 *
000145  000016 2350 03   000    334          LDA     =14,DU
000146  004247 7350 00   010    335          STA     OUTFL+1
000147  150000705000     030    336          CALL WTREC(OUTFL,WSPUN,=2,REP22)
000150  000156710000     010
000151  007210000520     010
000152  004246007000     010
000153  005047000000     010
000154  007215000000     010
000155  005046000000     010
                                        337 *
                                        338 *
                  005047                339          JSE     COUNT
005047  202063303162     000    340 WSPUN BCI     9,  THIS INFORMATION WILL BE
005050  203145264651     000
005051  442163313645     000
005052  206631439320     000
005053  222520206020     000
005054  202020202020     000
005055  202020202020     000
005056  202020206020     000
005057  202020202020     000
005060  202047645523     000    341          BCI     5,  PUNCHED IN HOLLERITH
005061  302524201145     000
005062  203046433325     000
005063  513163306020     000
005064  202020206020     000
                  000156                342          JSE
```

CALL WTREC EXAMPLES

```
                                        345 *
                                        346 *
                                        347 *     THIS IS AN EXAMPLE JSING CALL WTREC
                                        348 *     TO PRINT A LINE
                                        349 *
                                        350 *
000156  000025 2350 03   000    351            LDA     =21,DU          MOVE RECORD SIZE INCLUDING
000157  004247 7350 00   010    352            STA     OUTFL+1         THE SLEW CHARACTER TO THE FCB
000160  150000701000     030    353            CALL WTREC(OUTFL,PRWS,PRLIN,REP22)
000161  000167710000     010
000162  007210003541     010
000163  004246000000     010
000164  005065000000     010
000165  005112000000     010
000166  005046000000     010
                                        354 *
                                        355 *
                 005065                 356            JSE     COUNT
005065  202063303162     000    357 PRWS    BCI     9,  THIS INFORMATION
005066  203145264651     000
005067  442163314645     000
005070  202020206020     000
005071  202020206020     000
005072  202020206020     000
005073  202020206020     000
005074  202020206020     000
005075  202020206020     000
005076  202066315343     000    358            BCI     9,  WILL BE PRINTED
005077  202225204751     000
005100  314563252420     000
005101  202020206020     000
005102  202020206020     000
005103  202020206020     000
005104  202020206020     000
005105  202020206020     000
005106  202020206020     000
005107  202020206020     000    359            BCI     2,
005110  202020206020     000
005111  770400000000     000    360            OCT     770400000000    PRINTER SLEW CHARACTER
                                        361 *
                                        362 *
                                        363 *     A MEDIA CODE OF 3 DIRECTS THIS
                                        364 *     RECORD THROUGH SYSOJT TO THE PRINTER
                                        365 *
                                        366 *
005112  000000 000003    000    367 PRLIN  ZERO    0,3             MEDIA CODE
```

CALL PRINT EXAMPLES

```
                                        370 *
                                        371 *
                                        372 *     THIS IS AN EXAMPLE USING CALL PRINT
                                        373 *     TO WRITE A PRINT LINE
                                        374 *
                                        375 *
                          005113        376           FILCB   FLFF,FF,BUFFF
005130  005123400000      010           377 FFWD      VFD     18/FLFF,1/1,17/0
                          005131        378 BUFFF:    BSS     321
                          000167        379           JSE
                                        380 *
                                        381 *
000167  000026 2260 03    000           382           LDX6    =22,DU
000170  005124 7460 00    010           383           STX6    FLFF+1
000171  110000701000      030           384           CALL PRINT(FLFF,FFWS,=2)
000172  000177710000      010
000173  007210006600      010
000174  005123000000      010
000175  005632000000      010
000176  007215000000      010
                                        385 *
                                        386 *
                          005632        387           JSE     COUNT
005632  202063303162      000           388 FFWS      BCI     9,  THIS INFORMATION
005633  203145264651      000
005634  442163314645      000
005635  202020202020      000
005636  202020202020      000
005637  202020202020      000
005640  202020204020      000
005641  202020202020      000
005642  202020204020      000
005643  202063143430      000           389           BCI     9,  WILL BE PRINTED
005644  202225204751      000
005645  314563254420      000
005646  202020202020      000
005647  202020202020      000
005650  202020202020      000
005651  202020204020      000
005652  202020202020      000
005653  202020202020      000
005654  202020202020      000           390           BCI     4,
005655  202020204020      000
005656  202020202020      000
005657  202020202020      000
                          000177        391           JSE
```

CALL PUNCH EXAMPLES

```
                                        394 *
                                        395 *
                                        396 *     THIS IS AN EXAMPLE JSING CALL PUNCH
                                        397 *     TO PUNCH BINARY CARDS
                                        398 *
                                        399 *
000177  120000701000      030           400           CALL PUNCH(FLFF,BINWS,=0)
000200  000205710000      010
000201  007210000620      010
000202  005123000000      010
000203  005660001000      010
000204  007216000000      010
                  005660                401           JSE     COUNT
005660  202063303162      000           402 BINWS     BCI     9,  THIS INFORMATION
005661  203145255651      000
005662  442163311645      000
005663  202020206020      000
005664  202020202020      000
005665  202020202020      000
005666  202020204020      000
005667  202020206020      000
005670  202020206020      000
005671  202066311343      000           403           BCI     9,  WILL BE PUNCHED
005672  202225204764      000
005673  452330256420      000
005674  202020204020      000
005675  202020206020      000
005676  202020202020      000
005677  202020202020      000
005700  202020204020      000
005701  202020202020      000
005702  202031456022      000           404           BCI     6,  IN BINARY
005703  314521517020      000
005704  202020202020      000
005705  202020202020      000
005706  202020202020      000
005707  202020204020      000
                  000205                405           JSE
```

CALL PUNCH EXAMPLES

```
                                        408 *
                                        409 *
                                        410 *    THIS IS AN EXAMPLE USING CALL PUNCH
                                        411 *    TO PUNCH HOLLERITH CARDS
                                        412 *
                                        413 *
000205  120000701000      030           414          CALL PUNCH(FLFF,PUNWS,=1)
000206  000213710000      010
000207  007210009636      010
000210  005123000000      010
000211  005710009000      010
000212  007214000000      010
                                        415 *
                                        416 *
                    005710              417          USE     COUNT
005710  202063303162      000           418 PUNWS    BCI     9,  THIS INFORMATION WILL BE PUNCHED
005711  203145299651      000
005712  442163312645      000
005713  206631439320      000
005714  222520479445      000
005715  233025246820      000
005716  202020202020      000
005717  202020203820      000
005720  202020201020      000
005721  202031452030      000           419          BCI     3,  IN HOLLERITH
005722  464343259131      000
005723  633020201020      000
                    000213              420          USE
```

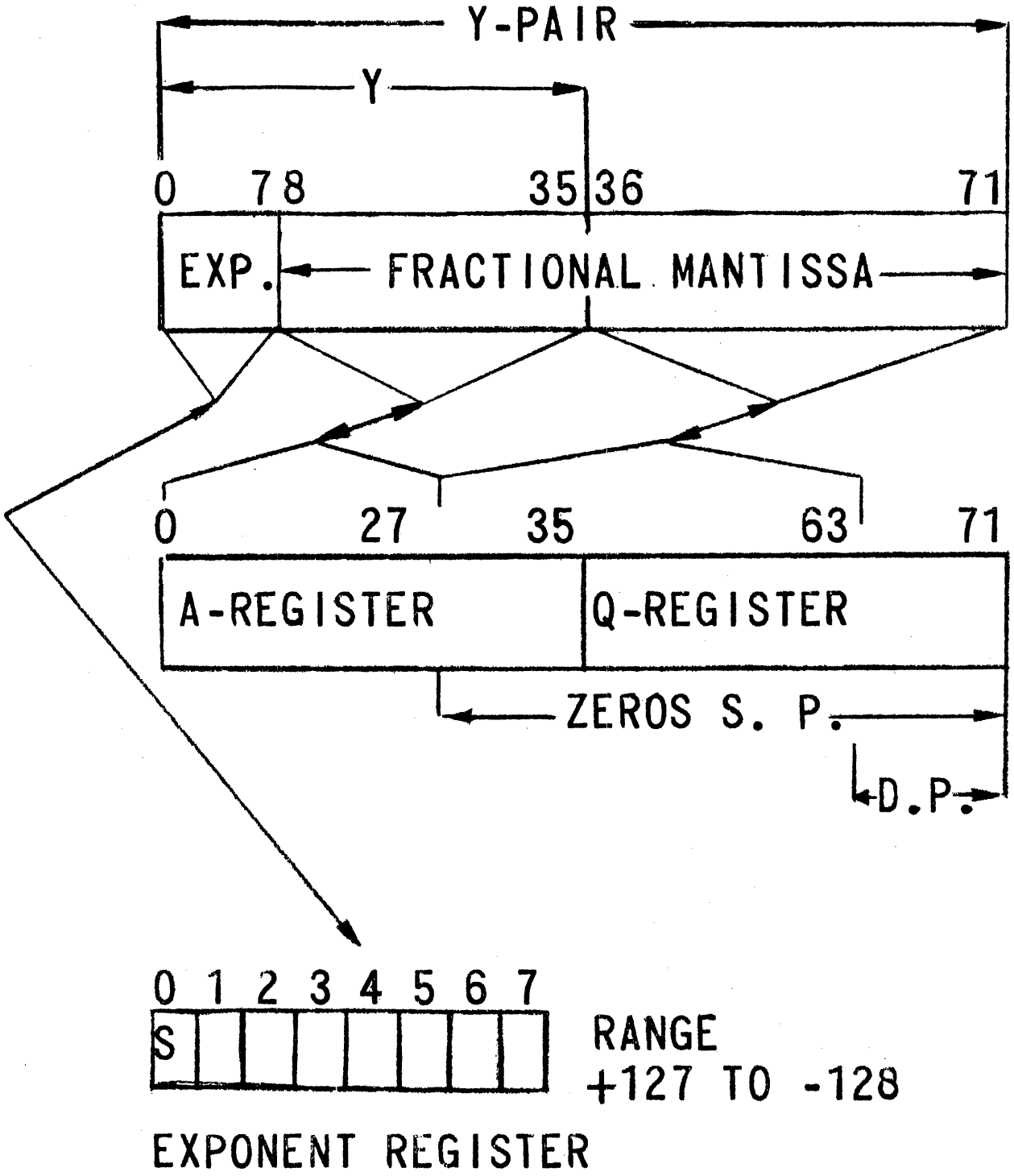
CALL IOEDIT

```
                                        423 *
                                        424 *
                                        425 *     THE CALL IOEDIT INITIALIZES PARAMETERS
                                        426 *     THAT WILL BE USED AS EDITING FUNCTIONS
                                        427 *     WHEN EITHER CALL PRINT OR CALL PUNCH
                                        428 *     ARE USED
                                        429 *
                                        430 *
000213  160000701800      030           431         CALL IOEDIT(LLIST,5)
000214  000220710000      010
000215  007210005657      010
000216  005724000000      010
000217  000005001000      000
                                        432 *
                                        433 *
                                        434 *     LIST OF CONTROL PARAMETERS AND CODES
                                        435 *
                                        436 *
                  005724                437           JSE     COUNT
005724  005731 000001     010          438 LLIST     ZERO    HEAD1,1
005725  005743 000002     010          439           ZERO    HEAD2,2
005726  005767 000003     010          440           ZERO    PCHCD,3
005727  005771 000007     010          441           ZERO    RPCD,7
005730  005772 000005     010          442           ZERO    PGNO,5
                                        443 *
                                        444 *
                                        445 *     TEN WORDS TO BE INSERTED IN PRINT POSITIONS
                                        446 *     OF 31 - 90 OF THE FIRST LINE OF THE HEADING
                                        447 *     WITH CALL PRINT
                                        448 *
                                        449 *
005731  202063303162      000          450 HEAD1     BCI     5,  THIS INFORMATION WILL BE
005732  203145264651      000
005733  442163314645      000
005734  206631433320      000
005735  222520202020      000
005736  202031450225      000          451         BCI     5,  INSERTED IN FIRST LINE
005737  516325242031      000
005740  452026312162      000
005741  632043314525      000
005742  202020206020      000
                                        452 *
                                        453 *
                                        454 *     TWENTY WORDS TO BE USED AS THE SECOND
                                        455 *     HEADING LINE OF EACH PAGE WITH CALL PRINT
                                        456 *
                                        457 *
005743  202063303162      000          458 HEAD2     BCI     9,  THIS INFORMATION WILL
005744  203145264651      000
005745  442163314645      000
```

CALL IOEDIT

```
005746  206631434320    000
005747  202020202020    000
005750  202020204020    000
005751  202020202020    000
005752  202020202020    000
005753  202020204020    000
005754  202022252047    000   459.        BCI     9,  BE PRINTED AS THE
005755  513145634524    000
005756  202162202330    000
005757  252020204020    000
005760  202020204020    000
005761  202020204020    000
005762  202020204020    000
005763  202020202020    000
005764  202020202020    000
005765  202062642220    000   460         BCI     2,  SUB HEADING
005766  302521243145    000
                              461 *
                              462 *
                              463 *     THIS WILL BE PUNCHED IN COLS, 73 - 80
                              464 *     OF EACH CARD WITH THE CALL PUNCH
                              465 *     AND EACH CARD WILL BE SEQUENCED
                              466 *     NUMBERED BEGINNING WITH 00100
                              467 *
                              468 *
005767  272151000001    000   469 PCHCD   BCI     2,GAR00100
005770  000020204020    000
                              470 *
                              471 *
                              472 *     THIS IS USED TO CHANGE THE REPORT
                              473 *     CODE WITH THE CALL PRINT
                              474 *     OCTAL 74 IS ASSUMED BY SYSOUT
                              475 *
                              476 *
005771  000000000024    000   477 RPCD    DEC     20
                              478 *
                              479 *
                              480 *     THE PAGES OF THE OUTPUT REPORT WILL
                              481 *     BE NUMBERED BEGINNING WITH THIS VALUE
                              482 *
                              483 *
005772  000000010000    000   484 PGNO    BCI     1,000100
```

FLOATING LOAD/STORE OPERATIONS

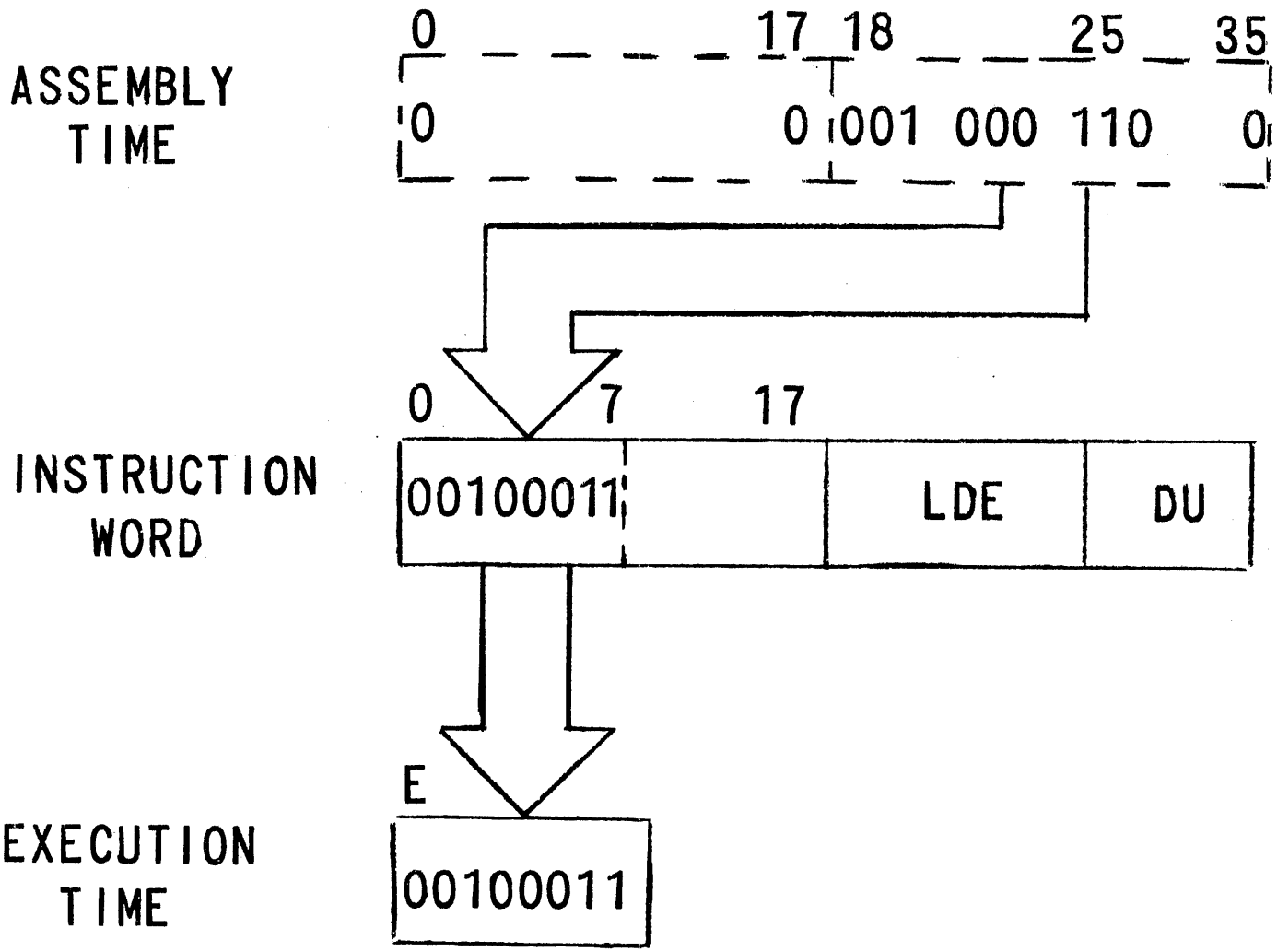


FLOATING LOAD/STORE

FLD	FST
DFLD	DFST
LDE	STE
	FSTR

LOADING EXPONENT REGISTER

EXAMPLE: LDE =35B25,DU



FLOATING POINT ARITHMETIC

FAD	FSB
DFAD	DFSB
ADE	
UFA	UFS
DUFA	DUFS
FNO	FNEG

FLOATING POINT

MULTIPLY AND DIVIDE

FMP	FDV
DFMP	DFDV
UFM	FDI
DUFM	DFDI

FLOATING POINT COMPARE

FCMP	Y
DFCMP	Y-PAIR
FCMG	Y
DFCMG	Y-PAIR

FLOATING POINT

ASSEMBLY NOTATION	FLOATING POINT MEMORY WORD										OCTAL
	S	EXPONENT	S	MANTISSA							
= .1	1	1111101	0	110	011	001	100	110	011	001	772631463146
= .2	1	1111110	0	110	011	001	100	110	011	001	774631463146
= .3	1	1111111	0	100	110	011	001	100	110	011	776463146315
= .4	1	1111111	0	110	011	001	100	110	011	001	776631463146
= .5	0	0000000	0	100	000	000	000	000	000	000	000400000000
= .6	0	0000000	0	100	110	011	001	100	110	011	000463146315
= .7	0	0000000	0	101	100	110	011	001	100	110	000546314632
= .8	0	0000000	0	110	011	001	100	110	011	001	000631463146
= .9	0	0000000	0	111	001	100	110	011	001	100	000714631463
= .10	1	1111101	0	110	011	001	100	110	011	001	772631463146
= 0.	1	0000000	0	000	000	000	000	000	000	000	400000000000
= 1.	0	0000001	0	100	000	000	000	000	000	000	002400000000
= 2.	0	0000010	0	100	000	000	000	000	000	000	004400000000
= 3.	0	0000010	0	110	000	000	000	000	000	000	004600000000
= 4.	0	0000011	0	100	000	000	000	000	000	000	006400000000
= 5.	0	0000011	0	101	000	000	000	000	000	000	006500000000

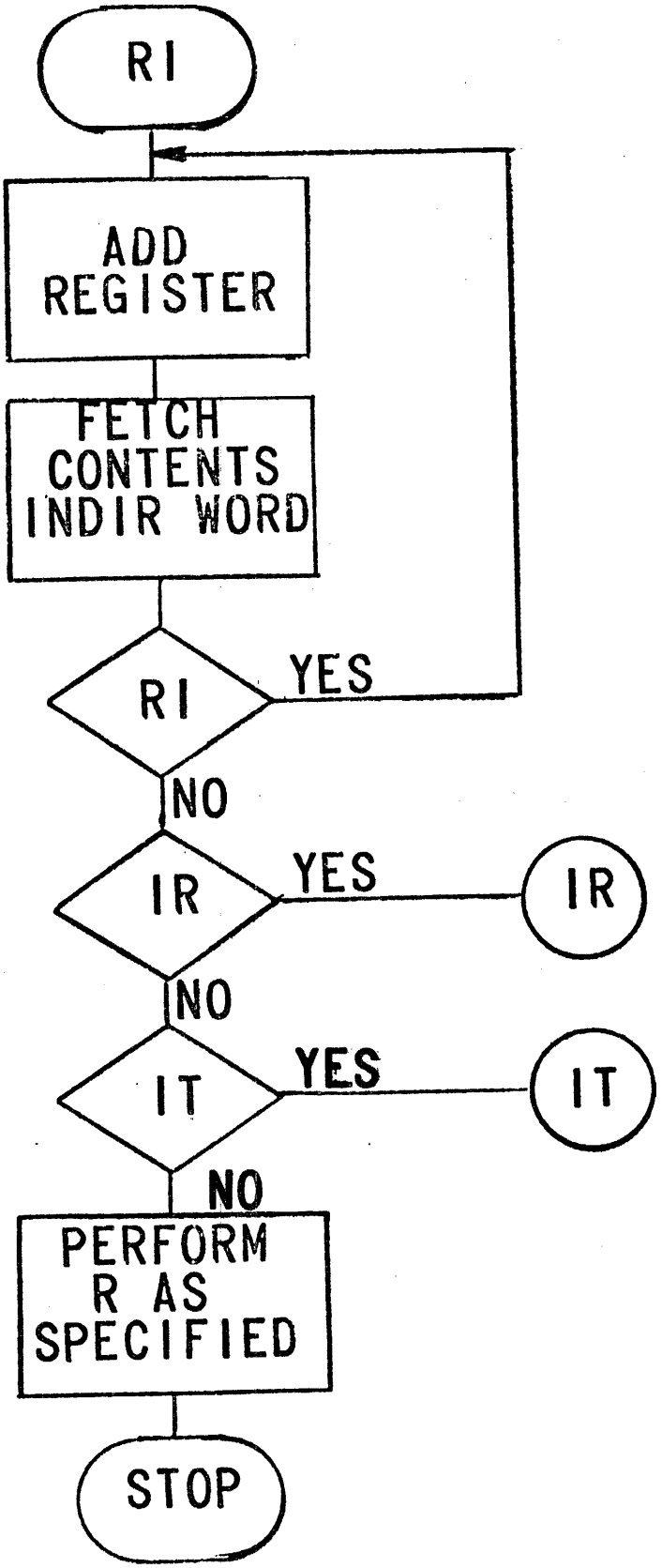
FLOATING POINT

ASSEMBLY NOTATION	FLOATING POINT MEMORY WORD										OCTAL
	S	EXPONENT	S	MANTISSA							
= -1.	0	0 0 0 0 0 0 0	1	000	000	000	000	000	000	000	001000000000
= -2.	0	0 0 0 0 0 0 1	1	000	000	000	000	000	000	000	003000000000
= -3.	0	0 0 0 0 0 1 0	1	010	000	000	000	000	000	000	005200000000
= -4.	0	0 0 0 0 0 1 0	1	000	000	000	000	000	000	000	005000000000
= -5.	0	0 0 0 0 0 1 1	1	011	000	000	000	000	000	000	007300000000
= -.1	1	1 1 1 1 1 0 1	1	001	100	110	011	001	100	110	011	010	773146314632
= -.2	1	1 1 1 1 1 1 0	1	001	100	110	011	001	100	110	011	010	775146314632
= -.3	1	1 1 1 1 1 1 1	1	011	001	100	110	011	001	100	110	011	777314631463
= -.4	1	1 1 1 1 1 1 1	1	001	100	110	011	001	100	110	011	010	777146314632
= -.5	1	1 1 1 1 1 1 1	1	000	000	000	000	000	000	000	000	000	777000000000

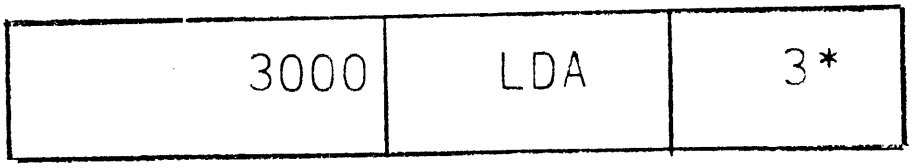
FLOATING POINT

ASSEMBLY NOTATION	FLOATING POINT MEMORY WORD											OCTAL
		EXPONENT		MANTISSA/UNNORMALIZED								
= 1.55E1	0	0 0 0 0 1 1 0	0	001	111	100	000	000	000	000	000	014174000000
				MANTISSA/NORMALIZED								
	0	0 0 0 0 1 0 0	0	111	110	000	000	000	000	000	000	010760000000
= 1.55E+1	1	1 1 1 1 1 1 0	0	100	111	101	011	100	001	010	001	111	774475341217
= 5.12D2	0	0 0 0 1 0 1 0	0	100	000	000	000	000	000	000	000	000	024400000000
	0	0 0 0 0 0 0 0	0	000	000	000	000	000	000	000	000	000	000000000000
= 5.2D-1	0	0 0 0 0 0 0 0	0	100	000	110	001	001	001	101	110	100	000406111564
= 12345.12345	0	0 0 0 1 1 0 0	0	110	000	001	110	010	001	111	110	011	034601621763
= 320.,DU	0	0 0 0 1 0 0 1	0	101	000	000	TRUNCATED @ BIT 17						002500

REGISTER THEN INDIRECT

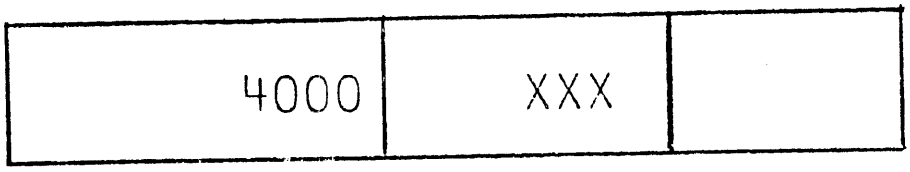


LDA 3000,3* C(X3)=50



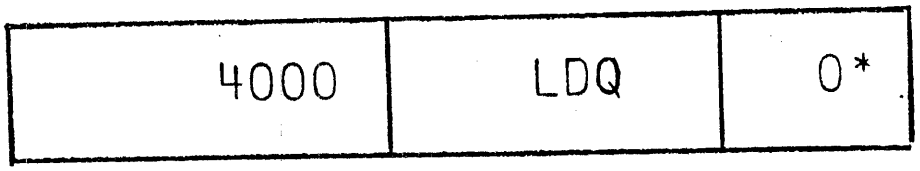
TEMP. EFF. ADDR.= 3000+C(X3)=3050

(3050) XXX 4000

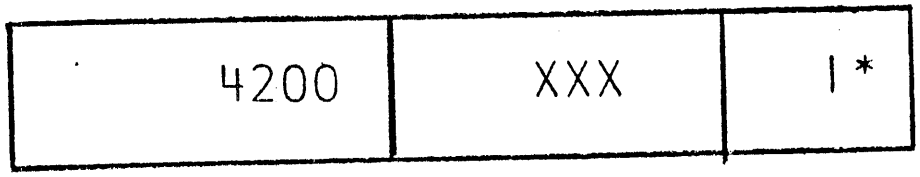


RI MODIFICATION

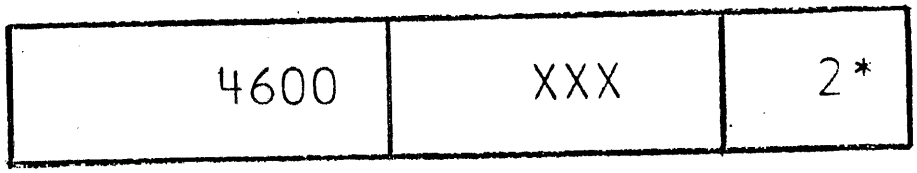
LDQ 4000,0* C(X0)=100



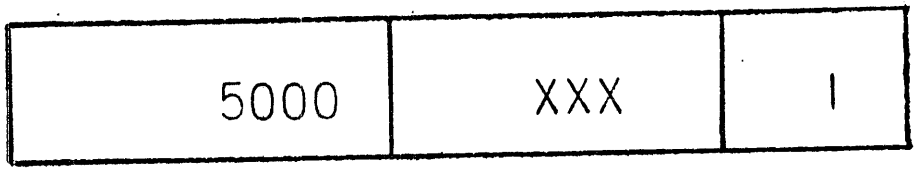
(4100) XXX 4200,1* C(X1)=200



(4400) XXX 4600,2* C(X2)=300



(4900) XXX 5000,1 C(X1)=(STILL)200



RI MODIFICATION

REGISTER INDIRECT (RI)

PERFORM THE REGISTER MODIFICATION TO OBTAIN THE ADDRESS OF THE INDIRECT WORD.

IF THE INDIRECT WORD SPECIFIES FURTHER INDIRECTION, THEN FETCH THE NEXT INDIRECT WORD.

CEASE INDIRECTION WHEN THE CANDIDATE FOR THE ACTUAL EFFECTIVE ADDRESS IS EITHER:

1. NOT INDEXED AT ALL
2. INDEXED BY A REGISTER (R) TYPE
3. INDEXED BY AN INDIRECT THEN TALLY (IT) TYPE

THE VARIATIONS DU AND DL OF REGISTER (R) MODIFICATION CANNOT BE USED WITH (RI) MODIFICATION.

WHEN WORKING WITH (RI) WE USE THE:

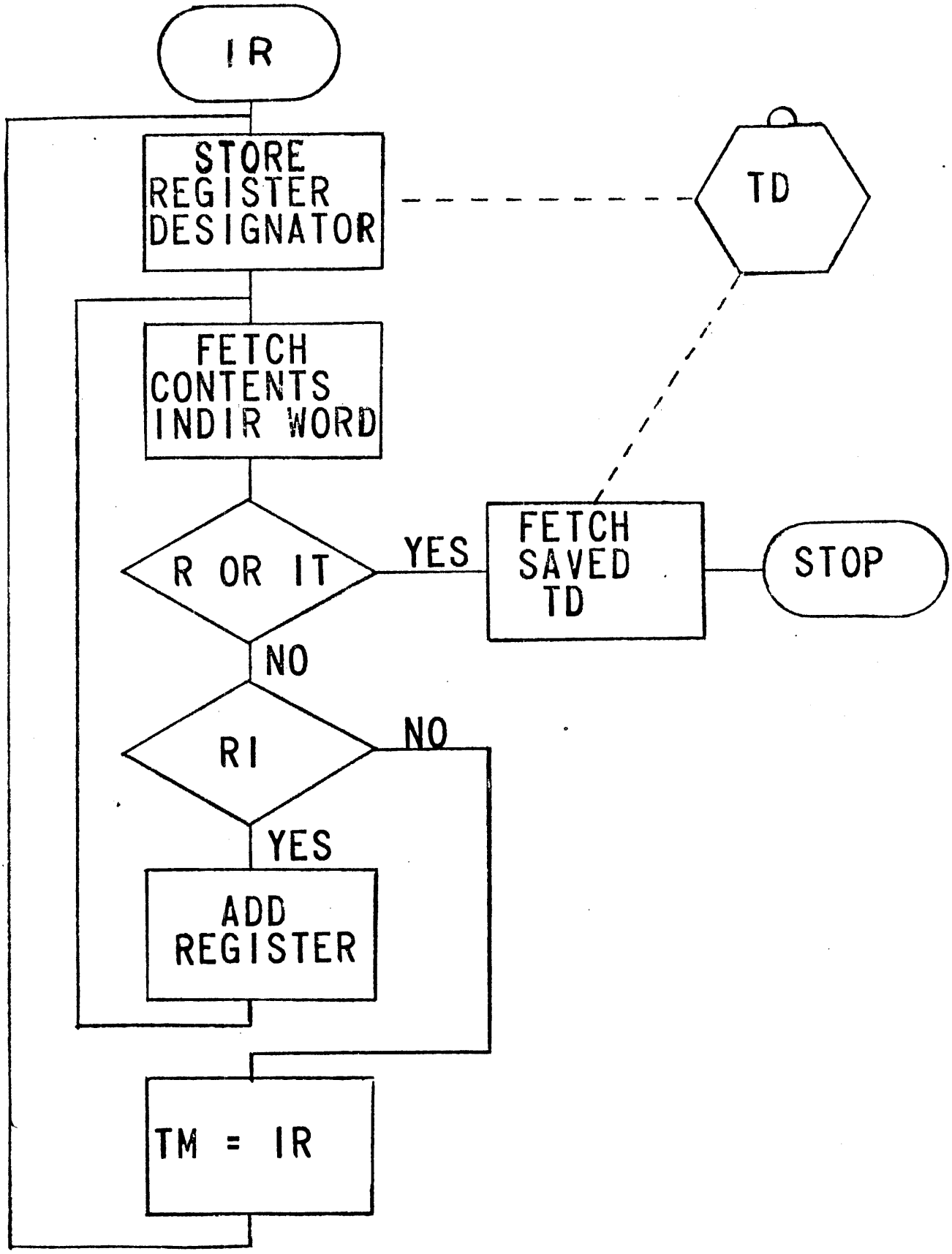
INDEX REGISTERS	X_N ($N = 0 -- 7$)
A-REGISTER	(UPPER AND LOWER PORTIONS)
Q-REGISTER	(UPPER AND LOWER PORTIONS)
INSTRUCTION COUNTER	IC

ALGORITHMS:

REGISTER AND INDIRECT (RI)

PERMISSIBLE REGISTERS		EFFECTIVE ADDRESS	CODING FORMAT
Xn	yi	y + C(Xn) 0-17	Y,0-7*
AU	yi	y + C(A) 0-17	Y,AU*
AL	yi	y + C(A) 18-35	Y,AL*
QU	yi	y + C(Q) 0-17	Y,QU*
QL	yi	y + C(Q) 18-35	Y,QL*
IC	yi	y + C(IC) 0-17	Y,IC*
N	yi	y	Y,N* or Y,*

INDIRECT THEN REGISTER



LDA 3000,*5 X5 SAVED

3000	LDA	*5

(3000) XXX 1000,*7 X5 NOW REPLACED BY X7

1000	XXX	*7

(1000) XXX 200 CEASE INDIRECTION AND MODIFY USING X7

200	XXX	

IR MODIFICATION

INDIRECT REGISTER (IR)

REGISTER SPECIFIED IS PLACED IN A SAVE-STORE AREA.

ADDRESS IS USED TO LOCATE THE INDIRECT WORD.

PROCESS IS CONTINUED UNTIL THE CANDIDATE FOR THE ACTUAL EFFECTIVE ADDRESS IS EITHER:

1. NOT INDEXED AT ALL
2. INDEXED BY A REGISTER (R) TYPE
3. INDEXED BY AN INDIRECT THEN TALLY (IT) TYPE

THE VARIATIONS DU AND DL OF REGISTER (R) MODIFICATION WITH (IR) MODIFICATION.

WHEN THE SEARCH FOR THE ACTUAL EFFECTIVE ADDRESS IS TERMINATED, THE LAST MODIFICATION PERFORMED IS DEPENDENT UPON THE CONTENTS OF THE SAVE-STORE AREA.

WHEN WORKING WITH (IR) WE USE THE:

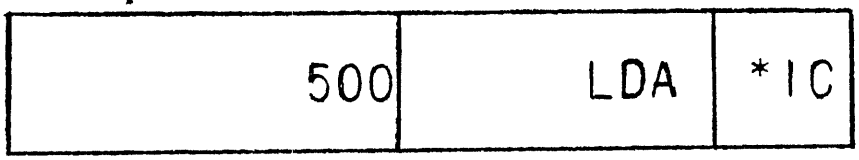
INDEX REGISTER	X_N (N = 0 - 7)
A-REGISTER	(UPPER AND LOWER PORTIONS)
Q-REGISTER	(UPPER AND LOWER PORTIONS)
INSTRUCTION COUNTER	IC

ALGORITHMS:

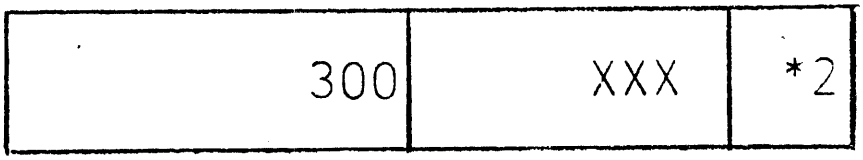
INDIRECT THEN REGISTER (IR)

PERMISSIBLE REGISTERS	EFFECTIVE ADDRESS	CODING FORMAT
Xn	yi = y (SAVE C(Xn)0-17)	Y,*0-7
AU	yi = y (SAVE C(AU)0-17)	Y,*AU
AL	yi = y (SAVE C(AL)18-35)	Y,*AL
QU	yi = y (SAVE C(QU)0-17)	Y,*QU
QL	yi = y (SAVE C(QL)18-35)	Y,*QL
IC	yi = y (SAVE C(IC)0-17)	Y,*IC
N	yi = y (SAVE NO REGISTER)	Y,*N

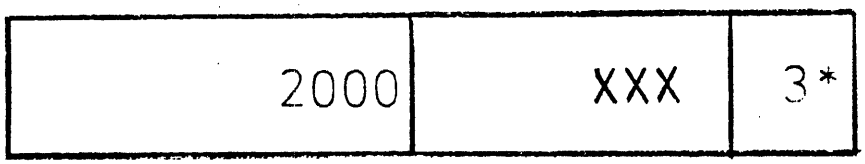
LDA 500,*IC IC SAVED



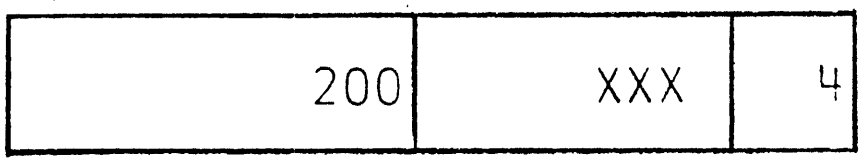
(500) XXX 300,*2 IC IS REPLACED BY X2



(300) XXX 2000,3* C(X3)=1000. USE X3; X2 STILL SAVED.



(3000) XXX 200,4 USE X2; X4 IGNORED.



IR AND RI MODIFICATION

EXAMPLE 1

