The 141 Data Processing System - An Educational Computer
for Instruction in Basic Programming 13.0.017
(DISK)

DISCLAIMER

Although each program has been tested by its contributor, no warranty, express or implied, is made by the contributor or 1620 USERS Group, as to the accuracy and functioning of the program and related program material, nor shall the fact of distribution constitute any such warranty, and no responsibility is assumed by the contributor or 1620 USERS Group, in connection therewith.

1620 USERS GROUP PROGRAM REVIEW AND EVALUATION

(fill out in typewriter or pencil, do not use ink)

Program No. _____ Date _____

Program Name: _____

1. Does the abstract adequately describe what the program is and what it does? Yes___ No___
   Comment _____

2. Does the program do what the abstract says? Yes___ No___
   Comment _____

3. Is the Description clear, understandable, and adequate? Yes___ No___
   Comment _____

4. Are the Operating Instructions understandable and in sufficient detail? Yes___ No___
   Comment _____
   Are the Sense Switch options adequately described (if applicable)? Yes___ No___
   Are the mnemonic labels identified or sufficiently understandable? Yes___ No___
   Comment _____

5. Does the source program compile satisfactorily (if applicable)? Yes___ No___
   Comment _____

6. Does the object program run satisfactorily? Yes___ No___
   Comment _____

7. Number of test cases run_____. Are any restrictions as to data, size, range, etc. covered adequately in description? Yes___ No___
   Comment _____

8. Does the Program Meet the minimal standards of the 1620 Users Group? Yes___ No___
   Comment _____

9. Were all necessary parts of the program received? Yes___ No___
   Comment _____

10. Please list on the back any suggestions to improve the usefulness of the program. These will be passed onto the author for his consideration.

Please return to:

Mr. Richard L. Pratt
Data Corporation
7500 Old Xenia Pike
Dayton, Ohio 45432

Your Name _____

Company _____

Address _____

User Group Code _____

THIS REVIEW FORM IS PART OF THE 1620 USER GROUP ORGANIZATION'S PROGRAM REVIEW AND EVALUATION PROCEDURE. NONMEMBERS ARE CORDIALLY INVITED TO PARTICIPATE IN THIS EVALUATION.

11/09/64

- THE 141 DATA PROCESSING SYSTEM -

AN EDUCATIONAL COMPUTER FOR

INSTRUCTION IN BASIC PROGRAMMING

By

Kenneth P. Swallow
College of San Mateo

and

Richard E. Gentry

Submitted by

College of San Mateo
1700 West Hillsdale Blvd.
San Mateo, California
Telephone: 341 - 6161

Users Group # 5194 November 12, 1964

Modifications or revisions to this program, as they occur, will be announced in the appropriate Catalog of Programs for IBM Data Processing Systems. When such an announcement occurs, users should order a complete new program from the Program Information Department.

//

1620 USERS GROUP LIBRARY

PROGRAM ABSTRACT

1. TITLE: The 141 Data Processing System - An Educational Computer for Instruction in Basic Programming.
   SUBJECT CLASSIFICATION: 13.0.

2. AUTHOR; ORGANIZATION: Kenneth P. Swallow, College of San Mateo and Richard E. Gentry,
   USERS GROUP MEMBERSHIP CODE: 5194

3. DIRECT INQUIRIES TO: Kenneth P. Swallow, College of San Mateo, 1700 West Hillsdale Blvd., San Mateo, California.
   Phone: 341 - 6161

4. DESCRIPTION/PURPOSE: The 141 Data Processing System is an educational tool for teaching basic computer concepts through a 1401 type machine language and 1401 SPS. The 141 Simulator set of basic 1401 instructions includes: SW, CW, R, P, W, MCW, LCA, CS, B, C, A, S, H, and NOP. This set is sufficient to illustrate such concepts as: (1) reading, punching, and printing, (2) looping for iterative processes, (3) sequence checking, (4) counted loops, (5) program switches, (6) load routines, (7) address modification, (8) subroutines and subroutine linkages, etc. Basic utility routines are incorporated in the 141 Simulator for easy debugging of 141 programs. 141 written subroutines are provided for multiplication, division, zero suppression, and editing.
   The 141 SPS Assembler processes the 141 set of instructions and all 1401 pseudo operations including: DCW, DC, DS, DSA, ORG, EX, and END.

7. SPECIFICATIONS:
   a. STORAGE USED BY PROGRAM: 20,000 positions of storage.
   b. EQUIPMENT REQUIRED BY PROGRAM:
      Version A - Basic 1620 card system
      Version B - 1620, 1622, and 1443
      Version C - 1620, 1622, 1311, and indirect addressing
      Version D - 1620, 1622, 1443, 1311, and indirect addressing
   c. PROGRAMMING TYPE:
      Versions A and B - SPS - 1620/1710
      Versions C and D - SPS II-D

///

TABLE OF CONTENTS

DECK LABELING

Card Number - Columns 77 - 80

Version A - Basic 1620 -13.0.015

141 SPS Assembler	1001 - 1154
141 Simulator	2001 - 2201

Version B - 1620 with 1443 Printer - 13.0.016

141 SPS Assembler	3001 - 3151
141 Simulator	4001 - 4195

Version C - 1620 with 1311 Disk Storage Drive and Indirect Addressing -13.0.017

141 SPS Assembler	5001 - 5109
141 Simulator	6001 - 6144

Version D - 1620 with 1443 Printer, 1311 Disk Storage Drive, and Indirect Addressing -13.0.018

141 SPS Assembler	7001 - 7108
141 Simulator	8001 - 8139

The program decks for Versions C and D are in 1620 Monitor System Output Format. They can be loaded onto the disk by means of the Disk Utility Program using an *DLOAD card with the program name, 141SPS or 141SIM, in columns 7 - 12 and 02402CM in columns 44 - 50.