```
FIRST     LDA     SECOND,AU*
SECOND+C(A)0-17   STA     THIRD,I
```

THIRD+C(XI)$_{0-17}$

EXAMPLE 2

```
NONE                MPY     ONCE,*
ONCE                ARG     TWICE,QU
```

TWICE+C(QU)

EXAMPLE 3

```
                    LDA     A,*QL
A                   ARG     B
```

B+C(Q)$_{18-35}$

EXAMPLE 4

```
                    LDA     B,*3
B                   ARG     C,*5
C                   ARG     D,*QU
D                   ARG     E,7
```

$E+C(Q)_{0-17}$

EXAMPLE 5

```
                    LDA     A,*3
A                   ARG     B,5*
B+C(X5)             ARG     C,IC
```

$C+C(X3)$

EXAMPLE 6

```
                    LDA     Z,*3
Z                   STA     B,DU*
```

NOT VALID

EXAMPLE 7

```
A+C(X3)          LDA     A,3*
B                STA     B,*3
C                LDQ     C,*N
D+C(X1)          STQ     D,1*
E                ARG     E,*
                 ARG     F,3
```

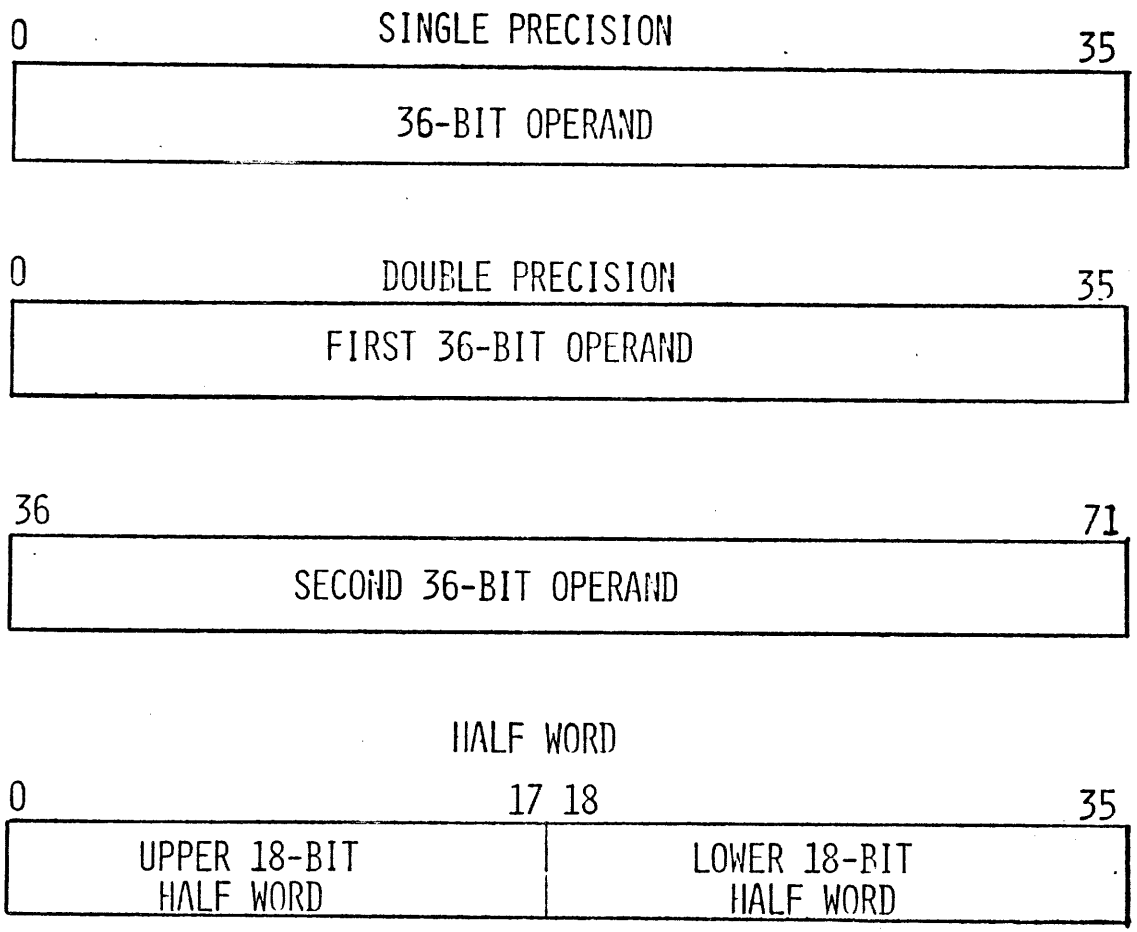
F

EXAMPLE 8

```
START            LDA     A,7*
A+C(X7)          ARG     B,4*
B+C(X4)          ARG     C,5*
C+C(X5)          ARG     D,*

1   D            ARG     E
2   D            ARG     E,3
3   D            ARG     E,ID

    E            TALLY   F,100
```

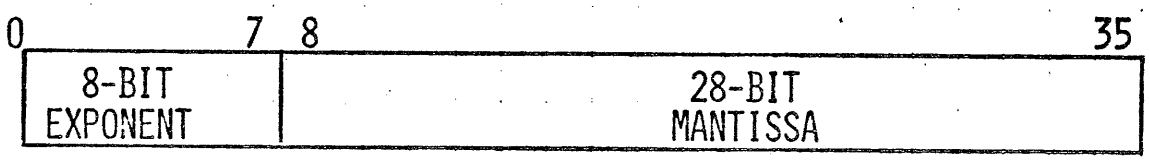
```
1   E
2   E + C(X3)
3   F
```

FIXED-POINT DATA FORMATS

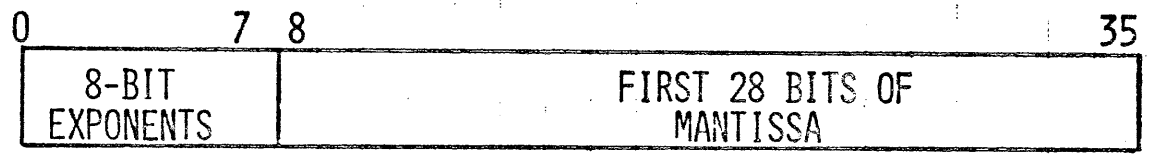


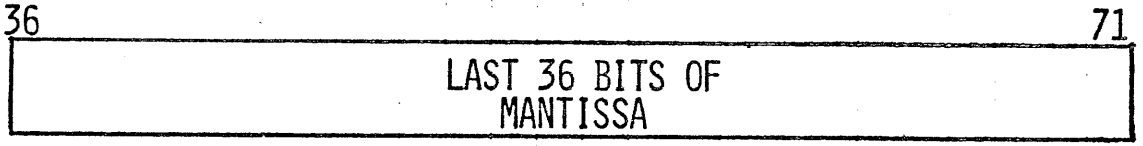
FLOATING-POINT DATA FORMATS

SINGLE PRECISION



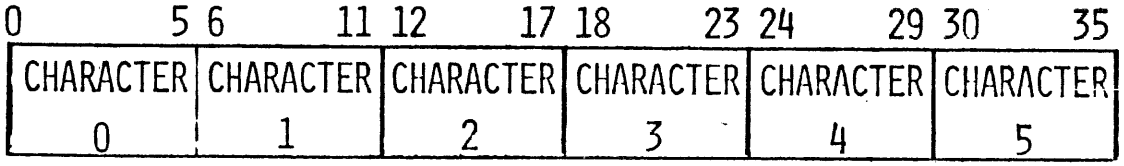
DOUBLE PRECISION



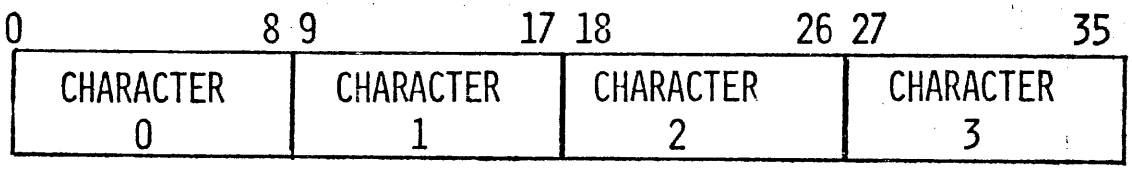


ALPHANUMERIC DATA FORMATS

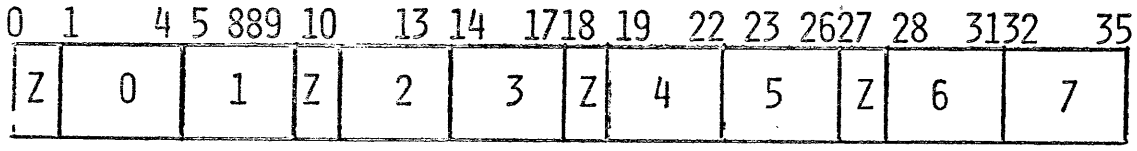
6-BIT FORMAT



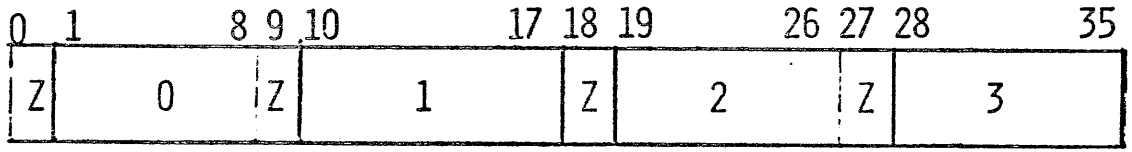
9-BIT FORMAT



DECIMAL FORMATS FOR EIS

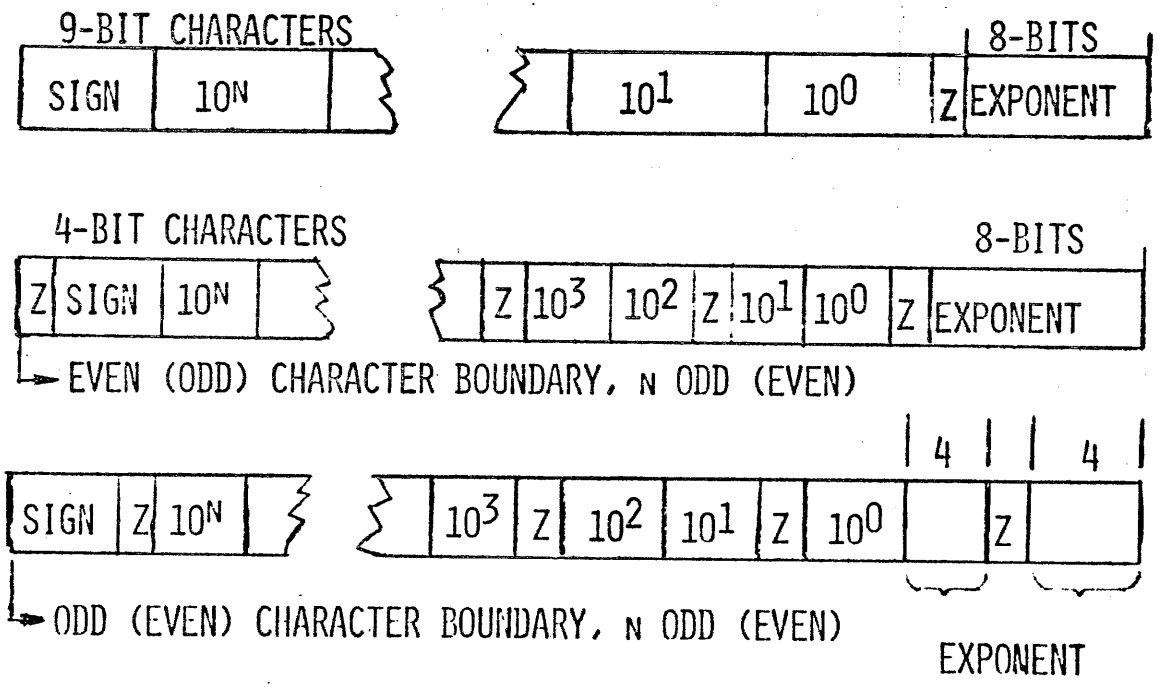


PACKED DECIMAL (4-BIT)

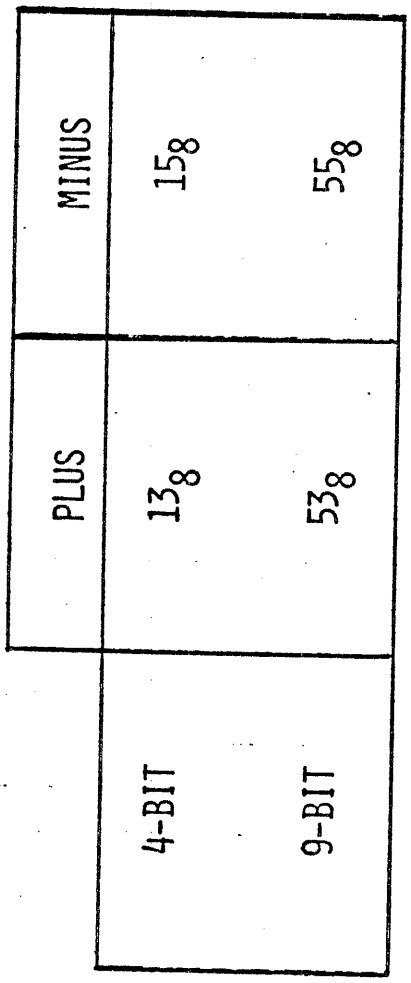


ASCII (9-BIT)

FLOATING-POINT DECIMAL FORMATS



DECIMAL SIGN POSITION FORMATS



	000(1)[1]	040(1)	100(1)	140(1)	200(1)	240(1)	300(1)	340(1)
000			MLR				MVN	
001			MRL				BTD	
002					AD2D			
003					SB2D		CMPN	
004								
005							DTB	
006			CMPC		MP2D			
007					DV2D			
010								
011								
012								
013								
014								
015								
015								
017								
020	MVE	CSL	SCD	MVT				
021		CSR	SCDR					
022					AD3D			
023					SB3D			
024	MVNE	SZTL	SCM	TCT				
025		SZTR	SCMR	TCTR				
026		CMPB			MP3D			
027					DV3D			
030								
031								
032								
033								
034								
035								
036								
037								

[1] Octal representation of the operation code consists of three octal digits corresponding to instruction bit positions 18-26 and a zero (0) or one (1) for instruction bit position 27 (the EIS bit).

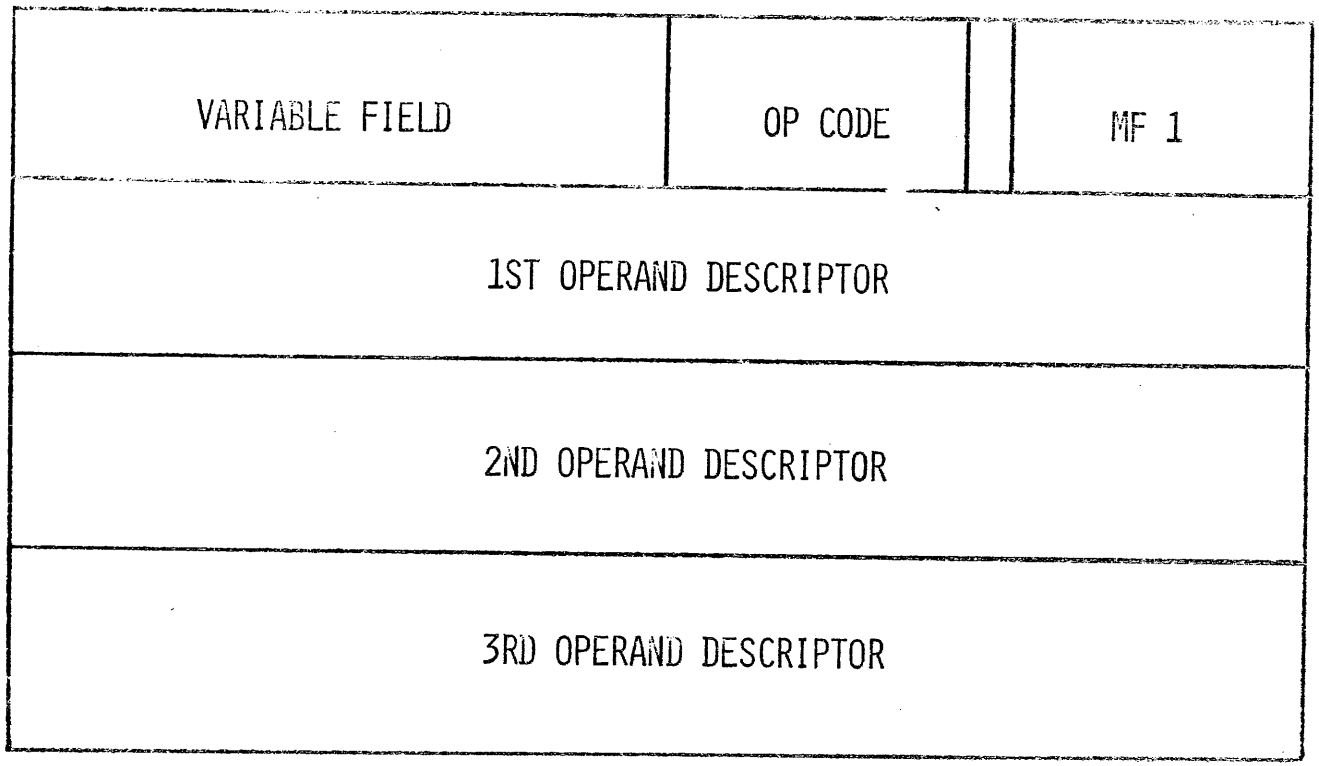
Figure 1-2. Operation Code Map for the Extended Instructions

	400(1)[1]	440(1)	500(1)	540(1)	600(1)	640(1)	700(1)	740(1)
000			A9BD	ARA0	TRTN	ARN0		SAR0
001			A6BD	ARA1	TRTF	ARN1		SAR1
002			A4BD	ARA2		ARN2		SAR2
003		SAREG	ABD	ARA3		ARN3		SAR3
004				ARA4	TMOZ	ARN4		SAR4
005				ARA5	TPNZ	ARN5		SAR5
006				ARA6	TTN	ARN6		SAR6
007		SPL	AWD	ARA7		ARN7		SAR7
010								
011								
012								
013								
014								
015								
016								
017								
020			S9BD	AAR0		NAR0		LAR0
021			S6BD	AAR1		NAR1		LAR1
022			S4BD	AAR2		NAR2		LAR2
023		LAREG	SBD	AAR3		NAR3		LAR3
024				AAR4		NAR4		LAR4
025				AAR5		NAR5		LAR5
026				AAR6		NAR6		LAR6
027		LPL	SWD	AAR7		NAR7		LAR7
030								
031								
032								
034								
035								
036								
037								

[1] Octal representation of the operation code consists of three octal digits corresponding to instruction bit positions 18-26 and a zero(0) or one (1) for instruction bit position 27 (the EIS bit).

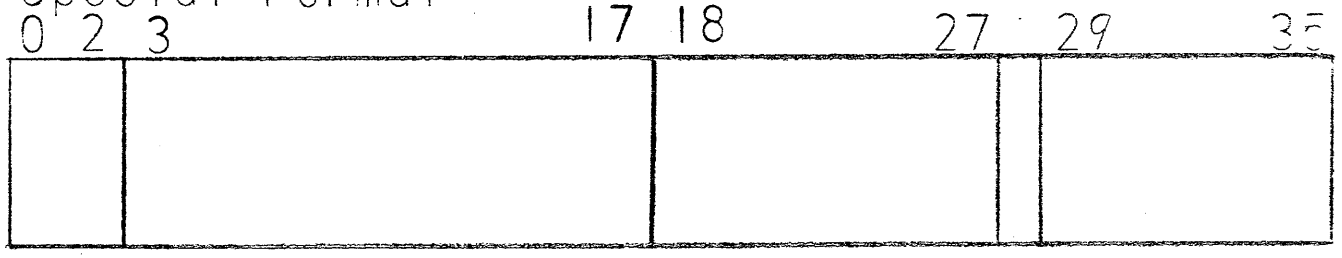
Figure 1-2 (cont). Operation Code Map for the Extended Instructions

EIS MULTI-WORD INSTRUCTIONS

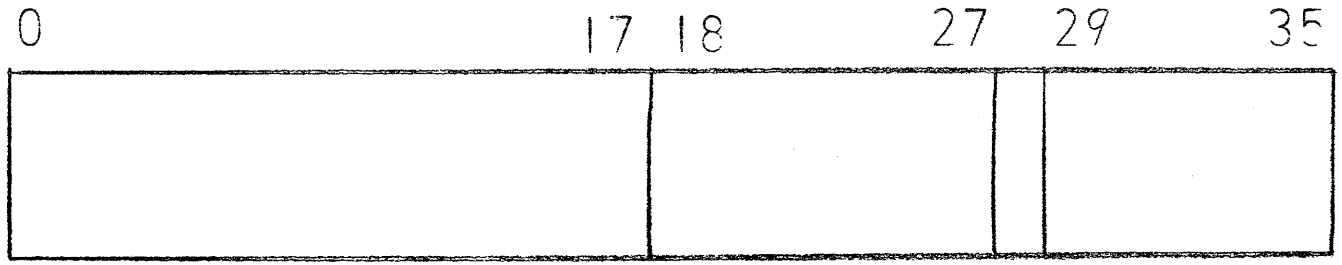


SINGLE WORD INSTRUCTIONS

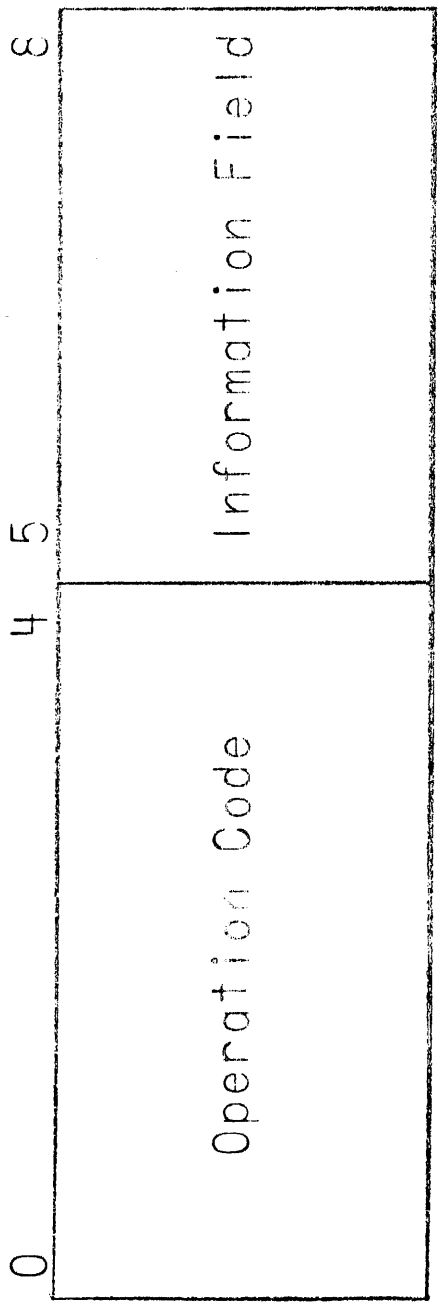
Special Format



Normal Format

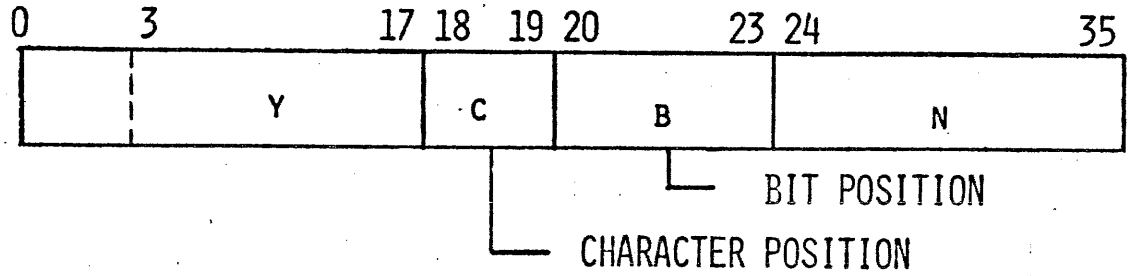


MICRO OPERATIONS

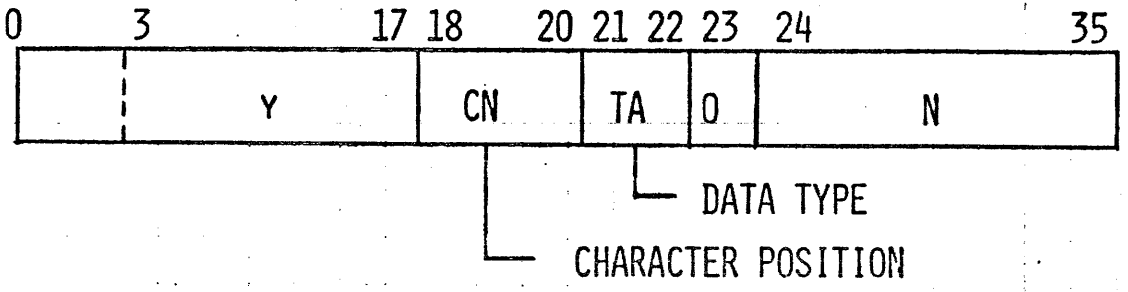


OPERAND DESCRIPTORS

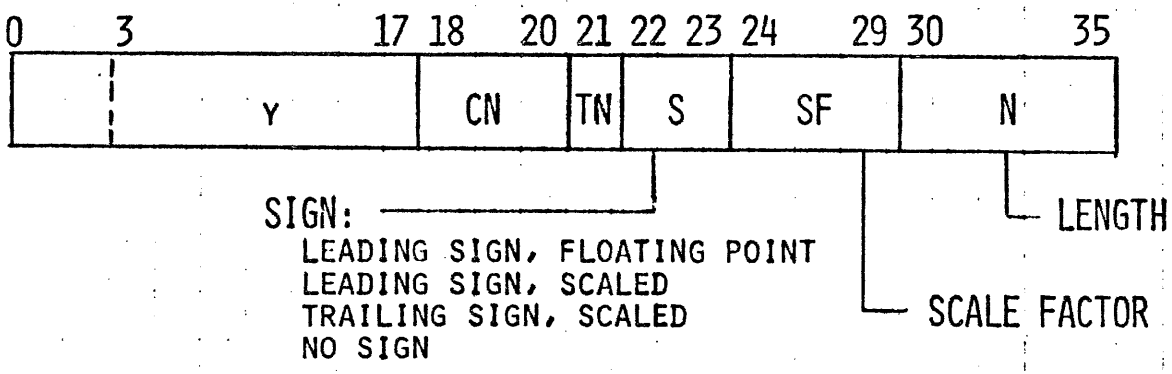
BIT STRING



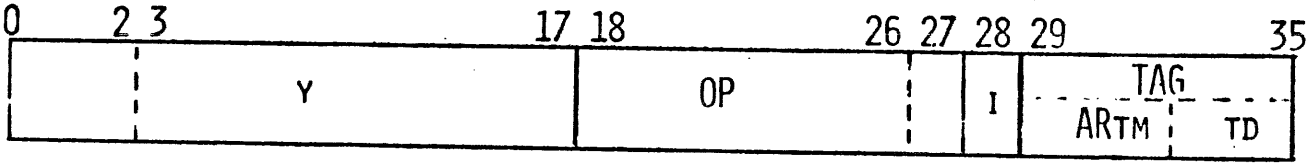
ALPHANUMERIC

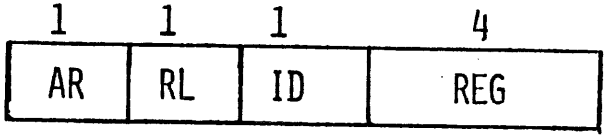


NUMERIC



EIS ADDRESS MODIFICATION





NUMBER OF BITS

SYMBOL

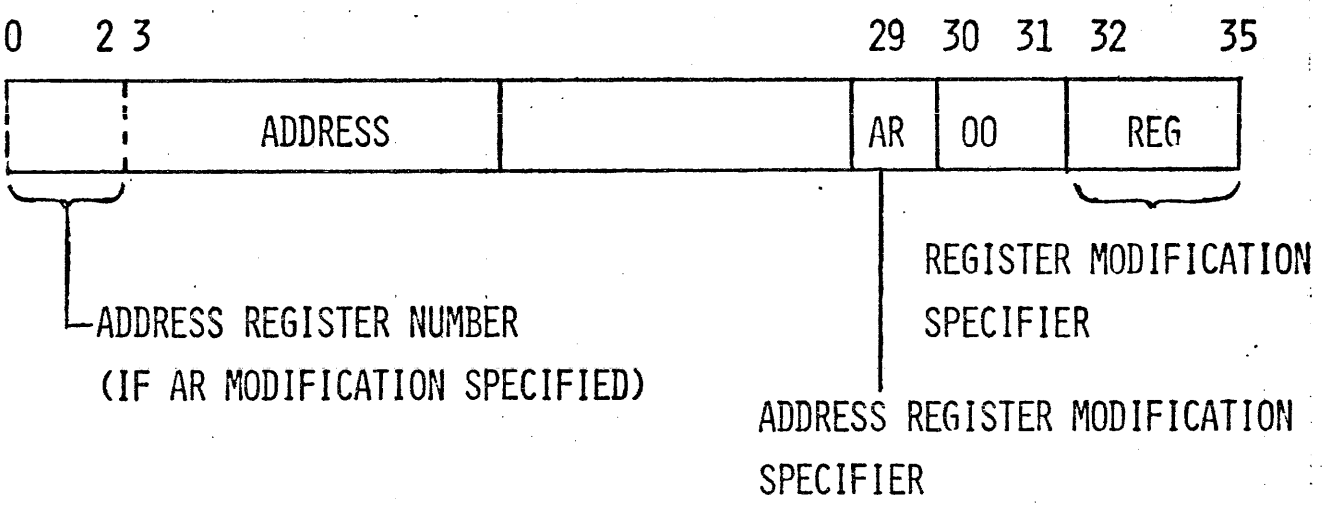
AR ADDRESS REGISTER SPECIFIER

RL REGISTER OR LENGTH

ID INDIRECT OPERAND DESCRIPTOR

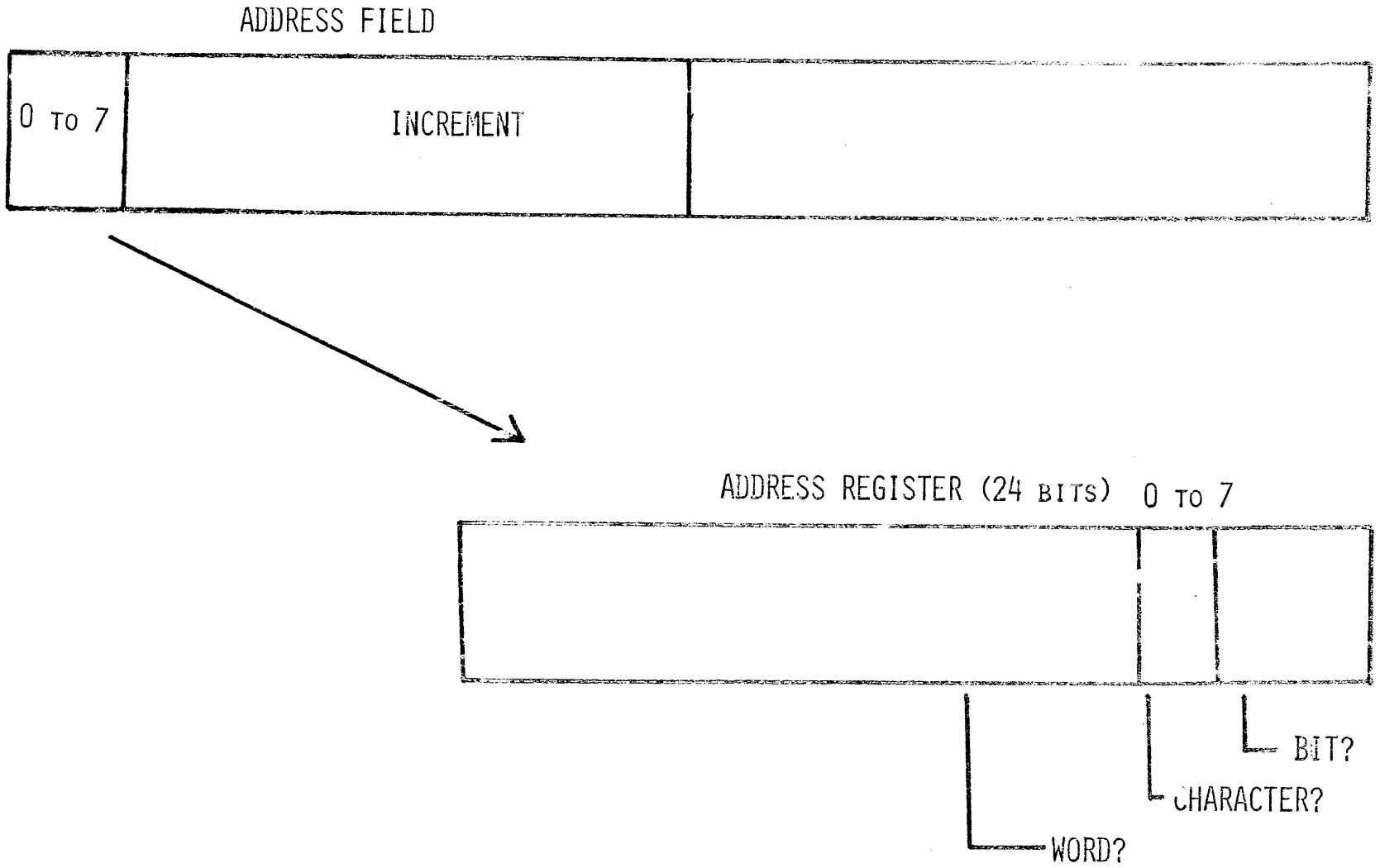
REG ADDRESS MODIFICATION REGISTER

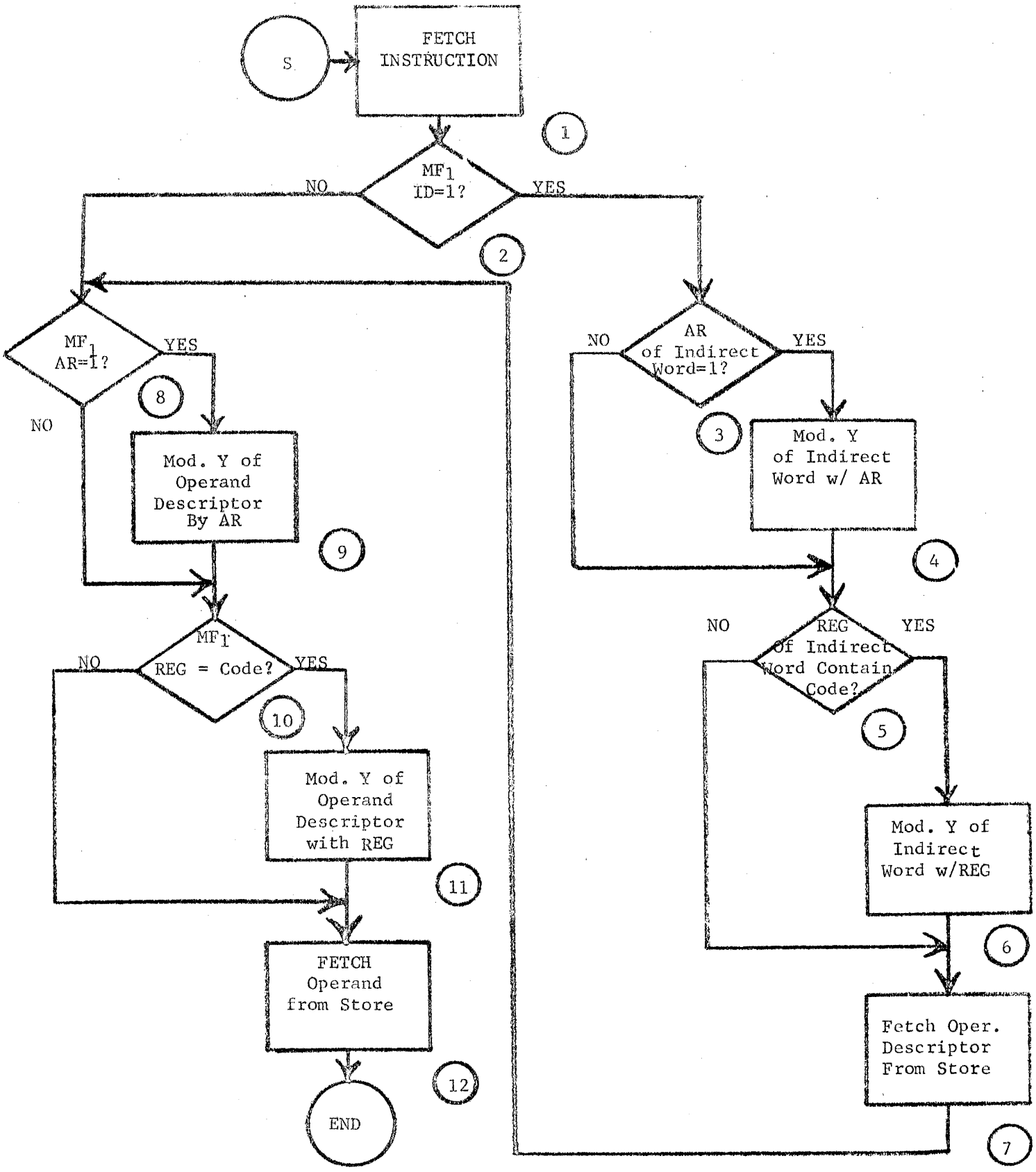
INDIRECT WORD



AR AND REG FIELDS IDENTICAL IN FUNCTION WITH CORRESPONDING MODIFICATION FIELDS IN INSTRUCTION.

ADDRESS REGISTERS




S
FETCH INSTRUCTION
1
MF1 ID=1?
NO
YES
2
MF1 AR=1?
YES
NO
8
Mod. Y of Operand Descriptor By AR
9
AR of Indirect Word=1?
NO
YES
3
Mod. Y of Indirect Word w/ AR
4
MF1 REG = Code?
NO
YES
10
Mod. Y of Operand Descriptor with REG
11
REG of Indirect Word Contain Code?
NO
YES
5
Mod. Y of Indirect Word w/REG
6
FETCH Operand from Store
12
END
Fetch Oper. Descriptor From Store
7

```
$$    46264 ENTERED FIN635 AT 12,971 FROM CD RDR   0-06-00

0001  $       SNUMB   46264
0002  $       IDENT   4817,=TRNG-RGK,J959,C959100,8177
0003  $       GMAP    NDECK
0004  $       LPDATE                                          GINA1
0006  $       GMAP    NDECK
0007  $       LPDATE                                          GINA2
0020  $       GMAP    NDECK
0021  $       LPDATE                                          GINA3
0026  $       EXECUTE DUMP
0027  $       LIMITS  ,5K,,1000
0028  $       SYSOUT  A1
0029  $       SYSOUT  A2
0030  $       DATA    A3
0031  $       ENDJOB
  TOTAL CARD COUNT THIS JOB = 000115


 * BEGIN ACTIVITY =01-  GMAP     10/12/71   SW=211400000000
  * NORMAL TERMINATION      AT 002420  INDICATORS  5020

  * NORMAL TERMINATION      AT 002420  INDICATORS  5020

  * NORMAL TERMINATION      AT 002420  INDICATORS  4020

     START 12,974      LINES    755      PROC  0,0011      I/O   0,003    IU  5   MEMORY   24K
     STOP  12,982      LIMIT  12446      LIMIT 0,0992      LIMIT          CU  5   M=T      807

     LAPSE  0,009   FC D TYPE     BUSY    IP/AT    FP/RT    IS/#C MS/#E   ADDRESS L#/T#

                    G* R D167 *    248        0        3        1     6   0-00=08  255
                    A* R D167 *    117        0        1        1     1   0-00=08  255
                    P*  SYOUT
                    K*  SYOUT
                    C*  SYOUT
                    *1 R D167     1967        0        0        4     4   0-00=02  633
                    B* S D167      207        0        4        2     2   0-00=03  123

     LIST   755  LINES


* BEGIN ACTIVITY =02-  GELOAD   10/12/71   SW=400000000000
 * USERS    ME GEBORT     AT 011717  INDICATORS  3020

     START 12,983      LINES    550      PROC  0,0003      I/O   0,004    IU  5   MEMORY    5K
     STOP  12,990      LIMIT   1000      LIMIT 0,0500      LIMIT          CU  5   M=T      27A