PROGRAM WRITEUP

AND

STUDENT MANUAL

FOR

- THE 141 DATA PROCESSING SYSTEM -

AN EDUCATIONAL COMPUTER FOR

INSTRUCTION IN BASIC PROGRAMMING

VI

- THE 141 DATA PROCESSING SYSTEM -

AN EDUCATIONAL COMPUTER FOR

INSTRUCTION IN BASIC PROGRAMMING

By

Kenneth P. Swallow
College of San Mateo

and

Richard E. Gentry

Submitted by

College of San Mateo
1700 West Hillsdale Blvd.
San Mateo, California
Telephone: 341 - 6161

Users Group # 5194

November 12, 1964

THE 141 DATA PROCESSING SYSTEM

CONTENTS

SECTION 1

THE 141 DATA PROCESSING MACHINE

INTRODUCTION

A basic course in the use of computers for data processing can serve many purposes, such as: introducing the terminology that is peculiar to the industry, describing the elements that are common to all computer systems, enumerating the application areas in data processing, exploring the coding systems used with computers, and many others. But undoubtedly the greatest service that a basic computer course can give to the beginner is to provide a firm, even if limited, foundation in stored programming concepts. The degree to which a beginning programmer grasps these basic principles will, to a large extent, determine his success in future programming classes.

The choice of the computer, the programming language, and the exercises to be used in the basic course is an important factor in the ease with which stored program concepts can be imparted to the student. The 141 was designed solely as a vehicle for teaching these initial concepts.

The 141 has sufficient instructions to allow the coding of a wide range of programming problems, but at the same time the instruction set is small enough that the primary effort of the student is directed toward the understanding of programming concepts and not the memorization of a large number of operational rules. The 141 set of fourteen instructions will permit field definition, data movement, input/output operations, comparing, arithmetic operations, and branching. Simple exercises in 141 programming can illustrate such concepts as (1) reading, punching and printing, (2) looping for iterative processes, (3) sequence checking, (4) counted loops, (5) address modification, (6) program switches, (7) subroutine linkages, etc.

If coding a program builds confidence in the new programmer, seeing his program run and seeing it printed as it is represented in core storage can only strengthen that confidence and cement further the whole concept of stored programming in his mind. Any 141 program can be run on an IBM 1401, 1460, or 1410 Data Processing System. It can also be run on an IBM 1620 Data Processing System through the use of a special simulator program.

In addition to the concept of stored programming, a basic computer course should also include the introduction to a symbolic assembly

language. 141 programs can be written in IBM 1401 SPS (Symbolic Programming System) and assembled on a 1401. If instead an IBM 1620 is available, programs written for the 141 in SPS can be assembled on the 1620.

DESCRIPTION

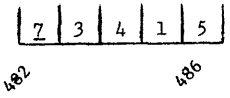
The 141 Data Processing System is an abbreviated version of the IBM 1401 Data Processing System. It is an internally stored program machine with the following features:

1. Input: IBM Card Reader.
2. Output: IBM Card Punch and 100 character per line Printer.
3. Storage: 1000 positions of core storage with three digit numerical addresses.
4. Instuction length and date length: Variable.

Each position is designated by a three digit address in the range of 000 through 999 and is capable of storing one character: a letter of the alphabet, a numeric digit or a special character such as . , / + or ⊟ . A group of consecutive storage positions make up a field. Both data and instruction fields are variable in length so that no storage space need be wasted with meaningless blanks or zeros.

DATA

If a field is used for data it is referred to by the address of its low order, or right most, position. A special indicator called a word mark is placed in the high order, or left most, position to indicate the length of the field. The machine processes data fields from right to left starting at the addressed low order position and continuing until a character with a word mark is met. For example, a five-digit data field in storage locations 482 through 486 would have a word mark (indicated by an underline) in position 482 and would be addressed by 486. Notice that the low order position has the highest address.



INSTRUCTIONS

When a field contains an instruction it is addressed by the high order, or left most, position of the field. This position contains the operation code character and a word mark. In addition to the operation code an instruction may also contain one or two three-digit addresses and/or a modifying character. An instruction may therefore contain one, four, five, seven, or eight characters. The machine reads instructions from left to right starting with the addressed high order position and continuing until it meets the word mark in the high order position of the next instruction.

CHARACTER CODING

Each position of storage is made up of eight magnetic cores, each of which can hold one "yes-or-no" bit. Four bits are needed to represent the digit portion of the Hollerith code of the characters; two more bits are needed to represent the zone portion of the Hollerith code of the characters; another bit is used to indicate the presence or absence of a word mark; and the eighth bit is used for checking.

The digit portion is coded in the Binary Code Decimal (BCD) system in which the four bits have the values of 8, 4, 2, and 1 respectively. The sum of the "yes" bits is equal to the value of the digit. Notice that the digit 0 is represented by the 8 bit and the 2 bit rather than no bits at all.

Digit	BCD Code			
	8	4	2	1
1				X
2			X	
3			X	X
4		X		
5		X		X
6		X	X	
7		X	X	X
8	X			
9	X			X
0	X		X	

In representing alphabetic and special characters the zone is represented by the A and the B bits as follows:

Zone		B	A
12	(A - I)	X	X
11	(J - R)	X	
0	(S - Z)		X
no	(0 - 9)		

The C bit is used for parity checking and is chosen so that the character will always contain an odd number of "yes" bits. For example, the letter "F" which is represented by the 12 and 6 punches on an IBM card is represented in the seven-bit alphameric code as:

Character	Card Code	Seven-bit Alphameric Code C	B	A	8	4	2	1
F	12-6	X	X	X		X	X	

Table 1 gives the seven-bit alphameric code for each Hollerith character used with the 141 Data Processing Machine.