     LAPSE  0,007   FC D TYPE     BUSY    IP/AT    FP/RT    IS/#C MS/#E   ADDRESS L#/T#

                    B* R D167      177        4        4        2     2   0-00=03  123
                    A3 R D167 *     50        0        1        1     1   0-00=08  256
                    B* R D167 *     56        0        0        1     1   0-00=08  256
                    A1  SYOUT
                    A2  SYOUT
                    P*  SYOUT
                    L* R D167     2963        0        0       28    28R  0-00=01  385

     LIST   540  LINES
     PUNCH   10  CARDS
```

PREFACE

PROGRAM BREAK 36
COMMON LENGTH 0
V CONST BITS 5

PRIMARY SYMDEF ENTRY

START 0

SECONDARY SYMDEF ENTRY

BLOCK LENGTH

SYMREF

1 CALC
2 EDIT
3 INIT
4 FLOAT
5 INPUT
6 TOBCD
7 TOBIN
10 OUTPUT
11 WRAPUP

```
46264 01  10-12-71   12,974      SUBPROGRAM NO. 1 - DRIVER                    PAGE    2

                                        1          TTL     SUBPROGRAM NO. 1 - DRIVER
                                        2          SYMDEF  START
 000000  030000701000     030       3 START    CALL    INIT
 000001  000003710000     010
 000002  000034000003     010
 000003  050000701000     030       4 READ     CALL    INPUT
 000004  000006710000     010
 000005  000034000004     010
 000006  070000701000     030       5 SKIP     CALL    TOBIN
 000007  000011710000     010
 000010  000034000005     010
 000011  040000701000     030       6          CALL    FLOAT
 000012  000014710000     010
 000013  000034000006     010
 000014  010000701000     030       7          CALL    CALC
 000015  000017710000     010
 000016  000034000007     010
 000017  060000701000     030       8          CALL    TOBCD
 000020  000022710000     010
 000021  000034000010     010
 000022  020000701000     030       9          CALL    EDIT
 000023  000025710000     010
 000024  000034000011     010
 000025  100000701000     030      10          CALL    OUTPUT
 000026  000030710000     010
 000027  000034000012     010
 000030  000003 7100 00   010      11          TRA     READ
 000031  110000701000     030      12          CALL    WRAPUP
 000032  000034710000     010
 000033  000034000014     010

ERROR LINKAGE

 000034  000000000000     000
 000035  626321516320     000
                                       13          END
  36 IS THE NEXT AVAILABLE LOCATION,   GMAP VERSION  JMPA/030271 JMPE/030271 JMPC/030271
THERE WERE    NO  WARNING FLAGS IN THE ABOVE ASSEMBLY
```

```
46264 01  10-12-71   12,975      SUBPROGRAM NO, 1 = DRIVER                    PAGE      3

OCTAL    SYMBOL    REFERENCES BY ALTER NO,

     1   CALC              7
     2   EDIT              9
     4   FLOAT             6
     3   INIT              3
     5   INPUT             4
    34   .E.L..            3     4     5     6     7     8     9    10    12
    10   OUTPUT           10
     3   READ        4     4    11
     0   START       3     2     3
     6   TCBCD             8
     7   TCBIN             5
    11   WRAPUP           12
** 18189 WORDS OF MEMORY WERE USED BY GMAP FOR THIS ASSEMBLY,
```

INITIALIZATION SUBROUTINE

PREFACE

PROGRAM BREAK	339
COMMON LENGTH	1703
V COUNT BITS	5

PRIMARY SYMDEF ENTRY

INIT	0
INPUT	166
OUTPUT	201
ENDFLP	300

SECONDARY SYMDEF ENTRY

	BLOCK	LENGTH
1	IO.WRK	64

SYMREF

2 GET
3 OPEN
4 CLOSE
5 PUNCH

INITIALIZATION SUBROUTINE

```
                                                    1          TTL     SUBPROGRAM NO. 2 - INPUT/OUTPUT SUBROUTINES
                                                    2          TTLS    INITIALIZATION SUBROUTINE
                                  000000            4 INIT     SAVE
000000  0000027100C0      010
000001  000315630000      010
000002  000315754000      010
000003  000315741000      010
000004  030000701000      030     5          CALL    OPEN(FUNOT,1)
000005  000011710000      010
000006  000315000005      010
000007  000163000000      010
000010  000301000000      000
000011  030000701000      030     6          CALL    OPEN(FRNOT,1)
000012  000316710000      010
000013  000315000006      010
000014  000164000000      010
000015  000301000000      000
000016  030000701000      030     7          CALL    OPEN(READIN,1)
000017  000023710000      010
000020  000315000007      010
000021  000165000000      010
000022  000001000000      000
000023  000016 2260 03    000     8          LDX6    14,DU
000024  000157 7460 00    010     9          STX6    CRDFIL+1
000025  000125 7460 00    010    10          STX6    PLNFIL+1
000026  000026 2260 03    000    11          LDX6    22,DU
000027  000142 7460 00    010    12          STX6    PRNFIL+1
000030  000021 0010 00    000    13          MME     GETIME
000031  000262 7550 00    010    14          STA     TIME1
000032  000006 7320 00    000    15          QRS     6
000033  000320 5060 00    010    16          DIV     =60000C         CONVERT TO MINUTES
000034  000321 5000 00    010    17          DIV     =60              CONVERT TO HOURS
000035  000217 7550 00    010    18          STA     MINS
000036  000014 1160 07    000    19          CMPQ    =12,DL
000037  000042 6000 00    010    20          TZE     P7
000040  000045 6040 00    010    21          TMI     A7
000041  000014 1760 07    000    22          SBQ     =12,DL
000042  000322 2350 00    010    23 P7       LDA     =H P.M.
000043  000275 7550 00    010    24          STA     TIME2+1
000044  000047 7100 00    010    25          TRA     *+3
000045  000323 2350 00    010    26 A7       LDA     =H A.M.
000046  000275 7550 00    010    27          STA     TIME2+1
000047  000220 7560 00    010    28          STQ     HCURS
000050  000000 6360 00    000    29          EAQ     0
000051  000000 2260 03    000    30          LDX6    ,DU
000052  000220 2350 00    010    31          LDA     HCURS
000053  014200 5202 01    000    32          RPT     6,1
000054  000222 5050 16    010    33          BCD     TAB,6
000055  000011 1160 07    000    34          CMPQ    =3H000,DL
000056  000060 6030 00    010    35          TRC     *+2
```

INITIALIZATION SUBROUTINE

```
                                                  56          ORG     =C202C00
                                                  57          ORG     =C200C00
000062 000221 7560 00 010                         58          STC     HRSBCD
000063 000000 2260 03 000                         59          LDX6    ,DU
000064 000217 2350 00 010                         60          LDA     MINS
000065 014200 5202 01 000                         61          RPT     6,1
000066 000222 5050 16 010                         62          BCT     TAB,6
000067 000215 7560 00 010                         63          STC     CKMINS
000070 000326 2760 00 010                         64          ORG     =C330C00
000071 000002 7360 00 000                         65          QLS     18
000072 000221 2350 00 010                         66          LDA     HRSBCD
000073 000022 7330 00 000                         67          LRS     18
000074 000274 7560 00 010                         68          STC     TIME2
000075 000262 2350 00 010                         69          LDA     TIME1
000076 000216 7550 00 010                         70          STA     CKTIME
000077 000014 7330 00 000                         71 SHIFTQ   LRS     12
000100 000006 7320 00 000                         72          QRS     6
000101 000327 2760 00 010                         73          ORG     =C610C00000000
000102 000014 7330 00 000                         74          LRS     12
000103 000006 7320 00 000                         75          QRS     6
000104 000327 2760 00 010                         76          ORG     =C610C00000000
000105 000330 1150 00 010                         77          CMPA    =C100
000106 000111 6040 00 010                         78          TMI     ALESS
000107 000331 2750 00 010                         79          ORA     =C202020200000
000110 000262 7570 00 010                         80          STAQ    TIME1
000111 000113 7100 00 010                         81          TRA     *+3
000112 000332 2750 00 010                         82 ALESS    ORA     =C202C20202000
000113 000107 7100 00 010                         83          TRA     *-3
000114 000001710000 000114 010                    84          RETURN  INIT
                                                  85          FILCB   PUNFIL,A1,PUNBUF
000114 000000000000 000                           86          VFD     R18/,18/0
000115 000000000000 000                           87          DEC     0
000116 000000000000 000                           88          VFD     18/0,1/0,3/,2/0,1/,11/0
000117 000000000400 000                           89          VFD     18/,5/0,1/,1/,1/,2/1,8/0
000120 000000002101 000                           90          VFD     24/0,+12/A1
000121 000000000000 000                           91          DEC     0,0,0
000122 000000000000 000
000123 000000000000 000
000124 000000000000 000                           92 PUNFIL   VFD     18/0,2/,2/,1/,13/0
000125 000000 000000 000                          93          ZERO    ,0
000126 000000 000000 022                          94          ZERO    PUNBUF,PUNBUF
000127 000000 000000 000                          95          ZERO    0
000130 000500 000000 000                          96          ZERO    320,0
000131 000131 000                                 97          FILCB   PRNFIL,A2,PRNBUF
000131 000000000000 000                           98          VFD     R18/,18/0
000132 000000000000 000                           99          DEC     0
000133 000000000000 000                          100          VFD     18/0,1/0,3/,2/0,1/,11/0
000134 000000000400 000                          101          VFD     18/,5/0,1/,1/,1/,2/1,8/0
000135 000000002102 000                          102          VFD     24/0,+12/A2
000136 000000000000 000                          103          DEC     0,0,0
```

INITIALIZATION SUBROUTINE

```
000137  000000000000  000
000140  000000000000  000
000141  000000000000  000
000142  010000 000000  000
000143  000501 000501  022
000144  000000 000000  000
000145  000500 000000  000
                000146
000146  000000000000  000
000147  000000000000  000
000150  000000000000  000
000151  000000000400  000
000152  000000002103  000
000153  000000000000  000
000154  000000000000  000
000155  000000000000  000
000156  000000000000  000
000157  000000 000000  000
000160  001202 001202  022
000161  000000 000000  000
000162  000500 000000  000
000163  000124000000  010
000164  000141400000  010
000165  000156200000  010
                000000
                000000
                000501
                001202
                000166
```

```
                PRNFIL  VFD     18/0,2/,2/,1/,13/0
                        ZERO    ,0
                        ZERO    PRNBUF,PRNBUF
                        ZERO    0
                        ZERO    320,0
     67                 FILCB   CRDFIL,A3,RDBLF1
                        VFD     R18/,18/0
                        DEC     0
                        VFD     18/0,1/0,3/,2/0,1/,11/0
                        VFD     18/,5/0,1/,1/,1/,2/1,8/0
                        VFD     24/0,+12/A3
                        DEC     0,0,0

                CRDFIL  VFD     18/0,2/,2/,1/,13/0
                        ZERO    ,0
                        ZERO    RDBLF1,RDBLF1
                        ZERO    0
                        ZERO    320,0
     68 PUNOT   VFD     18/PUNFIL,1/1,17/0
     69 PRNOT   VFD     18/PRNFIL,1/1,17/0
     70 READIN  VFD     18/CRDFIL,2/1
     71         BLOCK
     72 PUNBUF  BSS     321
     73 PRNBUF  BSS     321
     74 RDBUF1  BSS     321
     75         USE
```

BETTER TRY FOR INPUT

```
                    000166      77 INPUT  SAVE
000166  000170710003     010
000167  000315630000     010
000170  000315754000     010
000171  000315741000     010
000172  000203701000     030    78         CALL GET(CRDFIL,ENDCWE,RCWRK)
000173  000203710000     010
000174  000315000116     010
000175  000136000000     010
000176  000340000000     010
000177  000000000000     030
000200  000167710000     010    79         RETURN  INPUT
                    000201      80 OUTPUT SAVE
000201  000203710000     010
000202  000315630000     010
000203  000315754000     010
000204  000315741000     010
000205  000203701000     030    81         CALL PUNCH(PUNFIL,PUNCLT,=1)
000206  000213710000     010
000207  000315000121     010
000210  000124000000     010
000211  000144000000     030
000212  000333000000     010
000213  000202710000     010    82         RETURN  OUTPUT
000214  000000000000     000    83 CKHRS   DEC     0
000215  000000000000     000    84 CKMINS  DEC     0
000216  000000000000     000    85 CKTIME  DEC     0
                    000217      86 MINS    BSS     1
                    000220      87 HOURS   BSS     1
                    000221      88 HRSAV   BSS     1
000222  000003032400     000    89 [illegible]     ... 409500,327680,262144
000223  000002342000     000
000224  000004750000     000
000225  000001440000     000
000226  000001200000     000
000227  000001000000     000
000230  000000000030     000    90 LNCNT   DEC     24
000231  772020202020     000    91 SLEW1   OCT     772020202020
000232  202020202020     000    92 HDR1    BCI     9,
000233  202020202020     000
000234  202020202020     000
000235  202020202020     000
000236  202020202020     000
000237  202020202020     000
000240  202020202020     000
000241  202020202020     000
000242  202020202020     000
000243  304645257066     000    93         BCI     7,HONEYWELL MARKETING EDUCATION GMAP COURSE
000244  254343204421     000
000245  514225633145     000
```

BETTER TRY FOR INPUT

```
000246  272025246423  000
000247  216331464520  000
000250  274421472023  000
000251  466451622520  000
000252  770200000000  000    94          OCT     770200000000
000253  000000011007  000
000254  202020202020  000    95 HDR2     BCI     6,
000255  202020202020  000
000256  202020202020  000
000257  202020202020  000
000260  202020202020  000
000261  202020202020  000
000262  202020202020  000    96 TIME1    BCI     1,
000263  202020202020  000    97          BCI     9,
000264  202020202020  000
000265  202020202020  000
000266  202020202020  000
000267  202020202020  000
000270  202020202020  000
000271  202020202020  000
000272  202020202020  000
000273  202020202020  000
000274  202020202020  000    98 TIME2    BCI     1,
000275  204673234346  000    99          BCI     2, C:CLOCK
000276  234220202020  000
000277  770400000000  000   100          OCT     770400000000
                000300      101 WRAPUP  SAVE
000300  000302710000  010
000301  000315630000  010
000302  000315754000  010
000303  000315741000  010
000304  000305 7100 00  010  102 ENDCRD  TRA     *+1
000305  040000701000  030    103          CALL CLOSE(PUNOT,3)
000306  000312710000  010
000307  000315000147  010
000310  000163000000  010
000311  000003000000  000
000312  000334 2360 00  010  104          LDQ     =3HRGK,DL
000313  000010 0010 00  000  105          MME     GEBCRT
000314  000301710000  010    106          RETURN  WRAPUP
                000000      107          BLOCK   IC,WRK
000000  202020202020  000    108 RDWRK   EBCI    5,      8   2    2          1
000001  201020202002  000
000002  202020202002  000
000003  202020202020  000
000004  202020202001  000
                000005      109          BSS     9
000016  202020202020  000    110 PRTOUT  OCT     202020202020
000017  202020000000  000    111 PRSEQ   OCT     202020000000
000020  202020202020  000    112          OCT     202020202020
```

BETTER TRY FOR INPUT

```
000021  202020202020  000    113 PRE     BCI     3,
000022  202020202020  000
000023  202020202020  000
000024  202020202020  000    114 PRF     BCI     3,
000025  202020202020  000
000026  202020202020  000
000027  202020202020  000    115 PRVAR   BCI     6,
000030  202020202020  000
000031  202020202020  000
000032  202020202020  000
000033  202020202020  000
000034  202020202020  000
000035  202020202020  000    116         BCI     6,
000036  202020202020  000
000037  202020202020  000
000040  202020202020  000
000041  202020202020  000
000042  202020202020  000
000043  770200000000  000    117         OCT     770200000000
                      000044 118 PUNOUT  BSS     0
000044  202020202020  000    119 PUE     BCI     2,
000045  202020202020  000
000046  202020202020  000    120 PUF     BCI     2,
000047  202020202020  000
000050  202020202020  000    121 PUSED   BCI     1,
000051  202020202020  000    122 PUVAR   BCI     9,
000052  202020202020  000
000053  202020202020  000
000054  202020202020  000
000055  202020202020  000
000056  202020202020  000
000057  202020202020  000
000060  202020202020  000
000061  202020202020  000
                      000062 123 7SGGG EBSS    2
```

ERROR LINKAGE

```
000315  000000000000  000
000316  314531632020  000
```

LITERALS

```
000320  000000165140  000
000321  000000000074  000
000322  204733443320  000
000323  202133443320  000
000324  000000202000  000
000325  000000200000  000
000326  000000330000  000
```

BETTER TRY FOR INPUT

```
000327  610000000000    000
000330  000000000100    000
000331  202020200000    000
000332  202020202000    000
000333  000000000001    000
000334  512742202020    000
```

```
                                124          END
  335 IS THE NEXT AVAILABLE LOCATION,   GMAP VERSION  JMPA/030271 JMPE/030271 JMPC/030271
THERE WERE    NO  WARNING FLAGS IN THE ABOVE ASSEMBLY
```

OCTAL	SYMBOL	REFERENCES BY ALTER NO.										
45	A7	26	21	26								
111	ALESS	62	58	62								
215	CKMINS	84	43	84								
216	CKTIME	85	50	85								
4	CLOSE		103									
156	CRDFIL	67	9	67	70	78						
304	ENDCRD	102	78	102								
10	GEBORT		105									
2	GET		78									
21	GETIME		13									
220	HOURS	87	28	31	87							
221	HRSBCD	88	38	46	88							
0	INIT	4	4	64								
166	INPUT	77	77	79								
315	.F.I..		4	5	6	7	77	78	80	81	101	103
217	MINS	86	18	40	86							
3	OPEN		5	6	7							
201	OUTPUT	80	80	82								
42	P7	23	20	23								
501	PRNBLF	73	66	73								
141	PRNFIL	66	12	66	69							
164	PRNOT	69	6	69								
0	PUNBLF	72	65	72								
5	PUNCH		81									
124	PUNFIL	65	10	65	68	81						
163	PUNOT	68	5	68	103							
44	PUNOUT	118	81	118								
1202	RDBUF1	74	67	74								
0	RDWRK	108	78	108								
165	READIN	70	7	70								
222	TAB	89	33	42	89							
262	TIME1	96	14	49	60	96						
274	TIME2	98	24	27	48	98						
300	WRAPLP	101	101	106								

** 18259 WORDS OF MEMORY WERE USED BY GMAP FOR THIS ASSEMBLY.

FLOAT SUBROUTINE

PREFACE

PROGRAM BREAK	265
COMMON LENGTH	0
V CONST BITS	5

PRIMARY SYMDEF	ENTRY
FLOAT	0
CALC	102
TOBIN	146
TOBCD	211
EDIT	223

SECONDARY SYMDEF	ENTRY

	BLOCK	LENGTH
1	IO,WRK	62

SYMREF

FLOAT SUBROUTINE

```
                                  000000
000000  000002710000      010
000001  000260630000      010
000002  000260754000      018
000003  000260741000      010
000004  000101 2350 00     010
000005  000100 7550 00     010
000006  000004 2260 03     000
000007  000010 2270 03     008
000010  000011 6170 00     010
000011  000022 2350 16     010
000012  000000 2360 07     000
000013  106000 4110 03     000
000014  000000 5730 00     000
000015  000026 4570 17     010
000016  777777 0660 03     000
000017  000002 1670 03     000
000020  000100 7530 53     010
000021  000010 6010 00     010
000022  000001710000      010
000023  000000000010      000
000024  000000000002      000
000025  000000000002      000
000026  000000000001      000
000027  000000011007      000
                  000030
                  000032
                  000034
                  000036
                  000040
                  000040
000100  000040 0004 10     010
000101  000040 0004 10     010
```

```
1         TTL     SUBPROGRAM NO. 3 - DATA MANIPULATION SUBROUTINES
2         TTLS    FLOAT SUBROUTINE
4 FLOAT   SAVE

5         LDA     TLYREG
6         STA     REGTLY
7         LDX6    4,DU                LOAD X6 WITH 4
8         LDX7    8,DU                LOAD X7 WITH 8
9 AGAIN   TOV     *+1                 RESET OVERFLOW INDICATOR
10        LDA     ABIN-1,6
11        LDQ     =0                  ZERO Q-REGISTER
12        LDE     =35B25,DU           SET UP E-REGISTER FOR FP CONVERSION
13        FNO                         NORMALIZE
14        DFST    AFLOAT-2,7
15        ADX6    -1,DU               DECREMENT X6 BY ONE
16        SBX7    2,DU                DECREMENT X7 BY TWO
17        SREG    REGTLY,AD
18        TNZ     AGAIN               GO BACK FOR NEXT CONVERSION
19        RETURN  FLOAT
20 ABIN   DEC     8
21 BBIN   DEC     2
22 CBIN   DEC     2
23 DBIN   DEC     1

24 AFLOAT BSS     2
25 BFLOAT BSS     2
26 CFLOAT BSS     2
27 DFLOAT BSS     2
28        EIGHT
29 SAVREG BSS     32
30 REGTLY TALLYB  SAVREG,4,8
31 TLYREG TALLYB  SAVREG,4,8
```

CALCULATE SUBROUTINE

```
                    000102        33 CALC    SAVE
000102  000104710000     010
000103  000260630000     010
000104  000260754000     010
000105  000260741000     010
000106  000030 4330 00   010      34         DFLD    AFLCAT
000107  000036 4630 00   010      35         DFMP    DFLCAT
000110  000034 5670 00   010      36         DFDV    CFLCAT
000111  000032 4770 00   010      37         DFAD    BFLCAT
000112  000262 4630 00   010      38         DFMP    =1D3
000113  216000 4350 03   000      39         UFA     =71825,DU
000114  000141 7560 00   010      40         STQ     EBIN
000115  000121 6040 00   010      41         TMI     EMINUS
000116  000020 2350 07   000      42         LDA     =C20,DL
000117  000144 7550 00   010      43 STORE   STA     ESIGN
000120  000125 7100 00   010      44         TRA     DONE
000121  000052 2350 07   000      45 EMINUS  LDA     =C52,DL
000122  000117 7100 00   010      46         TRA     STORE
000123  000052 2350 07   000      47 FMINUS  LDA     =C52,DL
000124  000136 7100 00   010      48         TRA     STORE1
000125  000030 4330 00   010      49 DONE    DFLD    AFLCAT
000126  000032 5770 00   010      50         DFSB    BFLCAT
000127  000034 5670 00   010      51         DFDV    CFLCAT
000130  000036 5770 00   010      52         DFSB    DFLCAT
000131  000262 4630 00   010      53         DFMP    =1D3
000132  216000 4350 03   000      54         UFA     =71825,DU
000133  000142 7560 00   010      55         STQ     FBIN
000134  000123 6040 00   010      56         TMI     FMINUS
000135  000020 2350 07   000      57         LDA     =C20,DL
000136  000145 7550 00   010      58 STORE1  STA     FSIGN
000137  000143 0540 00   010      59         AOS     SEQBIN
000140  000103710000     010      60         RETURN  CALC
000141  000000000000     000      61 EBIN    DEC     0
000142  000000000000     000      62 FBIN    DEC     0
000143  000000000000     000      63 SEQBIN  DEC     0
000144  000000000000     000      64 ESIGN   DEC     0
000145  000000000000     000      65 FSIGN   DEC     0
```

LET,S CONVERT TO BINARY

```
                    000146
000146  000150710000     010          67 TOBIN  SAVE
000147  000260630000     010
000150  000260754000     010
000151  000260741000     010
000152  000210 2350 00   010          68        LDA     SVTAL3
000153  000207 7550 00   010          69        STA     TAL3
000154  000000 2260 03   000          70        LDX6    ,DU
000155  010000 2370 00   030          71 RTNX   LDAQ    REWRK
000156  000014 7370 00   000          72        LLS     12
000157  000014 7320 00   000          73        QRS     12
000160  010000 7570 00   030          74        STAQ    REWRK
000161  000172 7070 00   010          75        TSX7    CONVRT
000162  000174 7160 16   010          76        XEC     STORE2,6
000163  000001 0660 03   000          77        ADX6    1,DL
000164  000161 7100 00   010          78        TRA     *-3
000165  010005 2350 00   030          79 AROUND LDA     REWRK*5
000166  000202 3750 00   010          80        ANA     MASKA
000167  000200 2270 03   010          81        LDX7    CLT,DL
000170  000200 6270 00   010          82        EAX7    CLT
000171  000173 7100 00   010          83        TRA     CONVRT*1
000172  000202 3770 00   010          84 CONVRT ANAQ    MASKA
000173  000000 7100 17   000          85        TRA     0,7
000174  010001 2350 00   030          86 STORE2 LDA     REWRK*1
000175  010002 2350 00   030          87        LDA     REWRK*2
000176  010004 2350 00   030          88        LDA     REWRK*4
000177  000165 7100 00   010          89        TRA     AROUND
000200  000147710000     010          90 OUT    RETURN  TOBIN
000201  000000011007     000
000202  171717171717     000          91 MASKA  ECOT    171717171717
000203  000000000000     000          92        OCT     000000000000
                    000204           93 MYWRK  BSS     1
000205  000204 0006 00   010          94 TAL1   TALLY   MYWRK,6,0
000206  000204 0006 00   010          95 TAL2   TALLY   MYWRK,6,0
000207  000023 0004 00   010          96 TAL3   TALLY   AEIN,4
000210  000023 0004 00   010          97 SVTAL3 TALLY   AEIN,4
```

```
                                  NOW HOW ABOUT BCD!