TABLE 1

Character	Card Code	Seven-bit Alphameric Code *C	B	A	8	4	2	1
Blank	Blank	X						
.	12 - 3 - 8		X	X	X		X	X
¤	12 - 4 - 8	X	X	X	X	X		
&	12	X	X	X				
-	11		X					
/	0 - 1	X		X				X
,	0 - 3 - 8	X		X	X		X	X
A	12 - 1		X	X				X
B	12 - 2		X	X			X	
C	12 - 3	X	X	X			X	X
D	12 - 4		X	X		X		
E	12 - 5	X	X	X		X		X
F	12 - 6	X	X	X		X	X	
G	12 - 7		X	X		X	X	X
H	12 - 8		X	X	X			
I	12 - 9	X	X	X	X			X
J	11 - 1	X	X					X
K	11 - 2	X	X				X	
L	11 - 3		X				X	X
M	11 - 4	X	X			X		
N	11 - 5		X			X		X
O	11 - 6		X			X	X	
P	11 - 7	X	X			X	X	X
Q	11 - 8	X	X		X			
R	11 - 9		X		X			X
S	0 - 2	X		X			X	
T	0 - 3			X			X	X
U	0 - 4	X		X		X		
V	0 - 5			X		X		X
W	0 - 6			X		X	X	
X	0 - 7	X		X		X	X	X
Y	0 - 8	X		X	X			
Z	0 - 9			X	X			X
0	0	X			X		X	
1	1							X
2	2						X	
3	3	X					X	X
4	4					X		
5	5	X				X		X
6	6	X				X	X	
7	7					X	X	X
8	8				X			
9	9	X			X			X

* Check bit to produce odd-parity. Table shows values for positions with no word mark. Reverse if word mark is present.

141 INSTRUCTIONS

(IN ALPHABETIC ORDER OF SPS MNEMONICS)

Add — A

Op-Code	A-address	B-address
A	aaa	bbb

Description The Add instruction causes the data in the A-field to be added algebraically to the data in the B-field. The A-field is not disturbed and the resulting sum is stored in the B-field.

Word Marks The defining word mark of the B-field terminates the operation. If the A-field is shorter than the B-field, the A-field word mark will halt transmission of data from the A-field but any resulting carries will be added to the B-field until the B-field word mark is sensed. If a carry results beyond the B-field word mark, it is lost or if the A-field is longer, the high order positions of the A-field which exceed the limits imposed by the B-field word mark are lost. These both represent overflow conditions.

Add (Single Address) — A

Op-Code	A-address
A	aaa

Description The four position Add instruction causes the A-field to be added to itself with the sum replacing the original A-field.

Word Marks The word mark is not affected.

Branch — B

Op-Code	I-address
B	iii

Description The Branch instruction causes the program to branch to the instruction specified by the I-address.

Branch If Indicator On — B

Op-Code	I-address	d-character
B	iii	d

Description The Branch If Indicator On instruction is a conditional branch. The d-character specifies the indicator to be tested as a criterion for the branch. If the indicator is on, then the program branches to the instruction specified by the I-address. If the indicator is off, the next sequential instruction is executed. The indicators to be tested and their d-character codings are:

Indicator		d-character
Unequal compare	($B \neq A$)	/
Equal compare	($B = A$)	S
Low compare	($B < A$)	T
High compare	($B > A$)	U

Testing of an indicator does not affect its setting.

Branch If Character Equal — B

Op-Code	I-address	B-address	d-character
B	iii	bbb	d

Description The Branch If Character Equal instruction causes the single character at the B-address to be compared to the d-character. If they are identical, the program branches to the instruction specified by the I-address. If they are different, the program proceeds to the next instruction in sequence.

Word Marks Word marks at the B-address are ignored and thus do not affect the comparison.

Compare — C

Op-Code	A-address	B-address
C	aaa	bbb

Description The Compare instruction causes the information in the B-field to be compared with an equal number of characters in the A-field. The bit configuration of each character of the two fields is compared and appropriate indicators are set as described below.

Word Marks The first word mark encountered terminates the operation. If the A-field is longer than the B-field, extra positions beyond the length of the B-field will not be compared.

Indicators If the two fields are identical, character by character, an Equal Compare results and the Equal Indicator is turned on. If the fields are not equal then an Unequal Compare results and the Unequal Indicator is turned on. In addition, the High Indicator is turned on if the B-field is greater than the A-field and the Low indicator if the A-field is greater than the B-field. If the B-field is longer than the A-field, the Unequal and High Indicators are turned on regardless of their contents. All indicators are reset only by another Compare instruction.

Clear Storage — CS

Op-Code	A-address
/	aaa

Description The Clear Storage instruction causes the storage to be cleared to blanks beginning at the location specified by the A-address and continuing downward through the nearest hundreds position.

Word Marks Both word marks and data are cleared by this instruction.

Clear Storage and Branch — CS

Op-Code	I-address	B-address
/	iii	bbb

Description The Clear Storage and Branch instruction causes the storage to be cleared (including word marks) to blanks beginning at the location specified by the B-address and continuing downward through the nearest hundreds position in the same manner as the Clear Storage instruction. Upon completion of the clearing operation the program branches to the instruction specified by the I-address.

Clear Word Mark — CW

Op-Code	A-address	B-address
▣ or)	aaa	
▣ or)	aaa	bbb

Description The Clear Word Mark instruction causes a word mark to be cleared from the location specified by the A-address. If a B-address is also used, word marks will be cleared from the locations specified by each address. If no word mark existed in either location prior to the instruction, there will be no change at that location. Data will not be disturbed.

Halt — H

Op-Code
·

Description The Halt instruction causes the computer to stop. Depressing the start key will cause the program to proceed to the next instruction in sequence.

Halt and Branch — H

Op-Code	I-address
·	iii

Description The Halt and Branch instruction causes the computer to stop. Depressing the start key will cause the computer to proceed to the instruction designated by the I-address.

Load Characters to A Word Mark — LCA

Op-Code	A-address	B-address
L	aaa	bbb

Description The Load Characters to A Word Mark instruction causes the characters and word mark from the A-field to replace the B-field. The A-field remains undisturbed.

Word Marks The A-field must contain a word mark to terminate the transmission of data. All word marks in the B-field are cleared and a word mark is placed in the B-field corresponding to that in the A-field.