                          000211          99 TOBCD  SAVE
000211  000213710000      010
000212  000260630000      010
000213  000260754000      010
000214  000260741000      010
000215  000000 2270 03    000     100          LDX7     ,CU
000216  000143 2350 00    010     101          LDA      SEQBIN
000217  006200 5202 01    000     102          RPT      3,1
000220  000000 5050 17    000     103          BCD      CONTB,7
000221  000257 7560 00    010     104          STQ      SEQBCD
000222  000212710000      010     105          RETURN   TOBCD
```

```
46264 01  10-12-71   12,980      SUBPROGRAM NO. 3 - DATA MANIPULATION SUBROUTINES                PAGE      6

                                    EDIT SUBROUTINE

                                  107 DOIT    MACRO
                                  108         LDAQ    #1
                                  109         LLS     12
                                  110         STA     #1
                                  111         LLS     6
                                  112         GRS     6
                                  113         ORC     =C3.    )000000
                                  114         LRS     6
                                  115         ORC     #2
                                  116         STC     #3
                                  117         ENDM    DOIT
                                  118 MAC1    MACRO
                                  119         LDA     =MTNZ #6
                                  120         STA     #5
                                  121         LDQ     =C202020202020
                                  122 #4      LDA     #1,SC
                                  123 #5      TNZ     #6
                                  124         STC     #2,SC
                                  125         STC     #3,SC
                                  126         TRA     #4
                                  127 #6      LDQ     =MTRA #7
                                  128         STC     #5
                                  129 #7      STA     #2,SC
                                  130         STA     #3,SC
                                  131         TTF     #4
                                  132         ENDM    MAC1
                                  133 EDIT    SAVE

                      000223
000223  000225710000     010
000224  000260630000     010
000225  000260754000     010
000226  000260741000     010
                      000227
000227  000252 2370 00   010     134         DOIT    EBCD,ESIGN,EBCD+1
000230  000014 7370 00   000                 LDAQ    EBCD
000231  000252 7550 00   010                 LLS     12
000232  000006 7370 00   000                 STA     EBCD
000233  000006 7320 00   000                 LLS     6
000234  000264 2760 00   010                 GRS     6
000235  000006 7330 00   000                 ORC     =C330C00000000
000236  000144 2760 00   010                 LRS     6
000237  000253 7560 00   010                 ORC     ESIGN
                      000240                STC     EBCD+1
000251  000224710000     010     136         DOIT    FBCD,FSIGN,FBCD+1
                      000252      137         RETURN  EDIT
                      000254      138 EBCD    EBSS    2
                      000256      139 FBCD    BSS     2
                      000257      140 DUMMY   BSS     1
                      000000      141 SEQBCD  BSS     1
000000  202020202020     000     142         BLOCK   IC,WRK
000001  201020202002     000     143 RDWRK EBCI      5,      8   2     2          1
```

EDIT SUBROUTINE

```
000002  202020202002    000
000003  202020202020    000
000004  202020202001    000
                000005          144         BSS     9
000016  202020202020    000     145 PRTOUT  OCT     202020202020
000017  202020000000    000     146 PRSEQ   CCT     202020000000
000020  202020202020    000     147         OCT     202020202020
000021  202020202020    000     148 PRE     BCI     3,
000022  202020202020    000
000023  202020202020    000
000024  202020202020    000     149 PRF     BCI     3,
000025  202020202020    000
000026  202020202020    000
000027  202020202020    000     150 PRVAR   BCI     6,
000030  202020202020    000
000031  202020202020    000
000032  202020202020    000
000033  202020202020    000
000034  202020202020    000
000035  202020202020    000     151         BCI     6,
000036  202020202020    000
000037  202020202020    000
000040  202020202020    000
000041  202020202020    000
000042  202020202020    000
000043  770200000000    000     152         OCT     770200000000
                000044          153 PUNOUT  BSS     0
000044  202020202020    000     154 PUE     BCI     2,
000045  202020202020    000
000046  202020202020    000     155 PUF     BCI     2,
000047  202020202020    000
000050  202020202020    000     156 PUSEQ   BCI     1,
000051  202020202020    000     157 PUVAR   BCI     9,
000052  202020202020    000
000053  202020202020    000
000054  202020202020    000
000055  202020202020    000
000056  202020202020    000
000057  202020202020    008
000060  202020202020    000
000061  202020202020    000
```

ERROR LINKAGE

```
000260  000000000000    000
000261  264346216320    000
```

LITERALS

```
000262  024764000000    000
```

EDIT SUBROUTINE

```
000263  000000000000    000
000264  330000000000    000
                                158         END
  265 IS THE NEXT AVAILABLE LOCATION.   GMAP VERSION  JMPA/030271 JMPB/030271 JMPC/030271
THERE WERE      1 WARNING FLAGS IN THE ABOVE ASSEMBLY
ON PAGE NO.
    5
```

OCTAL	SYMBOL	REFERENCES BY ALTER NO,							
23	ABIN	20	10	20	96	97			
30	AFLOAT	24	14	24	34	49			
10	AGAIN	9	9	18					
185	AROUND	79	79	89					
32	BFLOAT	25	25	37	50				
102	CALC	33	33	60					
34	CFLOAT	26	26	36	51				
172	CONVRT	84	75	83	84				
36	DFLOAT	27	27	35	52				
125	DONE	49	44	49					
252	ERCD	138	134	138					
141	EBIN	61	40	61					
223	EDIT	133	133	137					
121	EMINUS	45	41	45					
144	ESIGN	64	43	64	134				
254	FRCD	139	136	139					
142	FBIN	62	55	62					
0	FLOAT	4	4	19					
123	FMINUS	47	47	56					
145	FSIGN	65	58	65	136				
260	.E.L,.		4	33	67	99	133		
202	MASKA	91	80	84	91				
204	MYWRK	93	93	94	95				
200	OLT	90	81	82	90				
0	RCWRK	143	71	74	79	86	87	88	143
100	REGTLY	30	6	17	30				
40	SAVREG	29	29	30	31				
257	SEQRCD	141	104	141					
143	SEQRIN	63	59	63	101				
136	STORE1	58	48	58					
174	STORE2	86	76	86					
117	STORE	43	43	46					
210	SVTAL3	97	68	97					
207	TAL3	96	69	96					
101	TLYREG	31	5	31					
211	TCBCL	99	99	105					
146	TCBIN	67	67	90					

** 18357 WORDS OF MEMORY WERE USED BY GMAP FOR THIS ASSEMBLY,

```
5264 02  10-12-71   12,983                                                    PAGE      1
 ORIGIN    062170      ENTRY LOCATION    ENTRY LOCATION    ENTRY LOCATION    ENTRY LOCATION    ENTRY LOCATION

                                SUBPROGRAMS INCLUDED IN DECK,

 011742   101271      START  011742
 011404   101271      INIT   011404     INPUT  011572     OUTPUT 011605     WRAPUP 011704
        BLOCK COMMON  IO,WRK 011320
 011030   101271      FLOAT  011030     CALC   011132     TOBIN  011176     TOBCD  011241     EDIT   011253
        BLOCK COMMON  IO,WRK 011320

                        SUBPROGRAMS OBTAINED FROM SYSTEM  LIBRARY,

 010752   021271     ,SETU, 010760     ,FRSIG 010752
 010744   073069     ,FLTPR 010744
 010632   101970     ,GPNCH 010632     ,GAPNC 010632     PUNCH  010632
 010562   073069     ,GWTRC 010562     ,GAWTR 010562     WTREC  010562
 010242   101970     ,GIL01 010422     ,GILLB 010242
 010126   101970     ,GEDIT 010126     ,GE062 010221     ,GE063 010222     ,GE064 010223     ,GE065 010225
                     ,GE066 010226     ,GE067 010227     ,GAEDI 010126     ,GE068 010230     ,GE069 010231
 007410   062470     ,GE071 010232     ,GE072 010233     EDATE  010234     ETIME  010235     IOEDIT 010126
                     ,GGTBK 007410     GETBK  007410     ,GGET  007412     GET    007412     ,GAGTB 007410
 007402   073069     ,GAGET 007412
 006656   100370     ,GOPNR 007402     ,GCLSR 007402     ,GGETR 007402     ,GPUTR 007402
                     ,GCOPY 006656     COPY   006656     ,GPTBK 006661     PUTBK  006661
 006114   010970     PUT    006664     ,GACOP 006656     ,GAPTB 006661     ,GAPUT 006664     ,GPUT  006664
 005430   020371     ,GOPEN 006114     ,GACPE 006114     OPEN   006114                        ,GFR67 007357
                     ,GCLSE 005430     ,GACLS 005430     ,GR185 005534     ,GR186 005625
 005412   101970     CLOSE  005430                                                           ,GR178 005541
 005220   101970     ,GBNRY 005412
 005172   101970     ,GR200 005220
 005120   101970     ,GBCD  005172
 005042   101970     ,GR250 005120
 004536   062470     ,GR225 005042
 004514              ,GR275 004536
 004344   101970     ,GH960 004521
 004312   101970     ,GR377 004400     ,GR385 004337     ,GR375 004344     ,GR37X 004417     ,GR390 004437
 004206   101970     ,GR990 004312     ,GR991 004333     15AUG5 004380
                     ,GR980 004206     ,GR979 004304     ,GR99X 004212     ,GR984 004250     ,GR985 004304
 003352              ,GR999 004216
                     ,GINHD 003357     ,GOUTH 003856     ,GINTL 003355     ,GOUTL 003354     ,GUSHH 003353
 003350   101970     ,GOVRL 003360     ,GLREA 003444     ,GRCVY 003352     PREVS  003403
                     ,GINID 003350

                                        RANGE                SIZE
                     ALLOCATED CORE     000000 THRU 011777   012000
                     OBJECT PROGRAM
                        RELOCATABLE     003350 THRU 011777   006430
                        COMMON          000100 THRU 002002   001703
                     $      SYSOUT  A1
                     $      SYSOUT  A2
```

ORIGIN 062170 ENTRY LOCATION ENTRY LOCATION ENTRY LOCATION ENTRY LOCATION ENTRY LOCATION

```
$      DATA    A3
     5K. IS THE MINIMUM MEMORY NEEDED TO LOAD THIS ACTIVITY

EXECUTION   PROGRAM ENTERED AT      011742
```

```
EI 011740236000  OI 000010001000  IC 011717  IR 302001  ER 107  AR 000000000000  QR 512742202020  TR 00007755
BA 230012  X0 000000  X1 011712  X2 000000  X3 000000  X4 000000  X5 000000  X6 000000  X7 000003

0000005  000000000000  000000000000  000000000000  000000000080  000000000000  000000000000  000000000000  000000000000
000010   000000000000  000000000000  000000000000  011717022020  000000000000  000305056014  000000000000  000000000000
000020   000000400000  000000000000  011717302001  000000000000  011742000000  000000000000  000000000000  000000000000
000030   000000000000  000000000000  000000000000  000000000000  000000000000  000000000000  000000000000  000000000000
000040   001334006661  001440227752  777212230000  000010000000  000000000000  000175000001  020406020604  002004003347
000050   005344000001  000001011567  011530000000  000000000003  000000000000  000100000100  000000000000  000075230000
000060   000000000000  000000000000  000000000000  000000000000  000000000000  000000000000  216000000000  200240000400
000070   275251274273  411105117323  110511010000  731001070720  202020202020  202020202020  202020041001  077352435145
000100   000101000205  000001000204  000000000000  000000000000  000014400276  202020202020  202020202020  202020202020
000110   202020202020  202020202020  202020202020  202020202020  202020202020  202020202020  202020202020  202020202020
000120   202020202020  000014400276  202020202020  202020202020  202020202020  202020202020  202020202020  202020202020
000130   202020202020  202020202020  202020202020  202020202020  202020202020  202020202020  202020202020  202020202020
000140   202020202020  202020202020  202020202020  202020202020  202020202020  202020202020  000014400276  202020202020
000150   202020202020  202020202020  202020202020  000014400276  202020202020  202020202020  202020202020  202020202020
000160   202020202020  202020202020  202020202020  202020202020  202020202020  202020202020  202020202020  202020202020
000170   000014400276  202020202020  202020202020  202020202020  202020202020  202020202020  202020202020  202020202020
000200   202020202020  202020202020  202020202020  202020202020  202020202020  000014400276  202020202020  202020202020
000210   202020202020  202020202020  202020202020  202020202020  202020202020  202020202020  202020202020  202020202020
000220   202020202020  202020202020  000014400276  202020202020  202020202020  202020202020  202020202020  202020202020
000230   202020202020  202020202020  202020202020  202020202020  202020202020  202020202020  202020202020  202020202020
000240   202020202020  202020202020  202020202020  202020202020  202020202020  202020202020  202020202020  000014400276
000250   202020202020  202020202020  202020202020  202020202020  000014400276  202020202020  202020202020  202020202020
000260   202020202020  202020202020  202020202020  202020202020  202020202020  202020202020  202020202020  202020202020
000270   202020202020  000014400276  202020202020  202020202020  202020202020  202020202020  202020202020  202020202020
000300   202020202020  202020202020  202020202020  202020202020  202020202020  202020202020  202020202020  202020202020
000310   000000000000  000000000000  000000000000  000000000000  000000000000  000000000000  000000000000  000000000000
001300*  000000000000  000000000000  001533000500  000001000227  000016000200  202020202020  201020202002  202020202002
001310   202020202020  202020202001  633031622062  214447432520  302162204546  202651212363  314645214320  472151632031
001320   452063302520  512562644363  202020202020  000016000200  202020202020  200520202000  202020202002  202020202020
001330   202020202001  633031622062  214447432520  302162202126  512123633146  452143204721  516320314520  633025205125
001340   626443632020  202020202020  000016000200  202020202002  000020200305  202020200100  202020202020  202020200105
001350   632562632062  214447432520  633051252520  202020206225  652545204446  512520634620  274620202020  202020202020
001360   202020202020  000016000200  202020200411  110507000000  202020200205  202020202020  202020050000  252066314343
001370   202295202120  474662316331  652520314563  252725517326  206631434320  222520452527  332044316725  242020202020
001400   000016000200  202007030402  010020050102  202004000000  202020202020  202020202000  632562632062  214447432520
001410   263165252020  202020202631  652520444651  252063462027  462020202020  202020202020  202020202020  000016000200
001420   202020202010  000001000000  202020202003  202020202020  202020020500  632562632062  214447432520  623167202020
001430   202020202020  202020202020  202020202020  202020202020  202020202020  202020202020  000016000200  202020111111
001440   111111111111  202020207011  202020202020  202020010000  632562632062  214447432520  622565254520  202020202020
001450   202020202020  202020202020  202020202020  202020202020  202020202020  000016000200  202020200100  000001000000
001460   202020010000  202020202020  202001000000  632562632062  214447432520  253127306320  202020202020  202020202020
001470   202020202020  202020202020  202020202020  202020202020  000016000200  202020202020  200820202000  202020010000
001500   202020202020  202020202090  632562632062  214447432520  453145252020  202020204925  676320464525  203162206330
001510   252043216263  204645252020  202020202020  000016000200  202020200607  101120010203  202020202003  202020202020
001520   202020250505  032562632062  214447432520  632545525220  234645275121  636443216331  464502202646  512027256363
001530   314527203025  512520202020  000000170000  000000000000  000000000000  000000000000  000000000000  000000000000
001540   000030000000  000000000000  000000000000  000000000000  000000000000  000000000000  000000000000  000000000000
```

```
006750  000001175012  007361175000  007104604000  000002224012  000001064003  000008004010  007361064000  000001224012
006760  000003755012  000000235013  000001075012  007361075000  000000755013  007361235000  006773600000  000001224012
006770  000000235012  777777450001  777777744001  000003235011  000022735000  007032601000  000000224012  011364623000
007000  000001235012  007366600000  777777236012  004000316007  007014603000  007400221000  006656620000  777777100011
007010  007014601000  777777236013  006000376007  007252601000  007360755000  000400115003  007020602000  000000235003
007020  000012735000  001400620001  000000011007  000000560201  000000236013  000000756014  007360235000  000400175003
007030  007032600000  007014605000  006670710000  777773235012  001000375007  007051601000  777775235012  007042601000
007040  000017001000  007036710000  170000375003  007051600000  004536700000  007362710000  007362710000  007370710000
007050  007364710000  000003450012  000000235012  002000375007  007057600000  000002235003  007361055000  000002223012
007060  777773235012  010000375007  007065601000  000001235003  000003755012  000002224012  000001064003  000003064012
007070  007361064000  000000744012  000001235012  000003075012  007361075000  000000755013  010606221003  000004220012
007100  000000160013  007134604000  000003740012  006765710000  000004235012  000001175012  007361175000  007134604000
007110  777777235012  004000315007  007123600000  000000235012  777777235001  006000375007  007123600000  000002220012
007120  000003620010  000000100012  007375604000  007127742000  005220701000  007130710000  007400000247  000000000000
007130  777773235012  001000375007  007051601000  007036710000  777777235012  004000315007  007375600000  000003235011
007140  000022735000  007375601000  007354450000  007354054000  000001235012  007355755000  000004235012  000001175003
007150  007361175000  007356755000  006777223000  010606235003  007211600000  007161742000  005220701000  007162710000
007160  007400000277  000000000000  777775220012  007166601000  000017001000  007162710000  777775235012  170000375003
007170  007176600000  004536700000  007362710000  007362710000  007370710000  007364710000  007855235000  007032600000
007200  000002220012  000004235012  000000755010  000002060003  777773235012  010000315007  007210601000  000001060003
007210  000000740012  007355235000  007356175000  007217605000  007355235000  007356755000  000000235003  007355755000
007220  000000224012  007354235000  000001115007  007234601000  006000336007  777777356014  002000235007  007357234000
007230  007240600000  007357275000  007357450000  007240710000  004000275007  007355236000  007240601000  002000275007
007240  777777375007  777777755014  007356220000  777777740014  777773236012  010000316007  007251601000  006002060003
007250  777775740014  007356235000  007360755000  000400115003  007256602000  000000235003  000012735000  001400620001
007260  000000011007  000000560201  000000236013  000000756014  007360235000  000400175003  007270600000  007252605000
007270  007400221000  006656620000  777777100011  007331600000  007354054000  007155710000  000000221003  000003223011
007300  000003224012  004000235007  777772655013  007310743000  007312744000  007412701000  007313710000  007400000423
007310  000000000000  007331000000  000000000000  004000235007  777777275013  006777221000  000000623032  777777235011
007320  000001635001  777773236012  010000316007  007330601000  000001075003  777776450013  777775755013  007331710000
007330  777776755013  006777235000  000000236012  777777235001  777777755002  007341742000  005220701000  007342710000
007340  007400000447  000000000000  777775220012  007346601000  000017001000  007342710000  000000235012  777777235001
007350  006000375007  006000115007  007276601000  007032710000  000000000000  000000000000  000000000276  000000000000
007360  000014000000  000001000000  004521700000  014764632020  004521700000  024764632020  004521700000  034764632020
007370  004344700000  006667000000  000000000000  007051710000  007036710000  004521700000  044764632020  001002810700
007400  010606002000  332723464770  004562236007  000010001000  000102010611  000000000000  332746474551  000000000000
007410  007737450000  007413710000  007737741000  010117450000  007421710000  000000222003  000000223003  000000224003
007420  010122630000  010122754000  010122741000  007415742000  007416743000  007417744000  010023741000  000004235011
007430  010120755000  000002222011  777773235012  760600375007  600600675007  007740601000  000000235012  001700375007
007440  007000115007  007456601000  000003235011  000004236011  007452742000  007453755000  007454756000  007402701000
007450  007455710000  010122000044  000000000000  000000000000  000000000000  007706710000  000002223012  007737235000
007460  007446600000  000000224013  000003104012  007603602000  777773235012  001000375007  000003275012  007502600000
007470  007474742000  005042701000  007475710000  010122000061  000000000000  004306220000  004306235000  777773236012
007500  001000376007  007507601000  777775220012  007506601000  000017001000  007502710000  777776235012  700001336007
007510  100000211007  007531601000  000000236012  005700376007  010124116000  007531601000  100000675007  777777236001
007520  007531601000  000001175003  000001075007  777773236012  001000376007  007530600000  004306755000  007531710000
007530  777776755012  777777375003  000003755012  000000635010  170000375003  007736601000  000002223012  000001224003
007540  777772064012  777772744012  777773235012  010000375007  007706601000  000001164013  010046601000  000000235012
007550  001700375007  000200115007  007572601000  777773235012  001000375007  007560600000  004306724000  007561710000
007560  777776724012  700000304003  010052601000  007777364003  007566744000  000001724013  000000024003  000001024003
007570  000004104012  010052601000  000001235013  000022735000  007600751070  000002224012  000002064003  000000744012
007600  000227064003  000003744012  007605710000  000000224013  000000744012  000000235012  000002735000  000003375003
007610  000003115003  007750603000  007613710001  007631710000  007625710000  007624743000  007622742000  000005701032
007620  007623710000  010122000220  000000000000  007622222000  000000223003  000001220012  007636601000  004521700000
007630  052725632020  000000224032  000001744012  007713600000  000001224003  000000044012  000000224012  000001064012
007640  000000744013  010023221000  010120235000  007706600000  000022735000  007706601000  010120224000  000000223012
007650  000001235012  007737755000  000400115003  007656602000  000000235003  000012735000  001400620001  000000560201
007660  000000236013  000000756014  007737235000  000400175003  007706600000  007651605000  777777235012  004000315007
007670  007706600000  000000223012  777777235013  006000375007  007706600000  777777223013  010117043000  006000115007
007700  007704601000  010117223000  000001743012  007706710000  010120744000  007470710000  007415710000  000000224012
007710  000001064003  000000744012  007605710000  000000235012  001700375007  000200115007  007627600000  000000224012
007720  000001064003  000000744013  777775235012  007726601000  000017001000  007722710000  770077235003  777775355012
007730  000000235032  770000375007  000014735000  777775255012  001700175003  010023601000  010021710000  011577000000
007740  010200675007  007752600000  000200675007  005120700000  010021710000  007766710000  007752710000  010025710000
007750  004521700000  022725632020  757400235003  000022731000  777773355012  400000235003  777775755012  007763742000
007760  005042701000  007764710000  010122000356  000000000000  010023221000  007431710000  010023221000  000003220011
```

007770	07414235020	000000740001	007415710000	00453670000	721710000	0100147100000	0100317100000	0100277100000
010000	400000220003	001000235007	777773375012	007506600000	776235012	004306755000	004312701000	0100117100000
010010	010172000401	400000220003	004306235000	007507710000	000200235007	777773255012	000001220003	777772040012
010020	010023710000	020000235007	777773255012	011577221003	007766710000	00452170000	012725632020	004521700000
010030	012725632020	004344700000	007414000000	000000000000	010113710000	777775235012	010041001000	000017001000
010040	010035710000	170000375003	007773601000	777776220012	000003740012	010000710000	000000235012	001700375007
010050	000200115007	010062601000	003352701000	010056710000	010122000443	005042000000	010034234000	010062600000
010060	010073221000	007431710000	777773220012	010066601000	004521700000	042725632020	000002236012	000022776000
010070	000002756012	004344700000	007414000000	000001000000	000002223012	000001220013	777772740012	777773235012
010100	001000375007	007572600000	000002236012	000022776000	000002756012	010111742000	005042701000	010112710000
010110	010122000475	000000000000	007572710000	004405220000	000006100003	010000601000	010021710000	000000000000
010120	011320000000	001100306610	011710502000	332727632242	000000004200	000000000000	010132710000	000000222003
010130	000010223003	010237635000	010237541000	010237741000	010127742000	010130743000	000002222011	000003223011
010140	010201600000	000021001000	010234755000	341000036007	702000000007	000044737000	005172701000	010151710000
010150	010237000330	010235752007	000006736000	010235752000	000000235012	000022735000	010175600000	000007115003
010160	010162600000	010175603000	000000235012	010163710025	010173710000	010202710000	010204710000	010207710000
010170	010212710000	010214710000	010216710000	010221755000	010233054000	000001123003	010201000000	000001002003
010200	010154710000	010127710000	010222755000	010174710000	010225751070	010230054000	010175710000	010226751070
010210	010231054000	010175710000	010223751070	010175710000	010227751070	010175710000	000000235001	010232755000
010220	010175710000	000000000000	000000000000	010224100000	000000000000	000000000000	000000000000	000000000000
010230	000000000000	000000000000	000000000074	000000000000	202020202020	202033202020	001101030610	000000010000
010240	332725243163	000000000000	010245710000	000000222003	010560630000	010560754000	010500741000	010243742000
010250	010325741000	000003220011	010536000000	000000222003	010225221000	000000235011	000001236011	010452117012
010260	010312600000	010366700000	010225221000	000000235011	000001236011	010432757012	010452757012	010422757000
010270	010424377000	010426757012	010424337000	010442177000	010432355012	010433396012	010432237012	000030773000
010300	010436117012	010310600000	000541270000	010305710000	010560000047	000001076007	010432756012	010311710000
010310	010432450012	010325710000	010432235012	010432054012	000127701000	010317710000	010500000057	010436117012
010320	010322601000	010432450012	000030737000	010426277012	010922757000	000000621000	000002235011	010346601000
010330	010422235003	000000275007	010444755000	000004235011	000000275007	010445755000	010445235052	010457236005
010340	000006737000	010445755052	000006737000	010445755052	010336607000	010243710000	000004222011	010422237000
010350	000001756012	000001756012	010243710000	000002222003	010226221000	000000235011	000001236011	010452117012
010360	010312600000	010366700000	010226221000	000000235011	000001236011	010265710000	000000235007	010444755000
010370	010444741000	010456450000	010444235052	000012115007	010400603000	010456054000	010372607000	010402710000
010400	010456450000	010372607000	010456236000	000006402007	000022736000	010477752070	010446237000	000000737000
010410	010424757000	000001236007	010456176000	000006402007	000022736000	010417752070	010450237000	000000773000
010420	010436757012	000000710010	000000000000	000000000000	000000000000	000000000000	000000000000	000000000000
010430	000000000000	000000000000	000000000000	000000000000	000000000000	000000000000	000000000000	000000000000
010440	000000000000	000000000000	000000000000	000000000001	000000000000	000000000000	777777777777	777700000000
010450	000000001111	111111111111	202020202020	202020202020	202020202020	202020202020	000000000000	100000000000
010460	040000000000	020000000000	010000000000	004000000000	002000000	001000000000	000400000000	000200000000
010470	000 [illegible] 000	020200000000	010200000000	004200000000	002200000000	001200000000	000600000000	000000000000
010500	440[illegible] 00	420000000000	410000000000	404000000000	402000000000	401000000000	400400000000	400200000000
010510	400[illegible] 00	400000000000	410200000000	404200000000	402200000000	401200000000	400600000000	300000000000
010520	240000000000	[illegible] 000000000000	210000000000	204000000000	202000000000	201000000000	200400000000	200200000000
010530	200100000000	200000[illegible] 000	210200000000	204200000000	202200000000	201200000000	200600000000	500000000000
010540	140000000000	120000000000	110000000000	104000000000	102000000000	101000000000	100400000000	100200000000
010550	100100000000	120200000000	110200000000	104200000000	102200000000	101200000000	100600000000	000100030606
010560	000000000000	332731430001	010565710000	011530222003	010563630000	010630754000	010630741000	010563742000
010570	000004235031	000077375007	000006735000	010626755000	000005235031	000077375007	010626255000	010626235000
010600	007357755000	000002222011	010610742000	000003235011	010611755000	000664701000	010612710000	010630000034
010610	011530000000	011364000000	000000235012	600000375007	010625601000	000000220012	000000235007	777777315010
010620	010625601000	777777235010	777777375003	010626275000	777777755010	010563710000	000000000276	000100030606
010630	010677102000	332766635123	010635710000	000000222003	010741630000	010741754000	010741741000	010633742000
010640	000002222011	010701742000	000003235011	010702755000	000001236012	010735756000	000004235031	010715755000
010650	010664600000	000014116003	010654600000	000014236003	010230234000	010660600000	010710752070	000002076003
010660	000001756012	000002235007	010736755000	010676710000	000030116003	010667600000	000030236003	010230234000
010670	010673600000	010710752070	000000076003	000001756012	000001235007	010736755000	010502701000	010705710000
010700	010741000055	011530000000	011364000000	010736100000	010737000000	010230234000	010720600000	000000221012
010710	000000061003	010717741000	010242701000	010720710000	010741000063	000000000001	000001000000	000000000000
010720	010735235000	000001755012	000200235007	777773375012	010733600000	010734235000	000000355012	010736235000
010730	000001731000	000016735000	000000235012	010633710000	777777637777	000016000000	000000000002	000000000076
010740	000100030606	011612502000	332747452330	000000700000	000007450000	000011450000	000000710011	000103010611
010750	000000000000	332643634751	000000710011	002004003347	000000000000	003350000000	000000000000	011742000000
010760	010755222000	010756235000	000100102003	010770600000	000001122003	000000755012	000100102003	010764601000
010770	000000221003	030200520201	000000450011	000031450000	010755235000	000000635005	011000000000	011022035000
011000	000015755000	010754235000	000025755000	010757235000	000024755000	010753235000	000037755000	010744701000

011010	011012710000	011022000047	000000220003	000000222003	000000223003	000000224003	000000225003	000000226003
011020	000000227003	010757701051	000000710000	011025000050	000102057701	000000000000	336225636433	000000000000
011030	011032710000	011310630000	011310754000	011310741000	011131235000	011130755000	000004226003	000010227003
011040	011041617000	011052235016	000000236007	100000411003	000000573000	011056457017	777777066003	000002167003
011050	011130753053	011040601000	011031710000	000000000010	000000000002	000000000002	000000000001	000000011007
011060	010400000000	000000000000	004400000000	000000000000	004000000000	000000000000	002400000000	000000000000
011070	001400011754	000000000000	000000000000	000003000006	200000000000	000000000000	002000000000	000064460000
011100	001400011754	000000000000	000000000000	000002000004	200000000000	000000000000	004000000000	000064440000
011110	001400011754	000000000000	000000000000	000001000002	200000000000	000000000000	004000000000	000064420000
011120	001400011754	000000000000	000000000000	000000000000	200000000000	000000000000	010000000000	000064400000
011130	011130000010	011070000410	011134710000	011310630000	011310754000	011310741000	011060433000	011066463000
011140	011064567000	011062477000	011312463000	216000435003	011171756000	011151604000	000020235007	011174755000
011150	011155710000	000052235007	011147710000	000052235007	011166710000	011060433000	011002577000	011064567000
011160	011066577000	011312463000	216000435003	011172756000	011153604000	000020235007	011175755000	011173054000
011170	011133710000	000000013560	000000003720	000000000012	000000000020	000000000020	011200710000	011310630000
011200	011310754000	011310741000	011240235000	011237755000	000000226003	011320257000	000014737000	000014732000
011210	011320757000	011222707000	011224716016	000001066003	011211710000	011325235000	011252375000	011230227003
011220	011230627000	011223710000	011232377000	000000710017	011321235000	011322235000	011324235000	011215710000
011230	011177710000	000000011007	171717171717	000000000000	000000000000	011234000600	011234000600	011053000400
011240	011053000400	011243710000	011310650000	011310754000	011310741000	000000227003	011173235000	006200520201
011250	000000505017	011307756000	011242710000	011255710000	011310630000	011310754000	011310741000	011302237000
011260	000014737000	011302755000	000006737000	000006732000	011314276000	000006733000	011174276000	011303756000
011270	011304237000	000014737000	011304755000	000006737000	000006732000	011314276000	000006733000	011175276000
011300	011305756000	011254710000	203320332033	203320202020	203320332033	203320202020	000000000000	003720171717
011310	011765502000	264346216320	024764000000	000000000000	330000000000	000000000000	000000000000	000000000000
011320	202006071011	000020010203	202020202003	202020202020	202002050505	632562632062	214447432520	632545525220
011330	234645275121	636443216331	466562202646	512027256363	314527203025	512520202020	202020202020	202020202020
011340	202020202020	202020202020	202020202020	202020202020	202020202020	202020202020	202020202020	202020202020
011350	202020202020	202020202020	202020202020	770200000000	202020202020	202020202020	202020202020	202020202020
011360	202020202020	202020202020	202020202020	202020202020	202020202020	202020202020	202020202020	202020202020
011400	202020202020	202020202020	000000000000	000000000000	011406710000	011721630000	011721754000	011721741000
011410	006114701000	011415710000	011721000005	011567000000	000001000000	006114701000	011427710000	011721000006
011420	011570000000	000001000000	006114701000	011427710000	011721000007	011571000000	000001000000	000016226003
011430	011563746000	011531746000	000026226003	011546746000	000021001000	011666755000	000006732000	011724506000
011440	011725506000	011623755000	000014116007	011446600000	011451604000	000014176007	011726235000	011701755000
011450	011453710000	011727235000	011701755000	011624756000	000000636000	000000226003	011624235000	014200520201
011460	011626505016	000011116007	011646603000	011730276000	011731276000	011625756000	000000226003	011623235000
011470	014200520201	011626505016	011621756000	011732276000	000022736000	011625235000	000022733000	011700756000
011500	011666235000	011622755000	000014733000	000006732000	011733276000	000014733000	000006732000	011733276000
011510	011734115000	011515604000	011735275000	011666757000	011517710000	011736275000	011513710000	011405710000
011520	000000000000	000000000000	000001000001	000000040400	000000002101	400000000000	000000000000	000000000000
011530	000272002000	000016000000	000100000100	000500000000	000500000000	000000000000	000000000000	000000000001
011540	000000040400	000000002102	400000000000	000000000000	000000000000	000000002000	000026000000	000601000601
011550	000500000000	000500000000	011553000001	000500036005	000001000001	000000000400	000000002103	401700000000
011560	000003040000	000000100000	000000000600	000000000000	001302001302	001533000000	000500000000	011534000000
011570	011545400000	011562200000	011574710000	011721630000	011721754000	011721741000	000741270100	011604710000
011600	011721000116	011562000000	011710000000	011320000000	011573710000	011607710000	011721630000	011721754000
011610	011721741000	010632701000	011617710000	011721000121	011530000000	011364000000	011737000000	011606710000
011620	000003000000	000000000511	010001020701	000000000073	000000000014	000000200102	000003032400	000002342000
011630	000001750000	000001440000	000001200000	000001000000	000000000030	772020202020	202020202020	202020202020
011640	202020202020	202020202020	202020202020	202020202020	202020202020	202020202020	202020202020	304645257066
011650	254343204421	514225633145	272025246423	216331464520	274421472023	466451622520	770200000000	000000011007
011660	202020202020	202020202020	202020202020	202020202020	202020202020	202020202020	202020200100	610102610701
011670	202020202020	202020202020	202020202020	202020202020	202020202020	202020202020	202020202020	202020202020
011700	200102330511	204733443320	234220202020	770400000000	011706710000	011721630000	011721754000	011721741000
011710	011711710000	005430701000	011716710000	011721000147	011567000000	000003000000	011740236000	000010001000
011720	011705710000	011746502000	314531632020	000000000000	000000165140	000000000074	204733443320	202133443320
011730	000000202000	000000200000	000000330000	610000000000	000000000100	202020200000	202020202000	000000000001
011740	512742202020	000000000000	011404701000	011745710000	011776000003	011572701000	011750710000	011776000004
011750	011176701000	011753710000	011776000005	011030701000	011756710000	011776000006	011132701000	011761710000
011760	011776000007	011241701000	011764710000	011776000010	011253701000	011767710000	011776000011	011605701000
011770	011772710000	011776000012	011745710000	011704701000	011776710000	011776000014	000000000000	626321516320

UPPER SSA PLUS SLAVE PREFIX

777000M	226003777700	227030000110	011717302001	000010000000	004203000200	340300000000	340100000000	006346102200
777010	1334006661	001440227752	777212230000	000010000000	24202200	000175000001	000000000000	000075410000