Move Characters to A or B Word Mark — MCW

Op-Code	A-address	B-address
M	aaa	bbb

Description The Move Characters to A or B Word Mark instruction causes the field specified by the A-address (A-field) to be moved to corresponding positions of the B-field. The A-field remains undisturbed but the B-field is lost.

Word Marks The first word mark encountered in either field stops the transmission of data. Existing word marks in either field are not disturbed.

No Operation — NOP

Op-Code

N

Description The only purpose of the No Operation instruction is to cause the program to proceed to the next instruction in sequence. The instruction may have the format of any allowable instruction, that is, it may have an A-address, an A-address and a B-address, etc.

Punch a Card — P

Op-Code

4

Description The Punch a Card instruction causes information in storage locations 101 through 180 (PUNCH area) to be punched in columns 1 through 80 respectively of a Hollerith coded aard. The machine coding is converted to Hollerith coding prior to punching. The information stored in the PUNCH area is undisturbed. This instruction punches only information from the PUNCH area of storage onto the card.

Word Marks Word marks are not punched, are not affected by the Punch instruction and do not affect punching in any manner.

Punch and Branch — P

Op-Code	I-address
4	iii

Description The Punch and Branch instruction causes the computer to punch the contents of the PUNCH area (in the same manner as the Punch a Card instruction) then branch to the instruction specified by the I-address.

Read a Card — R

Op-Code

1

Description The Read a Card instruction causes the information in columns 1 through 80 of a Hollerith coded card to be read into storage positions 001 through 080 respectively. The Hollerith code from each column is converted to the appropriate computer coding as it is read into the computer. The Read a Card instruction always reads into positions 001 through 080 (the READ area).

Word Marks Word marks which exist in the READ area prior to execution of the instruction are not disturbed nor do they affect the reading of information.

Read and Branch — R

Op-Code	I-address
1	iii

Description The Read and Branch instruction causes the computer to read one Hollerith coded card (in the same manner as the Read a Card instruction) then branch to the instruction specified by the I-address.

Subtract — S

Op-Code	A-address	B-address
S	aaa	bbb

Description The Subtract instruction causes the data in the A-field to be subtracted algebraically from the data in the B-field. The A-field is not disturbed and the resulting difference is stored in the B-field.

Word Marks Word marks control the subtract operation in the same manner as the Add instruction.

Subtract (Single Address) — S

Op-Code	A-address
S	aaa

Description The four position Subtract instruction causes the A-field to be subtracted from itself with zero replacing the original A-field.

Word Marks The word mark is not affected.

Set Word Mark SW

Op-Code	A-address	B-address
2	aaa	
2	aaa	bbb

Description The Set Word Mark instruction causes a word mark to be set at the location specified by the A-address. If a B-address is also used, word marks will be set at the locations specified by each address. Existing word marks are undisturbed. Data at the location (or locations) will not be disturbed.

Write a Line W

Op-Code
2

Description The Write a Line instruction causes the information in storage locations 201 through 300 (the WRITE area) to be printed on the printer (or typewriter). The information will remain in the WRITE area of storage after execution of the instruction. This instruction always prints information from all 100 positions of the WRITE area.

Word Marks Word marks are not printed, are not affected by the instruction and do not affect printing in any manner.

Write and Branch W

Op-Code	I-address
2	iii

Description The Write and Branch instruction causes the computer to print the contents of the PRINT area (in the manner as the Write a Line instruction) then branch to the instruction specified by the I-address.

SECTION 2

141 SYMBOLIC PROGRAMMING SYSTEM

The Symbolic Programming System, SPS, has been developed to facilitate logical, efficient programming with a minimum of actual coding effort. It almost completely relieves the programmer of the task of assigning actual storage location to the instructions and data of the program and allows him to refer to them by easy-to-remember names of his choice.

INSTRUCTIONS

Instructions written in SPS contain a label, an operation code, an A-operand, a B-operand, and a d-character. Any of the parts except the operation code may be left blank.

The label is the symbolic representation of the location in memory in which the instruction will be stored. It may have from one to six alphameric characters, the first of which must be alphameric and must be placed in column 8 of the coding sheet. The label may be left blank if no reference is made to the instruction elsewhere in the program.

The operation code is mnemonic and consists of from one to three characters starting in column 14.

If the instruction requires an A-operand it is written in columns 17 through 26. If no A-operand is used those columns may be left blank. The address of an operand may be expressed as a symbol, an actual location, or an asterisk. If it is written symbolically, it takes the same form as a label. If it is written as an actual location, it must be a four-digit number with leading zeros where necessary (although four digits are written on the coding sheet, only three characters will be used in memory). If an asterisk is used its equivalent address is that of the right most, or low order, position in the instruction defined by the statement.

The address of an operand may be adjusted by placing the number of characters of adjustment in columns 24 through 26 and the sign of the adjustment in colum 23. Leading zeros may be omitted but the units digit of the adjustment must be in column 26.

The indexing character (column 27) is not used with the 141.