```
777040
777050
777060
777070
777100
777110
777120
777130
777140
777150
777160
777170
777200
777210
777220
777230
777240
777310*
777320
777330
777340
777350
777360
777370
777710*
777720
777730
777740
777750
777760
777770
000000
000010
000020
000030
000040
000050
000060
000070
```

COMMON SYSTEM FILES

FILE CODE	NAME	REMARKS
A*	ALTER FILE	Created when System Input (GIN) reads $ UPDATE card. $ ALTER cards and source cards put on A*.
B*	OBJECT PROGRAM FILE	Created by General Loader (GLOAD) when $ EXECUTE included in source deck and no fatal error occurred during compilation. (General Loader loads from B* rather than from R* for compile-and-execute option)
C*	BINARY DECK FILE	Created when binary deck option ($ GMAP DECK) is specified. Goes to SYSOUT--report code: 76 . 8
G*	GMAP SOURCE FILE	System Input (GIN)creates G* when it sees $ GMAP. COBOL & FORTRAN create G* as output file. Used as input to GMAP Assembler.
H*	PROGRAM LINK FILE	Used for temporary storage of programs that are called into and out of memory.
I*	DATA STORAGE FILE	Used as storage area for input data required by User Programs.
K*	COMPRESSED DECK FILE	Created when COMDK option specified in language processor control card ($ GMAP COMDK).
L*	SYSTEM SUBROUTINE LIBRARY FILE	Appropriate System Subroutines loaded as required.
P*	SYSTEM OUTPUT FILE	Collection of all Systems Output.
R*	LOADER INPUT FILE	Normal Loader Input File. Includes directives to General Loader (GLOAD) as well as binary programs. $SOURCE directive generated by system and placed on R* for compile-and-execute jobs instructs General Loader to load from B* rather than R*.

SERIES 6000

GMAP PROGRAMMING

COURSE 605

EXERCISES

PROBLEMS

BINARY AND OCTAL REPRESENTATION

1. Do the following number conversions:

   a. $(285)_{10} = (\quad)_8 = (\quad)_2$

   b. $(.75)_{10} = (\quad)_8 = (\quad)_2$

   c. $(65)_8 = (\quad)_{10} = (\quad)_2$

   d. $(.101)_2 = (\quad)_8 = (\quad)_{10}$

   e. $(1101)_2 = (\quad)_8 = (\quad)_{10}$

   f. $(.31)_8 = (\quad)_{10} = (\quad)_2$

   g. $(1011011)_2 = (\quad)_8 = (\quad)_{10}$

   h. $(8)_{10} = (\quad)_8 = (\quad)_2$

2. Do the following arithmetic operations:

   a. $(1011)_2 + (111)_2 = (\quad)_2$

   b. $(23)_8 + (57)_8 = (\quad)_8$

   c. $(1110)_2 - (11)_2 = (\quad)_2$

   d. $(6307)_8 - (1436)_8 = (\quad)_2$

   e. $(111011)_2 - (11001)_2 = (\quad)_2$

   f. $(110)_2 + (11)_2 + (101)_2 + (111)_2 = (\quad)_2$

DATA MOVEMENT AND ARITHMETIC OPERATIONS

1. Move 150 words from "Table A" to "Table B." Check each word and:

   a. If negative, add one to "NEG" counter.

   b. If zero, add one to "ZERO" counter.

   Update your instruction word addresses without using register modification.

2. Repeat the problem for Step #1, except reverse the order of the Table and use register address modification.

3. Show the coding to complement a negative number back into its true form. Do not use the "NEG" instruction.

4. Show the coding to perform the following calculation:

$$\frac{(A \cdot B) - C}{D} + E = F$$

Note: If D equal to zero, store zero's in "F."

DATA REPRESENTATION AND SHIFT OPERATIONS

1. Create the following values in memory by means of pseudo operations:

   A. 1492_{10}

   B. 3466_{8}

   C. 1903_{10} in upper 18 bits

   D. 770077_{8} in upper 18 bits

   E. 3.1416_{10}

   F. 7.9×10^{3}

   G. 83.5×10^{9} in double precision

   H. 183×10^{-3} scaled to Bit 24

   I. 6 bits of BCD X, 6 bits octal 72, 6 bits of 28_{10}, 18 bits of address XRAY

   J. The statement "THREE-D SYSTEM"

2. Shift the binary number in memory location "Place" three binary places to the left and store in "New"

   A. Question: What effect does this operation have on this value?

3. Add "A" (= 14.73B23) to "B" (= 6.097B27) and store in Location "C"

4. Show the control cards and deck setup necessary to perform the following jobs:

   A. GMAP assembly and execute

   B. GMAP and FORTRAN assembly and execute

   C. Binary object deck and execute

INDIRECT THEN TALLY ADDRESS MODIFICATION

1. Code Problems 1 and 2 of Exercise #2 using Tally operations.

2. There are ten (10) words of 6-bit BCD characters beginning at Location "TABLEA." Completely reverse the characters so that the first character of the first word becomes the last characters of the last word and so on. Store result in TABLEB.

Write the GMAP coding to do the following:

3. Cause a fault trap to occur.

4. Load A with B indirect.

5. Move 5 characters starting with character position 3 in TAB to PRTOUT starting in character position 5. Use tally modification.

6. Move a 50 word table (TAB1) to TAB2 using tallies.

7. Do part 6 using LDAQ, STAQ.

8. Give the major difference between CI and SC modification.

MISCELLANEOUS INSTRUCTIONS

1. Code Problem #1 of Exercise #2 using a Repeat type instruction.

2. Convert a binary value in Location "BIN" to BCD and store in "BCDNR." The maximum size of the binary number will not exceed 7 BCD digits when converted.

3. A physical record of 1000 words starts at Location "WORD" and is composed of 10-word logical records. If the first word of a logical record is negative, the logical record is to be moved to a second table starting at Location "WORK."

4. Write the GMAP coding to do the following:

   a. Move a 50 word table using indexing from TAB1 to TAB2.

   b. Do part "a" using RPD.

   c. Find the data record with a key of 2AØ in a 50 record chain starting at MASCHN.

   d. Count all numbers <0 or >50 in table TAB which is 100 words long.

5. Show the contents of the instruction word.

   a. `RPT 10,1`

   b. `RPDA 25,2,TZE,TMI`

PSEUDO OPERATIONS

List the pseudo-operation which will do the following:

1. Reserve 500 words of core with the first word labeled A1.
2. Reserve 500 words of core with word 501 labeled A2.
3. Program documentation
4. Page title
5. Termination of assembly
6. Creates entry points
7. Used to begin a subroutine
8. Specifies labeled or blank common mode.

PROBLEM: Find the Coding Errors only and describe them in the Comments Field

Do not concern yourself with possible programming logic errors.

PROGRAMMER | DATE | PAGE | OF

LOCATION (1 2 6)	E/O (7)	OPERATION (8 14)	ADDRESS, MODIFIER (15 16)	COMMENTS (32 72)	IDENTIFICATION (73 80)
		ORG	2358		
END		LDA	GOOD		
		STA	.OCT		
		ANA	ONE,X7		
GO		BOOL	2286		
	E	RPD	29,3,TZE,TRA		
		LDAQ	REG,6		
		CMPQ	.OCT		
		TTF	IC-2		
		ANDQ	HI		
		TZE	HALT		
. HELP		LDX	2,0,DU		
		ARL	24,7		
		ASQ	BYE,DU		
GOOD	O	SREG	REG		
A=3		TMI	.HELP		
HALT		MME	GESTOP		
		ORG	712345		
GOOD		BCI	4,I hope you got this correct.		
REG	7	BSS	8		
BYE		ZERO	HELP,		
HI		DEC	66724		
. OCT		OCT	123456787654		
		END	END		

MACRO OPERATIONS

1. Write a conditional assembly MACRO to move a variable number of words from one location to another. (Range 1-4095 words to be moved.) Also provide the option to either check and count ZERO and NEGATIVE words, or to ignore them. (Use the conditional PSEUDO OPS.)

2. Write a MACRO to move n words from argument #2 to argument #3. N is argument #1. Write an example of the CALL.

3. Write a MACRO to: swap N words from two given locations. Write an example of the CALL.

1. Write the coding for the File Control Block, File Designator Word, and Buffers for an input tape file with the following parameters:

Fixed length records	
Record size	60 words
Block size	300 words
Two buffers	
File Code	AB
Standard labels	
Multireel file	
File name TAPE-FILE	
Location of File Control Block	TPFILE
Location of File Designator Word	FILEWD

2. Write the coding for the File Control Block, File Designator Word, Buffers and Working Storage Area for an input BCD card file with the following parameters:

File Code I*	
System Standard Format	
One Buffer	
Location of File Control Block	CDFILE
Location of File Designator Word	CARDWD

3. Give the effective address of the following examples:

a.
```
        LDA     A,*
        ~
A       LDQ     B,*5
        ~
B       LDX5    C,1
```

b.
```
        LDA     *+1,I
        LDQ     *-1,*
```

3.

```
c.              LDX1        B,1*
                ~
B+C(X1)         STX1        C,*1
                ~
C               ADX1        D,*5
                ~
D               SBX1        10,DU
                ~
10+C(X5)        ADX1        15,DU
```

4. Form the one's complement of ABLE in BAKER.

5. Turn on bits 1, 13-15, and 27 and turn off all others in SAM.

1. Write the coding for the File Control Block, File Designator Word, Buffer, and Working Storage Area for an output card file with the following parameters:

   File Code P*
   Standard System Format
   One Buffer
   Location of File Control Block PUNCD
   Location of File Designator Word PDWD

2. Using the card file described in No. 1 above, write the following File and Record Control (GFRC) CALL's:

   a. OPEN
   b. CLOSE

3. Write the coding for the File Control Block, File Designator Word, Buffer for an output tape file with the following parameters:

   File Code TO
   Variable Length Records
   Record Size 50 words
   Block Size 200 words
   Two Buffers
   Location of File Control Block OTFILE
   Location of File Designator Word OTWD

4. Using the tape file described in No. 3 above, write the following File and Record Control (GFRC) CALL's.

   a. OPEN
   b. CLOSE
   c. PUT

5. Using CALL WTREC, write the coding to punch a binary card and other required parameters.

6. Using CALL WTREC, write the coding to punch a Hollerith card and any other required parameters.

7. Using CALL WTREC, write the coding and any other required parameters to print a 20-word line.

8. Using CALL WTREC, write the coding and any other required parameters to create a tape record of 15 words.

9. Using CALL PUNCH, write the coding and any other required parameters to punch a binary card.

10. Using CALL PUNCH, write the coding and any other parameters to punch a Hollerith card.

11. Using CALL PRINT, write the coding and any other parameters to write a 22-word line. Skip to the top of page after printing.

12. Write the IOEDIT control parameters and codes to insert 30 characters in the first line of the heading.

13. Write the IOEDIT control parameters and codes to print a sub-heading.

14. Write the IOEDIT control parameters and codes to insert a sequence number in punched cards beginning with 00000100.

GMAP WORKSHOP PROBLEM
(Optional)

PROBLEM DEFINITION

The GMAP Workshop problem consists of reading an input file of data cards, performing calculations on fields contained in these cards and printing the results on a report.

The input cards have four data fields labeled A, B, C & D respectively and a text field. The numeric input data from fields A, B, C & D must be converted from BCD values to binary values for calculations. After performing the calculations on these four fields, the results should be converted to BCD for output. The text field is then moved intact to the output areas.

Print line images should be edited by inserting decimal points and suppressing leading zeroes. These results should be formatted as specified.

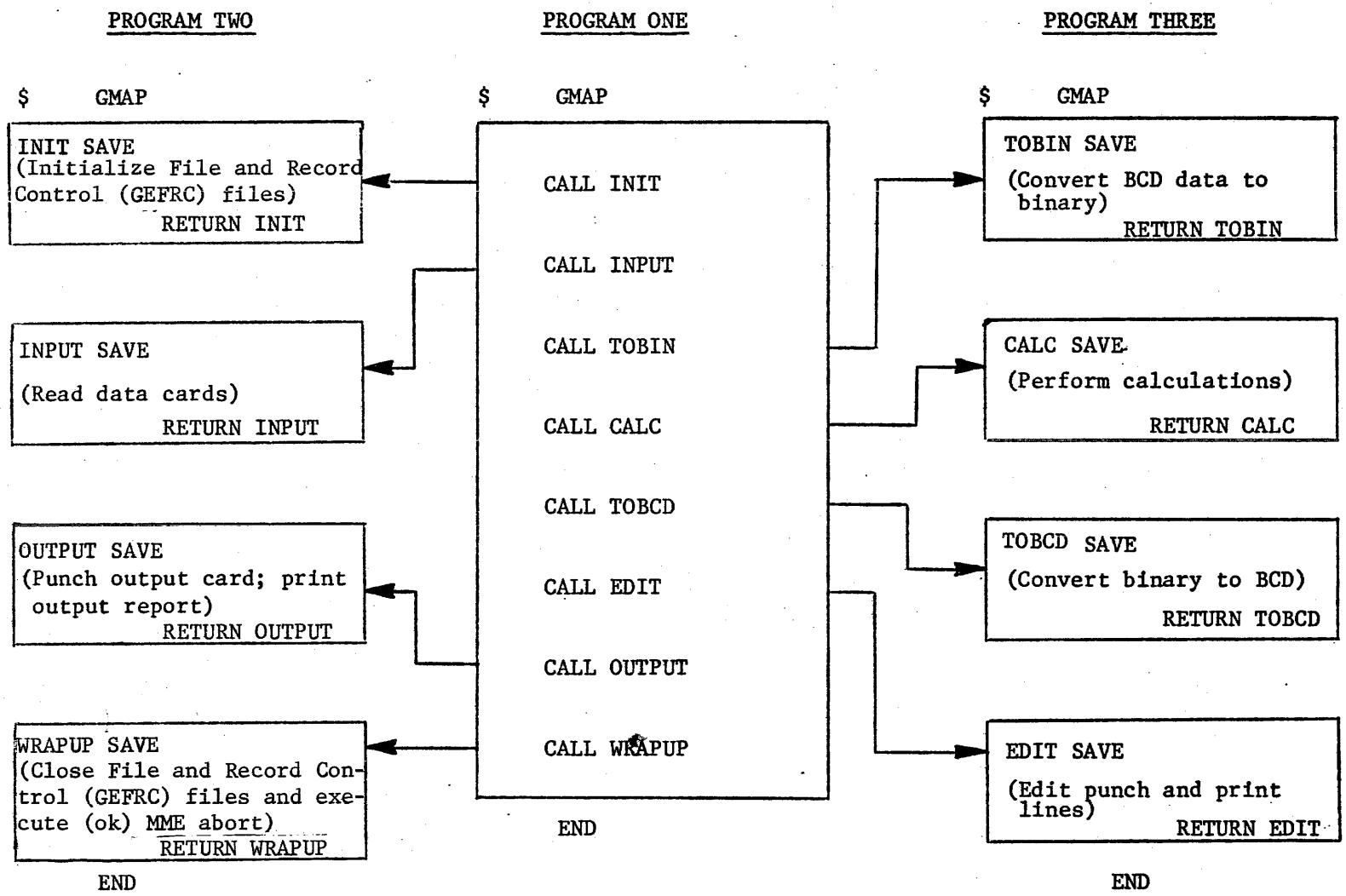
The program solution consists of three subprograms which form one program. Sub-program ONE is the main or executive routine that consists solely of calls to the subroutine modules contained in subprogram TWO and subprogram THREE. The sequence of the calls inserted into subprogram ONE determines the sequential order of processing at execution time. This process is repeated until all data cards have been processed and the job is terminated.

GMAP COURSE

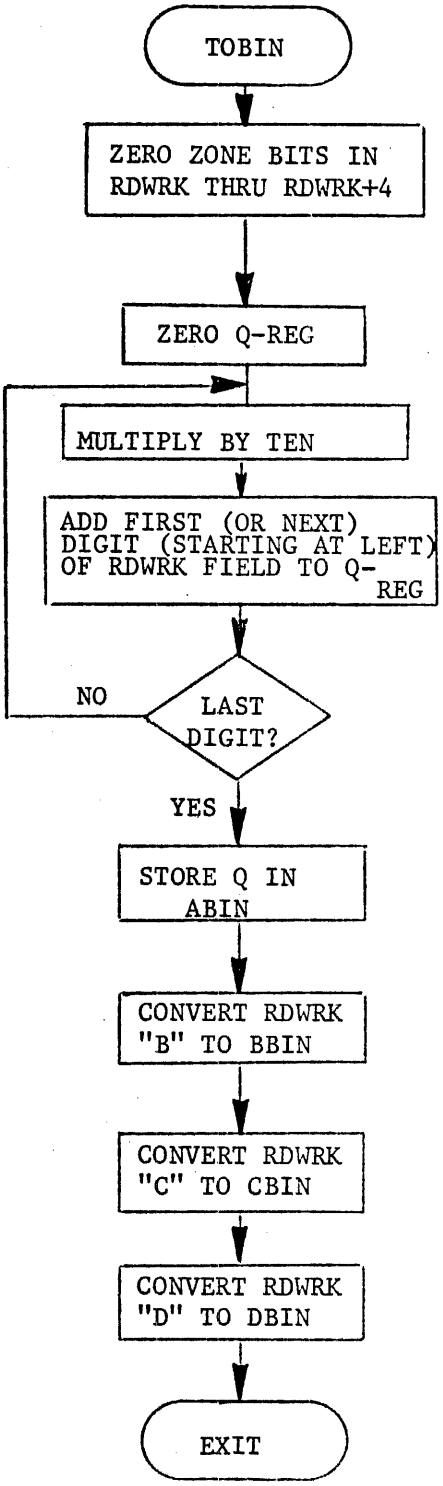
WORKSHOP PROBLEM JOB STACK
(Assignment No. 1)