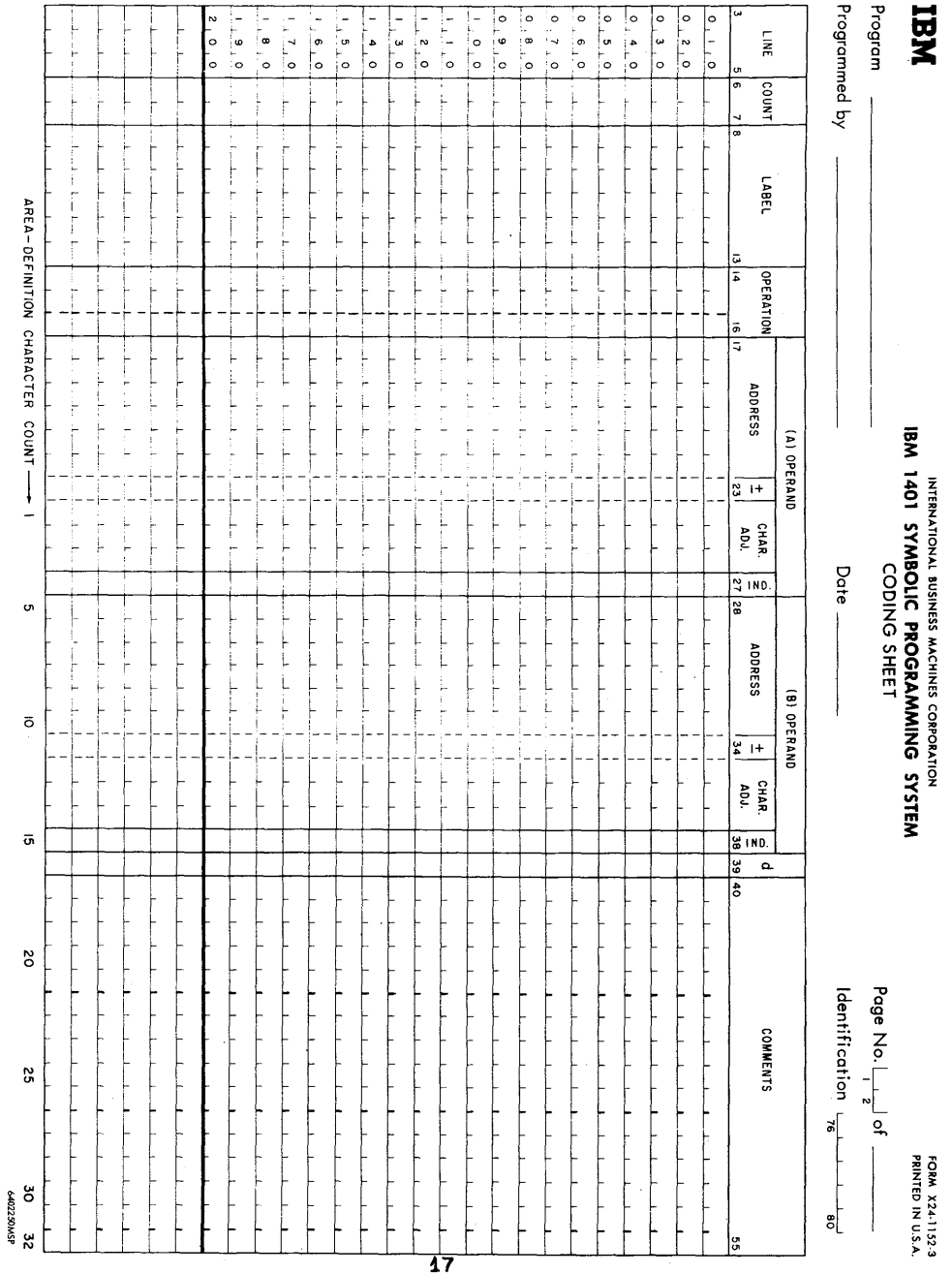
If the instruction requires a B-operand, its address is written in columns 28 through 37 in the same form as the A-operand.

When an instruction requires a d-character, the actual machine code is placed in column 39.

COMMENTS

Short comments may be placed in columns 40 through 55 of the instruction cards. Longer comments may be placed on "Comment Cards". These cards are identified by an asterisk in column 8. The remainder of the card, columns 9 through 55, is available for the comment.

A sample coding sheet is shown on the next page.



DECLARATIVES

Define Constant With Word Mark — DCW

The symbolic operation code DCW causes a constant to be loaded into storage and sets a word mark in the high-order (left most) position of the constant field. The number of characters in the constant field is specified in the Count portion of the coding sheet, (columns 6 and 7). The symbolic label by which the constant is referenced is placed in the Label area (columns 8 through 13). The code DCW is placed in columns 14 through 16. Column 17 must contain an asterisk to indicate to the assembler that it may choose the location of the constant field or else columns 17 through 20 must contain the desired storage location of the low order position (right most) of the constant field. The constant itself begins in column 24 and may extend through column 55 giving a maximum of 32 characters. If the constant is to be a signed number, the sign may be placed in column 23.

Define Constant — DC

The symbolic operation code DC causes a constant to be loaded into storage without a word mark. Otherwise, it is identical to the DCW.

Define Symbol — DS

The operation code DS causes the processor to assign equivalent addresses to labels or to assign storage for work areas. The DS differs from DC and DCW statements in that neither data nor word marks are loaded during assembly. The number of positions to be reserved in storage is specified in the Count portion of the coding sheet. If it is desired to refer symbolically to the low order position of the field reserved, then a label must be placed in the Label field. If the assembler is to assign the address, an asterisk must be placed in column 17 of the coding sheet. If it is desired to equate the label to an actual address, then that address is written beginning in column 17 and the Count field of the coding sheet is left blank. It is not possible to character adjust DS statements.

Define Symbolic Address — DSA

The DSA statement causes a three character machine language address which the assembler has assigned to a label to be stored as a constant when the program is loaded.

The number of characters need not be specified in the Count portion of the coding sheet since it is automatically assigned three storage positions by the processor. If it is desired to refer to the address of the address field, a symbol may be written in the Label portion of the coding sheet. Column 17 may contain an asterisk thus allowing the assembler to assign the storage positions or else columns 17 through 20 may contain the desired storage locations of the low order position for the address field. The symbol whose equivalent address is to be the address field is written beginning in column 28 of the B-operand.

CONTROL STATEMENTS

Origin — ORG

The ORG statement causes the assembler to assign addresses to the following instructions beginning at the location specified by the statement. The symbolic operation code ORG must be placed in the operation field and the absolute address at which storage assignment is to be made must be written in columns 17 through 20 of the coding sheet.