```
$           SNUMB
$           IDENT
$           GMAP            NDECK           (program one)
            .
            .
            .                               (source cards)
            END
$           GMAP            NDECK           (program two)
INIT        .
INPUT       .                               (source cards)
OUTPUT      .
WRAPUP      .
            END
$           GMAP            NDECK           (program three)
TOBIN       .
            .
CALC        .                               (source cards)
TOBCD       .
EDIT        .
            END
$           EXECUTE         DUMP
$           LIMITS          ,5K,,1000
$           ENDJOB
***EOF
```

GMAP WORKSHOP ASSIGNMENT

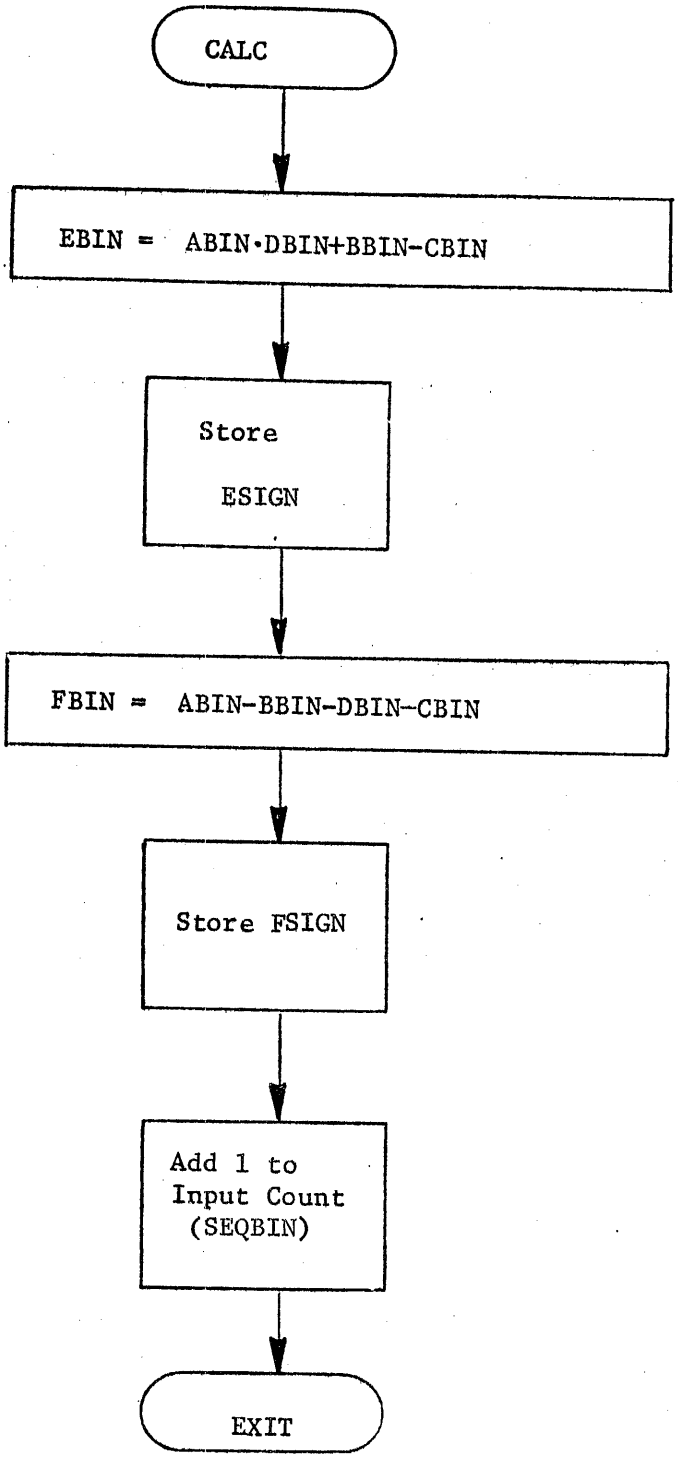


CLASS PROBLEM



ASSIGNMENT NO. 2

RDWRK contains the input card image.



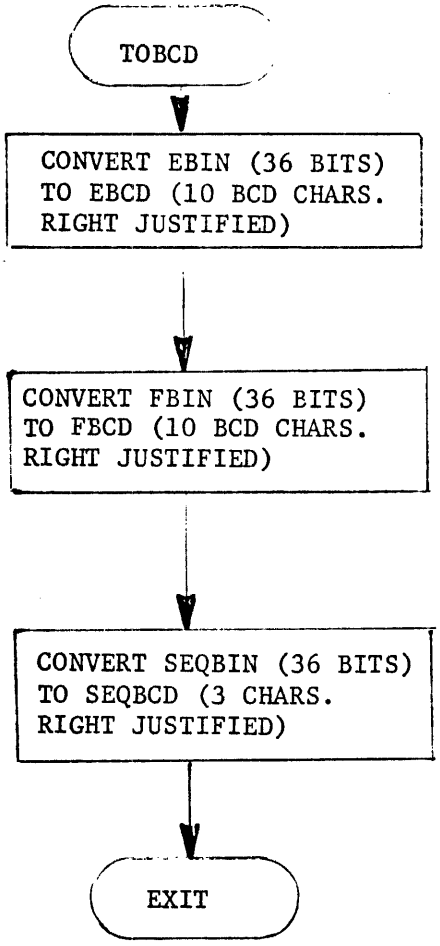
ASSIGNMENT NO. 3

1. Assume EBIN, FBIN, and SEQBIN single precision fixed point binary numbers with a binary point to right of Bit 35.

2. ESIGN and FSIGN will contain the sign of E and F, respectively.

GMAP COURSE

CLASS PROBLEM



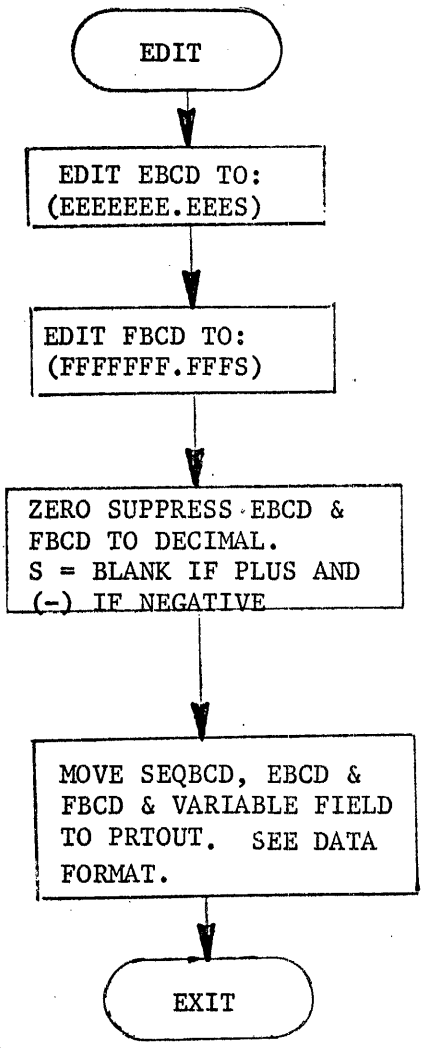
ASSIGNMENT NO. 4

1. EBCD and FBCD are double precision words.

2. SEQBCD is a single precision word.

GMAP COURSE

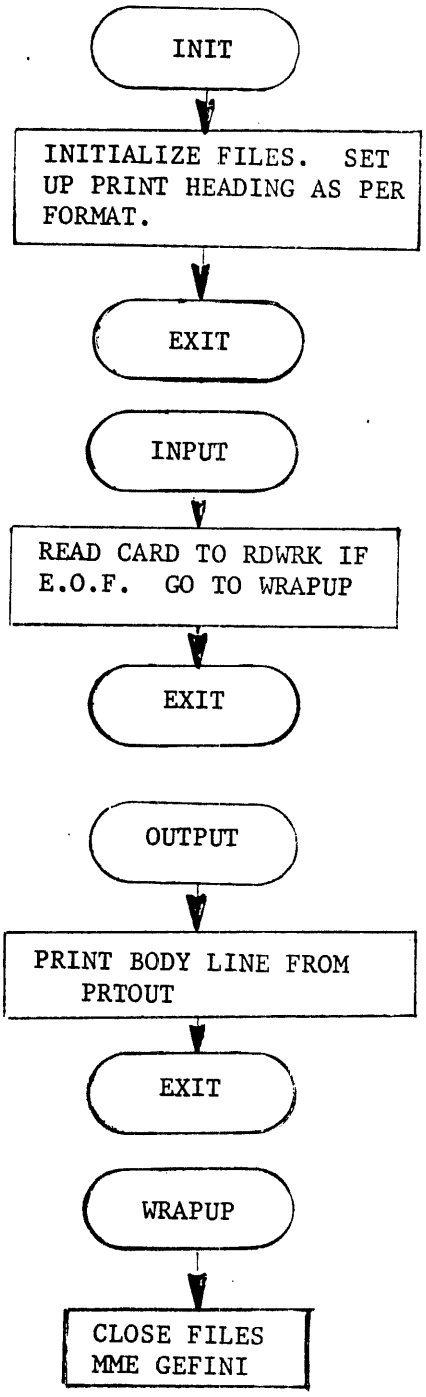
CLASS PROBLEM



ASSIGNMENT NO. 5

PRTOUT located in labeled common IØ.WRK.

CLASS PROBLEM

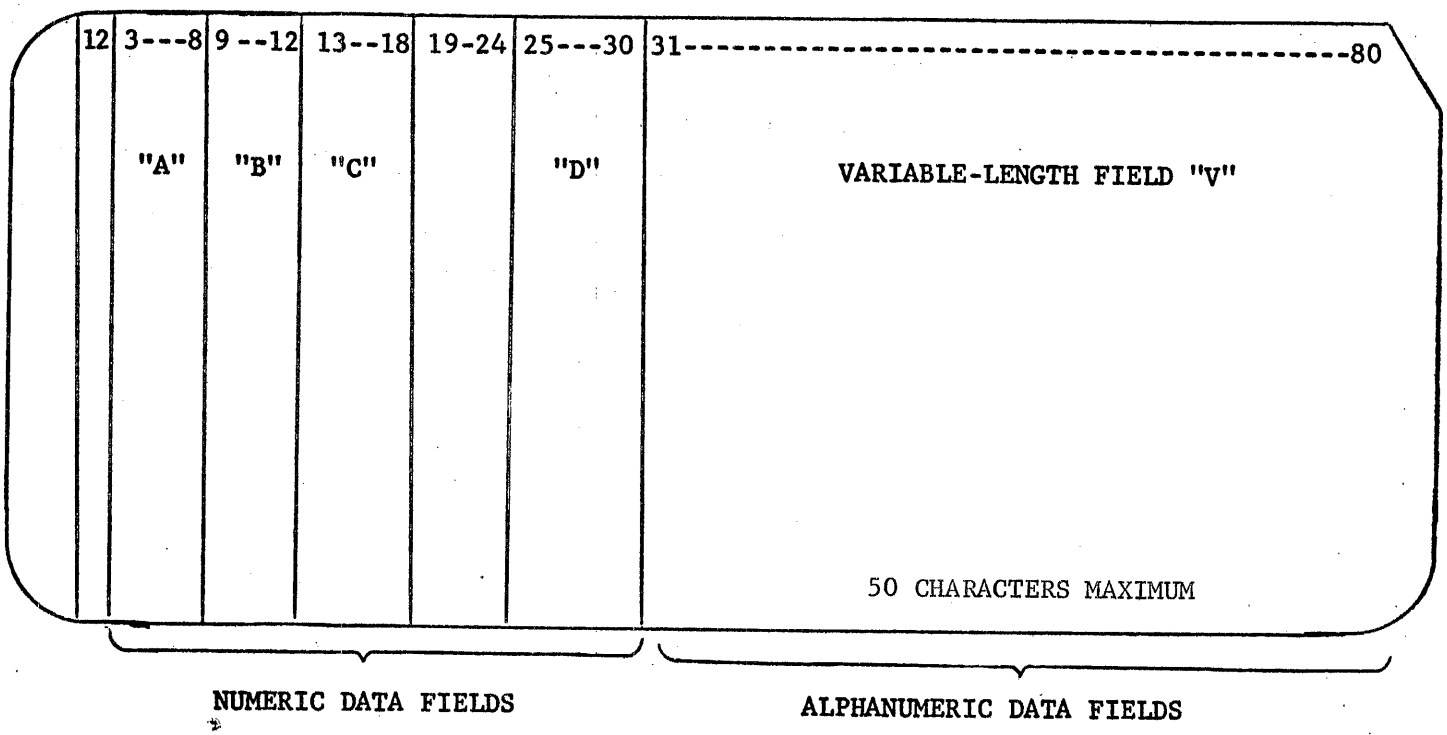


ASSIGNMENT NO. 6

Perform all I/O functions
using File and Record Control (GEFRC).

INPUT ITEM DESCRIPTION

Input Card Format



Description

1. Input item is in BCD.
2. Numeric data fields contain positive integers; field "C" does not contain zero.
3. Each numeric data field is right-justified within its own area. It does not contain leading zeroes.
4. Variable-length field "V" contains some type of alphanumeric descriptive material. If present, field "V" begins with Column 31.

GMAP COURSE

CLASS PROBLEM

Printed Report Format and Description

The printed report will consist of the following:

1. Heading line - Any desired information

2. Sub-heading line - Any desired information

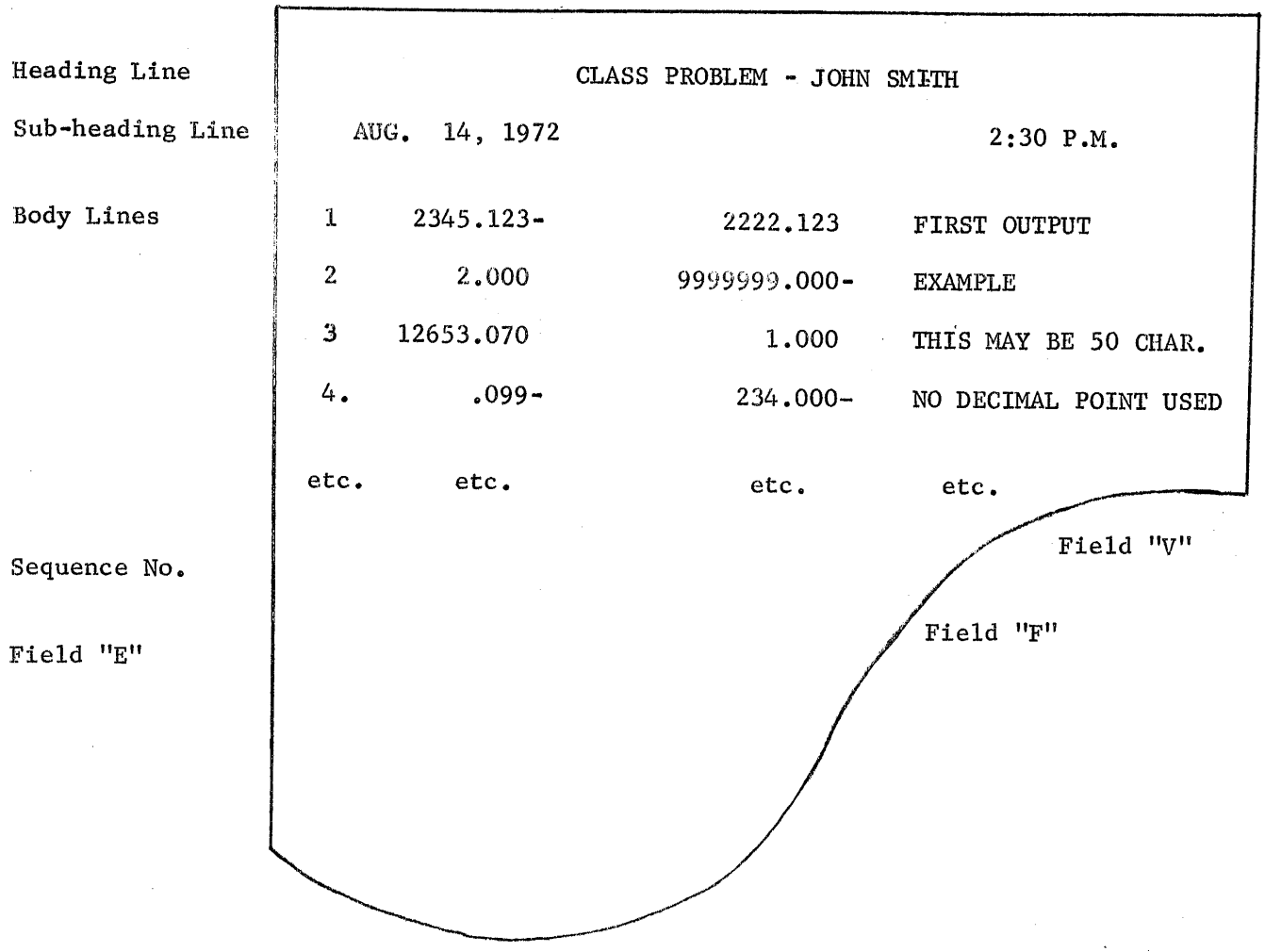
3. Body lines - The body line should be double-spaced. The print positions for each of the fields in the body line is as follows:

Char. position	Field
1 - 9	Blank
10 - 12	Seq. No.
13 - 18	Blank
19 - 30	EBCD
31 - 36	Blank
37 - 48	FBCD
49 - 54	Blank
55 ⟶	Variable Field

GMAP COURSE

CLASS PROBLEM

SAMPLE PRINTED REPORT



OPTIONAL PROBLEM NO. 1

SUMBIT CLASS PROBLEM

1. BLANK THE PRINT AREA USING INDEXING.

2. USING FILE AND RECORD CONTROL (GFRC)

   a. OPEN THE FILES
   b. USE CALL GET AND MOVE THE RECORD TO WORKING STORAGE.
   c. EOF ENCOUNTERED CLOSE FILES.
   d. USE MME GEFINI.

3. CHECK THE FIRST WORD OF WORKING STORAGE FOR CODE SUMBIT.

4. CODE NOT EQUAL TO SUMBIT.

   a. MOVE THE RECORD TO THE PRINT AREA.
      (USE THE DOUBLE REPEAT FOR THE MOVE)
   b. MOVE AN ERROR MESSAGE TO THE PRINT AREA.
   c. USE CALL PRINT TO PRINT THE RECORD.

5. CODE EQUAL TO SUMBIT.

   a. COUNT THE BITS SET TO ONE IN WORDS 3 AND 4.
   b. MOVE THE RECORD TO THE PRINT AREA (USING TALLYS).
   c. CONVERT THE SUM OF BITS FROM WORDS 3 AND 4 TO BCD.
   d. MOVE BCD SUM TO PRINT AREA.
   e. PRINT THE VALID RECORD WITH THE SUM USING CALL PRINT.

OPTIONAL PROBLEM NO. 2

Given values A, B, and C (all values 1)

IF A = B = C then X = A + B + C

IF A = B or B = C or A = C then add the two equal values and multiply by the third = Y

IF A ≠ B ≠ C then divide largest by smallest and subtract middle = Z.

GMAP COURSE

WORKSHOP PERFORMANCE CHART

NAME: _____

CLASS NO.: _____

WORKSHOP PROBLEM

	CODED	DEBUGGED
TOBIN		
CALC		
TOBCD		
EDIT		
INIT		
INPUT		
OUTPUT		
WRAPUP		
OPTIONAL PROBLEM NO. 2		
OPTIONAL PROBLEM NO. 3		

EXERCISES:

#1 _____
#2 _____
#3 _____
#4 _____
#5 _____
#6 _____
#7 _____
#8 _____
#9 _____
#10 _____

MACRO	_____	BOOL	_____	SYMDEF	_____
INE/IFE/IFL/IFG	_____	OCT	_____	SYMREF	_____
IDRP	_____	VFD	_____	EQU	_____
RPT	_____	DEC	_____	BLOCK	_____
PMC	_____	BCI	_____	USE	_____
TTL/TTLS	_____	ARG	_____	SET	_____

SERIES 6000

GMAP PROGRAMMING

COURSE 605

QUIZZES AND FINAL EXAM

OVERVIEW AND DATA QUIZ

1. Name the three component modules of the Series 6000.

   _____, _____, _____.

2. Name six peripheral devices available with the Series 6000.

   _____, _____, _____, _____,

   _____, _____.

3. Name the two processor modes: _____ and

   _____.

4. Name five processor registers excluding index registers.

   _____, _____, _____,

   _____, _____.

5. Maximum memory size on the Series 6000 is _____ K.

6. Fill in missing information as it will appear on the listing when the following pseudo operations are assembled by GMAP.

LOCATION	INSTRUCTION		CARD	IMAGE
000600	_____	DATA	ØCT	2,-5,7746,-2
_____	_____			
_____	_____			
_____	_____			
_____	_____			
_____	_____			
_____	_____		BCI	2, SEPTEMBER
_____	_____			
_____	_____		VFD	18/DATA,H12/CR
_____	_____		VFD	H36/ASEA
_____	_____		VFD	Ø24/74,6/2,6/-1,
_____	_____		ETC	36/40,2/,4/-6
_____	_____			
_____	_____		DEC	-4,1,+3,2B27
_____	_____			
_____	_____			
_____	_____			
_____	_____		DEC	1B71,-8B63,100B68
_____	_____			
_____	_____			
_____	_____			
_____	_____			
_____	_____			
_____	_____			
_____	_____			

INSTRUCTION COUNTER

Fill in missing information is it would appear after the execution of each instruction in the order of execution.

OCTAL LOCATION		INSTRUCTION		MEMORY OR REGISTER OCTAL CONTENTS AFTER EXECUTION
000500	START	LDX5	5,DU	X5=_____
		LDX7	-7,DU	X7=_____
		LDX2	-7,DL	X2=_____
		LDA	2,DU	A=_____
000601	BETA	TRA	ALPHA	IC=_____
000602		STC1	SAVE1	SAVE1=_____
000603		TSX3	TEST	X3=_____
000604		STC2	SAVE2	SAVE2=_____
000605		TRA	J	IC=_____
000606		TRA	*+1	IC=_____
000700	ALPHA	LDI	RGR	IR=_____
000701		STI	SAVE1	SAVE1=_____
000702		LDX2	RGR	X2=_____
000703		TRA	BETA+1	IC=_____
000704	RGR	OCT	000200404000	
000705	SAVE1	ZERO		
000706	TEST			
000707				
000710		TRA	0,3	IC=_____
000711	SAVE2	ZERO	2,5	
000712	J	RET	SAVE2	IC=_____

TALLY OPERATIONS

1. Sixty characters are stored in ten words in memory beginning at location DATA. Store the characters, one character to a word, in right justified form with leading zeros in a sixty-word table beginning at location SPRED.

2. You are given a table of 100 one-word values stored in memory beginning at MAYBZ. Move all nonzero values to a table starting at NOZER. Put the number of nonzero values you have found into memory cell NOZER-1.

3. Write the octal representation of the value resulting from each of the following pseudo operations:

   a. TALLY 1024,48

   b. TALLYD 384,40,4

   c. TALLY 2048,18,3

MISCELLANEOUS OPERATIONS

1. TRANS contains a one digit BCD code in right justified form. Write the coding which will test the code for a range of 0 thru 9. If not 0 thru 9, branch to ERROR, If it is one of the digits 0 thru 9, branch to UPDATE.

2. Location BRANCH contains a TRA 0. If location CODE contains a 1, set the instruction at BRANCH to transfer to RETURN. If CODE contains a 3, set the instruction to transfer to CALC.

3. A one word field called CODE normally contains only positive BCD numeric characters. Check the field for the presence of non-numeric characters. If any are found, branch to ERROR.

4. Set the 800 words of memory beginning at location BUILD to zero.

5. Sum the 100 words stored in TABLE and store the sum in SIGMA.

GENERAL QUIZ

SECTION I

This section of the quiz consists of True-False items, each of which has a value of one (1) point. Enter your answer for each item on this sheet, making sure that you check the correct line for the corresponding test item.

_____ 1. The BLOCK pseudo-op may be used to assemble data and/or instructions into blank common.

_____ 2. The TZE instruction will transfer and turn off the zero indicator if it is on.

_____ 3. The expression (Y-pair) means any two sequential memory locations that begin at an even address.

_____ 4. The contents of an index register may be incremented or decremented.

_____ 5. LDX 3,4 will load X3 with contents of Location 4.

_____ 6. LDX4 3,DL will load X4 with zeros.

SECTION II

This section of the quiz consists of multiple choice items, each of which has a value of two (2) points. Each wrong answer will deduct two points from the total. Enter your answer in the A, B, C, or D column on this sheet, making sure that you check the correct line for the corresponding test item.

NOTE! There is only <u>one</u> (1) correct answer for each item in this section. Answer sheets which contain more than one answer per item will be considered to be <u>wrong!</u>

1. At assembly time--

```
LDA   =MLDX1Ø5,DU
```

_____ A. Will generate an entry in the literal pool.

_____ B. The LDA is an instruction literal.

_____ C. The LDA is a direct operand instruction.

_____ D. Invalid coding.

2. LDA =MTRA⌀LOC,5

The above coding is an example of:

_____ A. Direct upper (DU) address modification.

_____ B. An instruction literal.

_____ C. An instruction literal, but the ",5" is illegal.

_____ D. An illegal operand.

3. Given the following coding example, what will be the contents of the A and Q registers after execution of the LDAQ instruction?