Execute — EX

The EX statement causes the computer to suspend loading of the object program and execute part of the program prior to continuing the loading process. The symbolic operation code EX must be placed in the operation field and the symbolic or actual address of the first instruction to be executed when the loading process is suspended must be placed in the A-operand portion of the coding sheet. The card containing the Execute statement must be inserted at the point in the source program where suspension of loading is desired in order to execute the preceeding portion.

End — END

The END statement is an indication to the assembler that the last card of the source program has been processed. The symbolic operation code END must be placed in the operation field and the address of the first instruction, either actual or symbolic, must be placed in the A-operand portion of the coding sheet.

SECTION 3

EXERCISES

Exercise 1

Write a program that will reproduce a card, that is, will read a card and punch a card identical to the one read.

Exercise 2

Write a program that will read a card and punch a card with the information from columns 1 - 40 of the card read in columns 41 -80 of the card punched and the information from columns 41 - 80 of the card read in columns 1 - 40 of the card punched.

Exercise 3

Write a program that will reproduce an entire deck of cards.

Exercise 4

Write a program that will read one card and will punch copy after copy of it until the machine is stopped by the operator.

Exercise 5

Write a program that will print a directory of telephone extensions from a deck of personnel cards. The cards and directory forms are as follows:

Card Columns	Field	Print Positions
1 - 18	Name	1 - 18
19	First Initial	20
20	Second Initial	22
21 - 60	Not used in this program	
61 - 64	Telephone Extension	28 - 31
65 - 80	Not used in this program	

Exercise 6

Write a program that will read cards containing numeric fields A, B, and C and will punch corresponding cards that contain fields A, B, C, and D, where D = A + B - C. The card columns are shown on the next page.

Field	Card Columns	Card
A	1 - 6	Input and Output
B	7 - 11	Input and Output
C	12 - 14	Input and Output
D	75 - 80	Output Only

Assume that no overflows will occur.

Exercise 7

Write a program that will check the sequence of employee numbers found in columns 75 - 80 of a deck of cards. The program should stop the machine if it finds any employee number that is not larger than the one in the previous card.

Exercise 8

Write a program that will punch consecutive numbers 001 through 015 in columns 78 - 80 of the first 15 blank cards in the punch hopper and stop automatically before punching a sixteenth card.

Exercise 9

Write a program that will calculate and punch D, where D = A + B - C (all values are positive). Provide for decimal alignment, rounding (half-adjustment), and over flow. The card columns and decimal form of each field is as follows:

Input Card	A	Col.	5 - 8	XXX.X
	B		9 - 12	XX.XX
	C		13 - 14	XX.
Output Card	D		7 - 10	XXXX.

Exercise 10

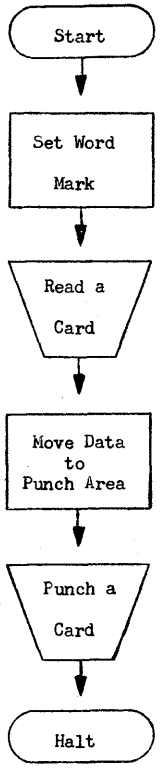
Write a program that will up-date a customer's charge account after a new purchase has been recorded. A new balance card is to be punched and a listing of each customer's name, new balance, and limit is to be printed. If the new balance exceeds the customer's limit the words OVER LIMIT are also to be printed on his entry. The card columns and print positions are as follows:

Filed	Input Card	Output Card	Listing
Name	1 - 20	1 - 20	11 - 30
Balance	21 - 30	21 - 30	35 - 44
Charge	31 - 40		
Limit	71 - 80	71 - 80	49 - 58
'OVER LIMIT'			63 - 72

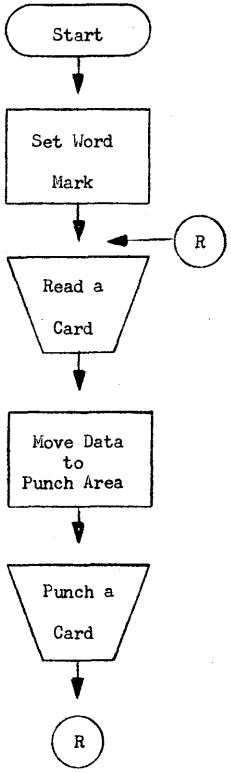
The Limit field is to be punched with leading zeros.

BLOCK DIAGRAMS

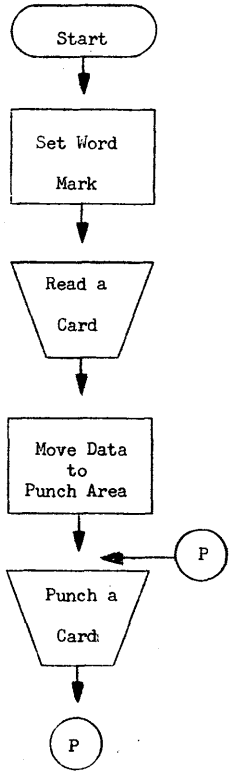
Exercises 1 and 2



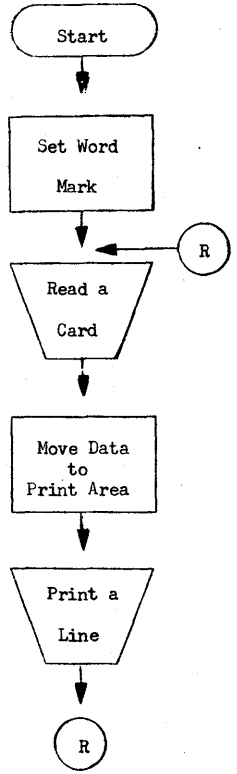
Exercise 3



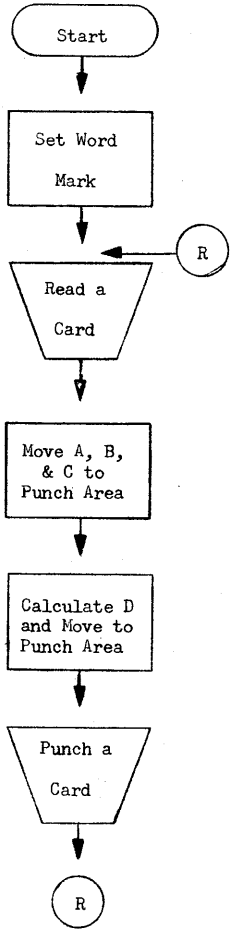
Exercise 4



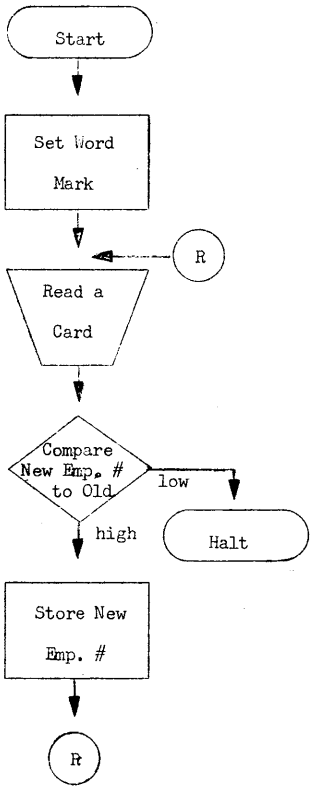
Exercise 5



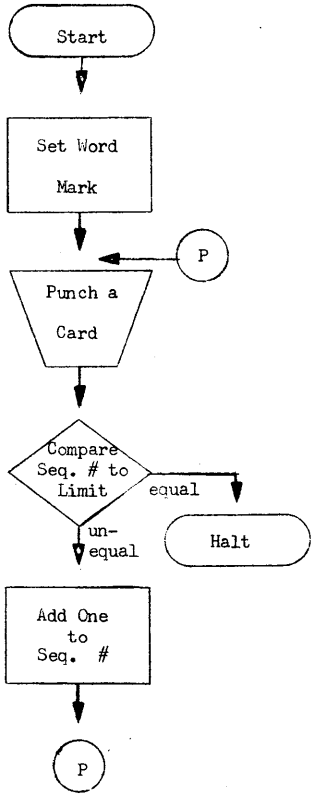
Exercise 6



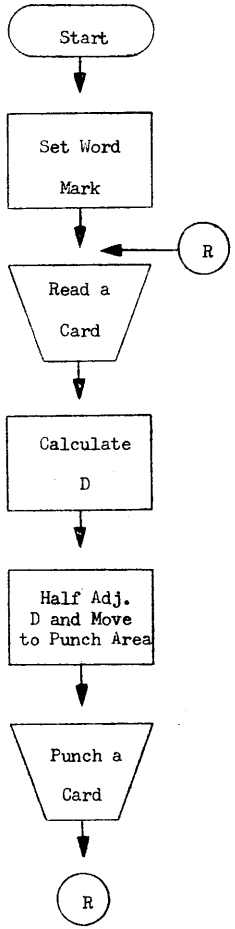
Exercise 7



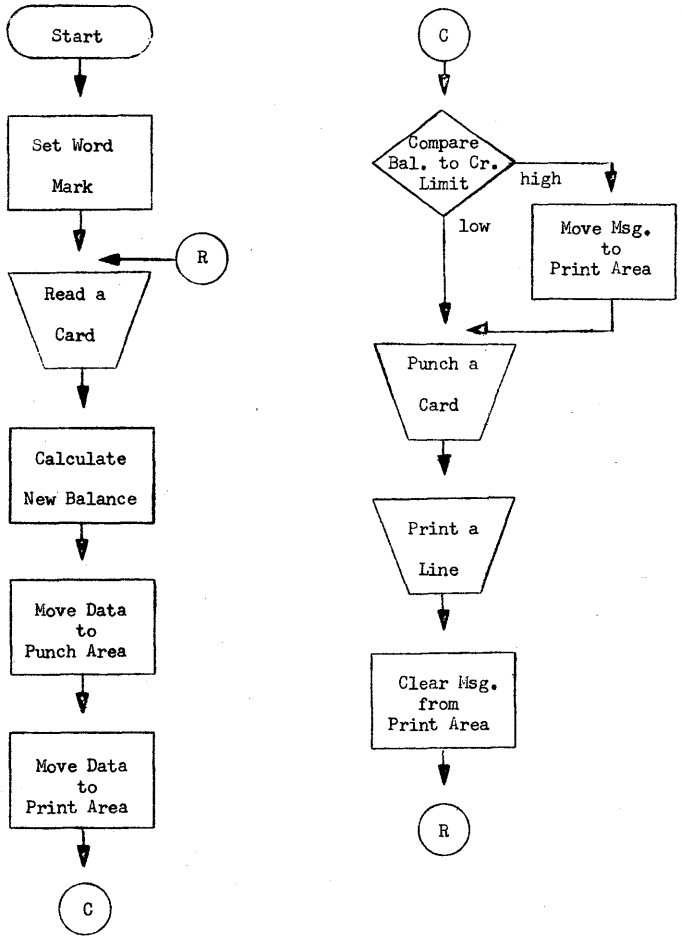
Exercise 8



Exercise 9



Exercise 10



SOLUTIONS TO EXERCISES

EXERCISE 1

```
PG LIN    LABEL   OP   A-OPERAND   B-OPERAND   COMMENTS
01 010    START   SW   0001                    DEFINE 80 POS FL
01 020            R                            READ ONE CARD
01 030            MCW  0080        0180        MOVE TO PCH
01 040            P                            PUNCH ONE CARD
01 050            H                            HALT
01 060            NOP                          PROVIDE WM
01 070            END  START
```

EXERCISE 2

```
02 010    START   SW   0001                    DEFINE FIRST FLD
02 020            SW   0041                    DEFINE SECOND FL
02 030            R                            READ ONE CARD
02 040            MCW  0040        0180        MOVE TO PCH AREA
02 050            MCW  0080        0140        MOVE TO PCH AREA
02 060            P                            PUNCH A CARD
02 070            H                            HALT
02 080            NOP                          PROVIDE WM
02 090            END  START
```