```
000062          LDAQ   TABLE2
000063          TRA    OUT
000064 TABLE1   DEC    5
000065          DEC    6
000066          DEC    7
000067 TABLE2   DEC    1
000070          DEC    2
000071          DEC    3
```

_____ A. A = 6, Q = 7

_____ B. A = 7, Q = 1

_____ C. A = 1, Q = 2

_____ D. A = 2, Q = 3

4. Indicate the coding which will store characters 2, 4, and 5 from the Q register to the same character positions of location X.

_____ A. STCQ X,13

_____ B. STCQ X,26

_____ C. STCQ X,31

_____ D. None of the above.

5. TABLE OCT -43

The above line of coding will cause which of the following octal configurations to be generated?

_____ A. 100000000043

_____ B. 777777777735

_____ C. 777777777743

_____ D. 400000000043

6. LDA 5,DU

In this instruction, the number five is:

_____ A. Found in bits 18 - 35 of the instruction itself.

_____ B. Found in bits 0 - 17 of the instruction itself.

_____ C. Memory location five.

_____ D. Memory location five as modified by the DU directive.

7. LDA ,DU

The above instruction will cause which of the following octal contents to appear in bits 0 - 17 of the instruction?

_____ A. 000023

_____ B. 000000

_____ C. 235003

_____ D. 235007

8. STX2 0,DL

What, if anything, is wrong with this instruction?

_____ A. Nothing is wrong with this coding.

_____ B. The DL should be changed to a DU.

_____ C. The zero operand is illegal in slave mode.

_____ D. Direct type address modification is not allowed with a STX2.

SECTION III

This section of the quiz consists of completion type items, each of which has a value of three (3) points. Each wrong answer will deduct three points from the total.

NOTE! Each answer must be entirely correct to receive credit. No partial credit will be given (1 or 2 points) or items which are "almost" correct.

All answers are brief, and should be entered in the space provided on this sheet.

1. TSX1 30113

Prior to the execution of the above instruction, the IC contains value 1122_8 and the 30113 is an octal number. What will the IC contain (in octal) after execution?

_____

2. Referring to the information and conditions set forth in Item (Question) #1, what will be the octal contents of index register one after execution?

---

3. Show the binary contents of a literal generated by: = 1.5B7

MACRO OPERATIONS

1. Write a MACRO procedure to:

   a. Clear N words

   b. Move N words from a given location to another given location.

   c. Swap N words from two given locations.

2. Write an example for the CALL of each of the above examples.

EFFECTIVE ADDRESS AND GFRC QUIZ

1. Give the effective addresses of the following instructions.

a.		LDA	C1,*
	C1	LDQ	B5,*3
	B5	LDX5	A,1
b.		LDA	*+1,I
		LDQ	*-1,*
c.		LDX1	B,1*
	B+C(X1)	STX1	A,*1
	A	ADX1	F,*5
	F	SBX1	10,DU
	10+C(X_5)	ADX1	15,DU

2. Reverse the bits in the upper half of CHAR. That is

   if $CHAR_i$ = 1 turn it off

   if $CHAR_i$ = 0 turn it on

3. Show the coding to step through TAB until you find a zero word using index registers.

4. Define the following:

   a. File Control Block

   b. File Designator Word

   c. Buffer

FILE AND RECORD CONTROL (GFRC) EXAMINATION

The following statements are either true or false. Write in the blank on the left a T for true or F for false.

_____ 1. A file must be opened before it can be read from or written on.

_____ 2. The user may open all his files with the use of one OPEN command.

_____ 3. If the user tires to OPEN a file that is already open, he will be aborted.

_____ 4. If the OPEN routine determines that a file has been assigned to SYSOUT, but the file control block for the file does not indicate standard systems format, the user will be aborted.

_____ 5. After executing a GET, the current record index (word 0 of the file control block) contains the location of the first data word of the requested logical record.

_____ 6. When executing a GET, the EOF exit will be taken any time that any file mark is found.

_____ 7. The user can obtain the previous logical record processed by using the GETBK command.

_____ 8. The PUT command must be used any time the user desires to move a logical record from the input buffer to working storage.

_____ 9. The PUTBK command can be used if the programmer desires to put his logical record at the beginning of the next buffer, regardless whether or not the current buffer is full.

_____ 10. CALL PRINT always requires a CALL IOEDIT to have been previously executed.

_____ 11. Every time a CALL GET or CALL PUT is executed, the physical device (tape, for example) will be moved.

_____ 12. All the information in a file control block is stored there by the user via the FILCB macro.

_____ 13. File and Record Control (GFRC) subroutines are loaded into memory by General Loader (GLOAD).

_____ 14. File and Record Control (GFRC) does not use GCOS.

_____ 15. The subset of File and Record Control (GFRC) used by the user resides in a special area within.

_____ 16. Standard magnetic tape labels are 14 words long.

_____ 17. Random disc/drum files must be processed using logical record processing commands.

_____ 18. In the standard system format a block serial number will exist as the first word of each block.

_____ 19. When output is to be stacked on SYSOUT, the user must allow for two extra words in the buffer(s).

_____ 20. The only type of records File and Record Control (GFRC) can handle are variable length records.

_____ 21. One file control block must be provided for each file used in the program.

_____ 22. Mixed density is not allowed within a magnetic tape file processed by File and Record Control (GFRC).

_____ 23. The OPEN and CLOSE routines obtain their orders from file designator words.

_____ 24. When buffers are used in conjunction with File and Record Control (GFRC) logical record processing commands, each buffer must be one word larger than the largest physical record it is to hold.

_____ 25. When using the command READ, the user must provide his own list of control words.

To the left of each function listed in column "A" write the number of the item listed in column "B" that performs this function.

	"A"	"B"
_____	Primary communication between the user program and File and Record Control (GFRC) subroutines	1. GETBK 2. One Word greater than physical record 3. Standard Labels 4. File Control Blocks 5. CALL 6. GET 7. PUT 8. PUTBK 9. COPY 10. READ 11. File designator word 12. File Control Cards 13. FILCB 14. OCT 15. OPEN 16. VFD 17. RDREC
_____	Provides information to the OPEN and CLOSE routines	
_____	Whenever the logical record blocking/ deblocking facilities of File and Record Control (GFRC) are used, the input/output buffers must meet this requirement	
_____	Used by GCOS to assign physical devices to the various files used by the program	
_____	System macro-instruction used to define file control blocks	
_____	Pseudo-operation commonly used to define file designator words	
_____	Used to initialize files so that they may be properly accessed by other File and Record Control (GFRC) functions	
_____	Used to obtain the next logical input record from a designated input file	
_____	Used to move a logical record from working storage into an output buffer	
_____	To initiate reading of a physical record with the user providing the list of control words (DCW's)	

The following statements are followed by a series of choices, only one of which is correct. Read each item carefully and mark the correct answer.

1. The command used to initialize the edit functions, such as those used when printing or punching is:

   _____ OPEN
   _____ IOEDIT
   _____ PRINT
   _____ PUNCH

2. The File and Record Control (GFRC) command with which the user can either allocate space within a buffer for a logical record and, if desired, move that logical record into the buffer is:

   _____ GET
   _____ PUT
   _____ GETBK
   _____ PUTSZ

3. The pseudo-operation used in conjunction with all File and Record Control (GFRC) commands is:

   _____ ERLK
   _____ IDRP
   _____ SAVE
   _____ CALL

4. The File and Record Control (GFRC) command used to disconnect files on which no further activity is to be performed is:

   _____ CLOSE
   _____ PUTBK
   _____ RELSE
   _____ SETOUT

5. The File and Record Control (GFRC) command used to set a currently open file to an output file is:

   _____ CLOSE
   _____ PUT
   _____ SETIN
   _____ PUTBK
   _____ SETOUT

6. The File and Record Control (GFRC) command that may be used to obtain the next logical input record from a file is:

   _____ GET
   _____ PUT
   _____ CLOSE
   _____ RELSE

7. For disk and drum two types of files are provided:

_____ Variable- and fixed-length
_____ File Control Block and File Designator Word
_____ Linked and Random
_____ Letter and Nail

8. In the standard system format data blocks (physical records) may be variable in length up to a maximum of:

_____ 320 bits
_____ 321 words
_____ 320 characters
_____ 320 words

9. If in standard system format only one of the five media named below may use header and trailer labels. Indicate which.

_____ Card Reader
_____ Magnetic Tape
_____ Paper Tape
_____ Disk/Drum
_____ Card Punch

10. The maximum block (physical record) size allowed with File and Record Control (GFRC) is:

_____ 320 words
_____ 1024 words
_____ 4095 characters
_____ 4095 words

1. Define the following terms:

   a. Physical Record:

   b. Logical Record:

   c. File Control Block:

   d. File Designator Word:

   e. Buffer:

   f. File:

2. What File and Record Control (GFRC) commands obtain information from the file designator words?

3. Why are there report codes and media codes in SYSOUT records?

4. Why must buffers that are used with logical records processing be one word larger than the largest physical record?

5. What are the types of records that can be handled by File and Record Control GFRC?

6. What are the characteristics of Standard System Format?

7. List the significant capabilities that are provided when Standard System Format is used.

8. What features does File and Record Control (GFRC) provide the user?

SECTION I

GMAP QUIZ 9

This section of the quiz consists of True-False items, each of which has a value of one (1) point. Enter your answer for each item on this sheet, making sure that you check the correct line for the corresponding test item.

_____ 1. The following coding will convert one binary word to seven (7) BCD characters.

```
LDX1    0,DU
LDA     X       (X is some binary number)
RPT     6,1
BCD     PUB,1   (PUB is the start of the
                 conversion table)
STQ     DATA
BCD     0,1
QLS     30
STQ     DATA+1
```

_____ 2. The following is a valid literal expression: =12H JUMP HOME.

_____ 3. STCA LOC,40 will affect bit positions 18-23.

_____ 4. STCA LOC,04 will affect bit positions 18-23.

_____ 5. HERE TRA **,5 will transfer to location HERE plus the value in X5.

_____ 6. LDX4 =1B18,DU will turn on bit zero in X4.

SECTION II

This section of the quiz consists of multiple choice items, each of which has a value of two (2) points. Each wrong answer will deduct two points from the total. Enter your answer in the A, B, C, or D column on this sheet, making sure that you check the correct line for the corresponding test item.

NOTE! There is only one (1) correct answer for each item in this section. Answer sheets which contain more than one answer per item will be considered to be wrong!

1. What is meant by the expression =1B17?

_____ A. A one in bit position seventeen.

_____ B. A type of airplane.

_____ C. A one followed by seventeen zero bits.

_____ C. A one followed by seventeen zeros.

2. In a floating-point number in memory the binary point of the mantissa is:

_____ A. Following bit 0

_____ B. Following bit 7

_____ C. Following bit 8

_____ D. Following bit 35

3. LDA =.5B3,DU

The above line of coding will cause which of the following octal configurations to be generated?

_____ A. 500000235003

_____ B. 400000235003

_____ C. 200000235003

_____ D. 000000235003

4. We wish to move fifty words from table A to table B. The two tables were established as follows:

```
A   EBSS   50
B   EBSS   50
```

Which of the following coding examples (A or B) will do the job?

_____ A.
```
LDX3   A,DU
LDX4   B,DU
RPD    25,2
LDAQ   0,3
STAQ   0,4
```

_____ B.
```
LDX7   50,DU
LDAQ   A-50,7
STAQ   B-50,7
SBX7   2,DU
TNZ    *-3
```

5. =32.7B7

This literal represents a/an:

_____ A. Floating point number

_____ B. Fixed point number

_____ C. Integer

_____ D. Octal literal

6. When using the DEC pseudo-op, what over-rides all other designations in determining whether the constant will be fixed or floating?

_____ A. The letter B

_____ B. The letter D

_____ C. The letter E

_____ D. A decimal point

7. A floating point number in memory has the following octal contents: 000260000000_8. How many shift operations will be required to normalize this value?

_____ A. 9

_____ B. 6

_____ C. 3

_____ D. 1

SECTION III

This section of the quiz consists of completion type items, each of which has a value of three (3) points. Each wrong answer will deduct three points from the total.

NOTE! Each answer must be entirely correct to receive credit. No partial credit will be given (1 or 2 points) or items which are "almost correct.

All answers are brief, and should be entered in the space provided on the following sheet.

1. How many times will the instruction at X be executed?

```
        LDX2    0,DU
        RPT     14,1,TMI
X       LDA     Z,2
        ---
        ---
        ---
Z       DEC     0,-1,-2,-3,-4,-5
```

2. The number 013461000000_8 is in core at location X. Show the octal contents of the E and A registers after execution of a FLD X.

3. Is the mantissa of the number in Item (Question) #2 positive or negative?

SECTION I

GMAP QUIZ 10

This section of the quiz consists of True-False items, each of which has a value of one (1) point. Enter your answer for each item on this sheet, making sure that you check the correct line for the corresponding test item.

_____ 1. If the letter Ø is coded in Column 7 of a source deck card, a NØP instruction may be inserted into the program by the assembler.

_____ 2. For the following coding the effective transfer address depends upon the contents of Index Register four (X4).

```
        TRA     TALLYX,I
        ---     --------
        ---     --------
TALLYX  TALLYC  ADDRS,12,*4
```

_____ 3. The tally run-out indicator is not affected by using IT type modification.

_____ 4. The instruction: LDAQ Y will not execute if Y is an odd address.

_____ 5. STBA LOC,40 will affect bit positions 0-8.

_____ 6. STBA LOC,02 will affect bit positions 24-29.

SECTION II

This section of the quiz consists of multiple choice items, each of which has a value of two (2) points. Each wrong answer will deduct two points from the total. Enter your answer in the A, B, C, or D column on this sheet, making sure that you check the correct line for the corresponding test item.

NOTE! There is only <u>one</u> (1) correct answer for each item in this section. Answer sheets which contain more than one answer per item will be considered to be <u>wrong</u>!

1. The correct coding at the end of subroutine DØIT which will return us back to the normal return is:

_____ A. TRA BETA,1

_____ B. TRA 2,1

_____ C. TRA BETA+1

_____ D. TRA TSX1+3

```
TSX1    DOIT
ARG     ALPHA
ARG     BETA
STA     BOOL
```

2.
```
GOOD    LDA     ALPHA,*N
ALPHA   LDQ     BETA,*5
BETA    CMPA    STAR,DU
```

In the above program, the effective address of the LDA instruction is:

_____ A. ALPHA

_____ B. STAR+C(X5)

_____ C. BETA+C(X5)

_____ D. ALPHA+C(X5)

3. What is the effective address of the following coding sequence?

```
        LDA     W,*
W       ---     X,6*
X+C(X6) ---     Y,7*
Y+C(X7) ---     Z
```

_____ A. W

_____ B. X+C(X6)

_____ C. Y+C(X7)

_____ D. Z

4. The instruction 000026735011_8 is contained in memory location BETA. Its representation on a symbolic coding form would be:

_____ A. BETA ALS 22,1

_____ B. BETA ALS 26,1

_____ C. BETA ALR 22,1*

_____ D. BETA ALR 26,1*

5. What is the maximum number of levels of indirection permitted?

_____ A. Less than 63

_____ B. 63

_____ C. 64

_____ D. Time limit consideration only.

6. LDQ PLACE,*

The above coding example employs which type of address modification?

_____ A. IR

_____ B. RI

_____ C. IT

_____ D. CI

SECTION III

This section of the quiz consists of completion type items, each of which has a value of three (3) points. Each wrong answer will deduct three points from the total.

NOTE! Each answer must be entirely correct to receive credit. No partial credit will be given (1) or 2 points) or items which are "almost" correct.

All answers are brief, and should be entered in the space provided on this sheet.

1. LDX3 0,1*

Prior to the execution of the above instruction, the contents of the various registers and memory cells are as follows:

(ALL ARE IN OCTAL)

	X1	000177
	X3	002207
	177	001406000000
contents	307	001114000000
of core	1406	000307000000
locations	1760	000014000000
	2207	001760000000

What will be the octal contents of index register three after execution?

2. Referring to the information and conditions set forth in Item (Question) #1, what will be the octal contents of index register one be after execution?

3.
```
DUMMY   MACRO
K       SET     0
        IDRP    #2
K       SET     K+1
        IDRP
        TRA     *+K
        ENDM    DUMMY
```

If we use the above MACRO with this calling sequence:

```
        DUMMY   ABLE,(2,3,4,5,6,7),BAKER
```

What would be the actual value of K when the transfer instruction is built by the assembler?

GMAP EXAM

SECTION ONE

This section of the exam consists of twenty True-False items, each of which has a value of one (1) point. Each wrong answer will deduct one point from the total of twenty possible points for this section. Enter your answer for each item on the answer sheet provided, making sure that you check the correct line for the corresponding test item.

1. The slave prefix area can be accessed by a program in the slave mode.

2. The slave service area can be accessed by a program in the slave mode.

3. The BLOCK pseudo-op may be used to assemble data and/or instructions into blank common.

4. The TZE instruction will transfer and turn off the zero indicator if it is on.

5. The following coding will convert one binary word to seven (7) BCD characters.

```
        LDX1    0,DU
        LDA     X               (X is some binary number)
        RPT     6,1
        BCD     PUB,1       }   (PUB is the start of the
        STQ     DATA        }    conversion table)
        BCD     PUB,1
        QLS     30
        STQ     DATA+1
```

6. The following is a valid literal expression: =12H JUMP HOME.

7. If the letter Ø is coded in Column 7 of a source deck card, a NØP instruction may be inserted into the program by the assembler.

8. For the following coding the effective transfer address depends upon the contents of Index Register four (X4).

```
        TRA     TALLYX,I
        ---     --------
        ---     --------
TALLYX  TALLYC  ADDRS,12,*4
```

9. The tally run-out indicator is not affected by using IT type modification.

10. The instruction: LDAQ Y will not execute if Y is an odd address.

11. The expression (Y-pair) means any two sequential memory locations that begin at an even address.

12. The contents of an index register may be incremented or decremented.

13. STCA LOC,40 will affect bit positions 18-23.

14. STCA LOC,04 will affect bit positions 18-23.

15. STBA LOC,40 will affect bit positions 0-8.

16. STBA LOC,02 will affect bit positions 24-29.

17. HERE TRA **,5 will transfer to location HERE plus the value in X5.

18. LDX 3,4 will load X3 with bits 0... 17 of location 4.

19. LDX4 3,DL will load X4 with zeros.

20. LDX4 =1B18,DU will put a bit in X4.

Go on to SECTION TWO of the test at this time.

GMAP EXAM

SECTION TWO

This section of the exam consists of twenty-five multiple choice items, each of which has a value of two (2) points. Each wrong answer will deduct two points from the total of fifty possible points for this section. Enter your answer in the A, B, C, or D column on the answer sheet, making sure that you check the correct line for the corresponding test item.

NOTE! There is only one (1) correct answer for each item in this section. Answer sheets which contain more than one answer per item will be considered to be wrong!

21. Which of the following coding examples represents IR type modification?

A. ADR,N*
B. ADR,2*
C. ADR,*
D. ADR,*N

22. What is meant by the expression =1B17?

A. a one in bit position seventeen.
B. a type of airplane.
C. a one followed by seventeen zero bits.
D. a one followed by seventeen zeros.

23. Given the following MACRO prototype:

```
MAC     MACRO
        LDA     #1
        ADA     #2
        STA     #3
        ENDM    MAC
```

Assuming that all symbols have been defined, which of the following examples contains improper coding to use the above MACRO?

```
A.      PLEASE  MAC     =20DL,=30DL,CAT
B.      GO      MAC     (ABLE,4),CAT,DOG
C.      GET     MAC     GOOD,BAD,INDIFF
D.      HELP    MAC     (,4),=45,UNABLE
```

24. At assembly time--

```
        LDA     =MLDX1Ø5,DU
```

A. will generate an entry in the literal pool.
B. The LDA is an instruction literal.
C. The LDA is a direct operand instruction.
D. Invalid coding.

25. In a floating-point number in memory the binary point of the mantissa is:

A. following bit 0
B. following bit 7
C. following bit 8
D. following bit 35

26. The following coding is used to go to a subroutine called "DØIT" with the indicated normal return:

```
TSX1   DØIT
ARG    ALPHA
ARG    BETA
STA    BØØL        (this is the normal return)
```

The correct coding at the end of subroutine DØIT which will return us back to the normal return is:

A. TRA BETA,1
B. TRA 2,1
C. TRA BETA+1
D. TRA TSX1+3

27. LDA =MTRAØLOC,5

The above coding is an example of:

A. Direct upper (DU) address modification.
B. An instruction literal.
C. An instruction literal, but the ",5" is illegal.
D. An illegal operand.

28. The decimal value of the following floating point number contained in a core dump is:
770700000000_8

A. 5.4687
B. 0.54687
C. 0.054687
D. 0.0054687

29. LDA =.5B3,DU

The above line of coding will cause which of the following octal configurations to be generated?

A. 500000235003
B. 400000235003
C. 200000235003
D. 000000235003

30. In a floating-point literal, why would you use a D instead of an E or a decimal point?

A. To indicate that this is a decimal literal.
B. To cause a literal table dump if the program aborts.
C. To obtain a double-precision number.
D. To assemble the literal directly into the operand of the instruction.

31.
```
GOOD    LDA     ALPHA,*N
ALPHA   LDQ     BETA,*5
BETA    CMPA    STAR,DU
```

In the above program, the effective address of the LDA instruction is:

A. ALPHA
B. STAR+C(X5)
C. BETA+C(X5)
D. ALPHA+C(X5)

32. What is the effective address of the following coding sequence?

```
        LDA     W,*
W       ---     X,6*
X+C(X6)---      Y,7*
Y+C(X7)---      Z
```

A. W
B. X+C(X6)
C. Y+C(X7)
D. Z

33. The instruction 000026735011_8 is contained in memory location BETA. Its representation on a symbolic coding form would be:

A. BETA ALS 22,1
B. BETA ALS 26,1
C. BETA ALR 22,1*
D. BETA ALR 26,1*

34. Given the following coding example, what will be the contents of the A and Q registers after execution of the LDAQ instruction?

```
000062                      LDAQ    TABLE2
000063                      TRA     OUT
000064              TABLE1  DEC     5
000065                      DEC     6
000066                      DEC     7
000067              TABLE2  DEC     1
000070                      DEC     2
000071                      DEC     3
```

A. A = 6, Q = 7
B. A = 7, Q = 1
C. A = 1, Q = 2
D. A = 2, Q = 3

35. Indicate the coding which will store characters 2, 4, and 5 from the Q register to the same character positions of location X.

A. STCQ X,13
B. STCQ X,26
C. STCQ X,31
D. None of the above.

36. We wish to move fifty words from table A to table B. The two tables were established as follows:

```
A     EBSS    50
B     EBSS    50
```

Which of the following coding examples (A or B) will do the job?

A.
```
LDX3    A,DU
LDX4    B,DU
RPD     25,2
LDAQ    0,3
STAQ    0,4
```

B.
```
LDX7    50,DU
LDAQ    A-50,7
STAQ    B-50,7
SBX7    2,DU
TNZ     *-3
```

37. `TABLE  OCT  -43`

The above line of coding will cause which of the following octal configurations to be generated?

A. 100000000043
B. 777777777735
C. 777777777743
D. 400000000043

38. `LDA  5,DU`

In this instruction, the number five is:

A. found in bits 18 - 35 of the instruction itself.
B. found in bits 0 - 17 of the instruction itself.
C. memory location five.
D. memory location five as modified by the DU directive.

39. `LDA  ,DU`

The above instruction will cause which of the following octal contents to appear in bits 0 - 17 of the instruction?

A. 000023
B. 000000
C. 235003
D. 235007

40. `STX2  0,DL`

What, if anything, is wrong with this instruction?

A. Nothing is wrong with this coding.
B. The DL should be changed to a DU.
C. The zero operand is illegal in slave mode.
D. Direct type address modification is not allowed with a STX2.

41. What is the maximum number of levels of indirection permitted?

A. Less than 63
B. 63
C. 64
D. Time limit consideration only.

42. =32.7B7

This literal represents a/an:

A. floating point number
B. fixed point number
C. integer
D. octal literal

43. When using the DEC pseudo-op, what over-rides all other designations in determining whether the constant will be fixed or floating?

A. The letter B
B. The letter D
C. The letter E
D. A decimal point

44. A floating point number in memory has the following octal contents: 000260000000_8 How many shift operations will be required to normalize this value?

A. 9
B. 6
C. 3
D. 1

45. LDQ PLACE,*

The above coding example employs which type of address modification?

A. IR
B. RI
C. IT
D. CI

Go on to SECTION THREE of the test at this time

SECTION THREE

This section of the exam consists of ten completion type items, each of which has a value of three (3) points. Each wrong answer will deduct three points from the total of thirty possible points for this section.

NOTE! Each answer must be entirely correct to receive credit. No partial credit will be given (1 or 2 points) for items which are "almost" correct.

All answers are brief, and should be entered in the space provided on the answer sheet.

46. TSX1 30113

Prior to the execution of the above instruction, the IC contains the value 1122_8 and the 30113 is an octal number. What will the IC contain (in octal) after execution?

47. Referring to the information and conditions set forth in Item (Question) #46, what will be the octal contents of index register one after execution?

48. How many times will the instruction at X be executed?

```
        LDX2    0,DU
        RPT     14,1,TMI
X       LDA     Z,2
        ---
        ---
        ---
Z       DEC     0,-1,-2,-3,-4,-5
```

49. The number 013461000000_8 is in core at location X. Show the octal contents of the E and A registers after execution of a FLD X.

50. Is the mantissa of the number in Item (Question) #49 positive or negative?

51. LDX3 0,1*

Prior to the execution of the above instruction, the contents of the various registers and memory cells are as follows:

(ALL ARE IN OCTAL)

	X1	000177
	X3	002207
contents of core locations	177	001406000000
	307	001114000000
	1406	000307000000
	1760	000014000000
	2207	001760000000

What will be the octal contents of index register three after execution?

52. Referring to the information and conditions set forth in Item (Question) #51, what will be the octal contents of index register one be after execution?

53.
```
DUMMY MACRO
K     SET     0
      IDRP    #2
K     SET     K+1
      IDRP
      TRA     *+K
      ENDM    DUMMY
```

If we use the above MACRO with this calling sequence:

```
DUMMY   ABLE,(2,3,4,5,6,7),BAKER
```

What would be the actual value of K when the transfer instruction is built by the assembler:

54. Show the binary contents of a literal generated by: =1.5B7

55. Show the octal equivalent of the number generated in Item (Question) #54.

BEFORE TURNING IN YOUR ANSWER SHEET:

1. Be sure that your name and class number appear on the sheet.

2. Be sure that you have not forgotten to answer any items on the exam.

Please retain this exam question sheet for use during the review.

There are a total of 100 points available on this exam. It should be mentioned that your performance in this course, however, is not determined solely from the numerical grade achieved on this exam. The grade is used, along with several other factors, to contribute to a detailed personal evaluation of your work in this course.

TEST
VERSION_____

GMAP EXAM ANSWER SHEET

_____
Student's Name

_____
Student's Class Number

SECTION ONE

ITEM	TRUE	FALSE
1		
2		
#		
4		
5		
6		
7		
8		
9		
10		
11		
12		
13		
14		
15		
16		
17		
18		
19		
20		

SECTION TWO

ITEM	A	B	C	D
21				
22				
23				
24				
25				
26				
27				
28				
29				
30				
31				
32				
33				
34				
35				
36				
37				
38				
39				
40				
41				
42				
43				
44				
45				

SECTION THREE

ITEM	ANSWER
46	_____ IC
47	_____ X1
48	_____
49	_____ E _____ A
50	_____
51	_____ X3
52	_____ X1
53	_____
54	_____ (binary)
55	_____ (octal)

The Other Computer Company:

Honeywell

HONEYWELL INFORMATION SYSTEMS

6034
4C1272
Printed in U.S.A.

In the U.S.A.: 200 Smith Street, MS 061, Waltham, Massachusetts 02154
In Canada: 2025 Sheppard Avenue East, Willowdale, Ontario

AF52, Rev. 1