EXERCISE 3

```
03 010    START   SW   0001                    DEFINE FIELD
03 020    READ    R                            READ A CARD
03 030            MCW  0080        0180        MOVE TO PCH AREA
03 040            P                            PUNCH A CARD
03 050            B    READ                    BRANCH TO READ
03 060            END  START
```

EXERCISE 4

```
PG LIN     LABEL   OP   A-OPERAND   B-OPERAND   COMMENTS

04 010     START   SW   0001                    DEFINE FIELD
04 020             R                            READ CARD
04 030             MCW  0080        0180        MOVE TO PCH AREA
04 040     PUNCH   P                            PUNCH
04 050             B    PUNCH                   REPEAT PUNCH
04 060             END  START
```

EXERCISE 5

```
05 010     START   SW   0001                    DEFINE FIELDS
05 020             SW   0019
05 030             SW   0020
05 040             SW   0061
05 050     READ    R                            READ CARD
05 060             MCW  0018        0218        ASSEMBLE LINE
05 070             MCW  0019        0220
05 080             MCW  0020        0222
05 090             MCW  0064        0231
05 100             W                            PRINT A LINE
05 110             B    READ                    RETURN TO READ
05 120             END  START
```

EXERCISE 6

```
06 010     START   SW   0001                    DEFINE FIELDS
06 020             SW   0007
06 030             SW   0012
06 040     READ    R                            READ CARD
06 050             MCW  0006        0106        MOVE INPUT TO
06 060             MCW  0011        0111        PUNCH AREA
06 070             MCW  0014        0114
06 080             S    0014        0006        A-C
06 090             A    0011        0006        A+B-C
06 100             MCW  0006        0180        MOVE D TO PCH AR
06 110             P                            PUNCH CARD
06 120             B    READ                    LOOP
06 130             END  START
```


EXERCISE 7

```
PG LIN     LABEL   OP   A-OPERAND   B-OPERAND   COMMENTS

07 010     BEGIN   SW   0075                    DEFINE EMPNO FLD
07 020     READ    R                            READ CARD
07 030             C    STORE       0080        COMP WITH LST CD
07 040             B    LOOP                    ULOOP IF OK
07 050             H                            HALT
07 060     LOOP    MCW  0080        STORE
07 070             B    READ
07 080   6 STORE   DCW  *
07 090             END  BEGIN
```

EXERCISE 8

```
08 010     FIRST   SW   0178                    DEFINE FLD
08 020     PUNCH   P                            PUNCH
08 030             C    0180        LIMIT       TEST FOR LIMIT
08 040             B    HALT                    S
08 050             A    ONE         0180        STEP SEQ NO
08 060             B    PUNCH                   LOOP
08 070     HALT    H                            HALT
08 080   3 LIMIT   DCW  *           015
08 090   1 ONE     DCW  *           1
08 100   3         DCW  0180        001
08 110             END  FIRST
```

EXERCISE 9

```
09 010     START   SW   0005                    DEFINE FIELDS
09 020             SW   0009
09 030             SW   0013
09 040     READ    R                            READ CARD
09 050             A    0008        ACCUM -  1  A
09 060             A    0012        ACCUM       A+B
09 070             S    0014        ACCUM -  2  A+B-C
09 080             A    FIVE        ACCUM -  1  HALF ADJUST
09 090             MCW  ACCUM -  2  0110        MOVE TO D
09 100             P
09 110             MCW  ZEROS       ACCUM       CLEAR ACCUM
09 120             B    READ
09 130   6 ACCUM   DCW  *           000000
09 140   6 ZEROS   DCW  *           000000
09 150   1 FIVE    DCW  *           5
09 160             END  START
```

EXERCISE 10

```
PG LIN     LABEL    OP   A-OPERAND   B-OPERAND   COMMENTS
10 010     START    SW   0001                    DEFINE FIELDS
10 020              SW   0021
10 030              SW   0031
10 040              SW   0071
10 050     READ     R                            READ CARD
10 060              A    0040        0030        CALC NEW BAL
10 070              MCW  0020        0120        MOVE TO PCH
10 080              MCW  0030        0130
10 090              MCW  0080        0180
10 100              MCW  0020        0230        MOVE TO PRINT
10 110              MCW  0030        0244
10 120              MCW  0080        0258
10 130              C    0030        0080        TEST FOR HI BAL
10 140              B    OVER                    T
10 150     PUNCH    P                            PUNCH
10 160              W                            WRITE
10 170              MCW  BLANK       0272        CLEAR MSG
10 180              B    READ
10 190     OVER     MCW  MSG         0272        INSERT MSG
10 200              B    PUNCH
10 210 10  BLANK    DCW  *
10 220 10  MSG      DCW  *           OVER LIMIT
10 230              END  START
```


SUBROUTINES

The following subroutines written in 141 language were contributed by Mr. Wilson T. Price of Merritt College, Oakland, California. In preparing these routines, simplicity of arithmetic method, compatability with the 1401, and compatability with each other were primary considerations. Speed of operation was deemed the least important feature since students write 141 programs as learming experience and not for production runs.

THE MULTIPLY SUBROUTINE

TITLE: Multiply

MNEMONIC: MULT

PURPOSE: To provide the capability of multiplying a number containing up to 8 digits by a second number containing up to 8 digits to form a product up to 16 digits in length.

STORAGE REQUIREMENTS:

Multiplicand	(MULTD)	081 through 089
Multiplier	(MULTR)	091 through 099
Product	(PROD)	181 through 196
Additional work areas		197 through 200
Program		100 additional locations as assigned by assembler

LINKAGE: Move the multiplicand of m digits to MULTD. This field will then occupy storage positions (090 - m) through 089. Move the multiplier of n digits to MULTR. This field will then occupy storage positions (100 - n) through 099. Move the return Branch instruction to MULTX + 3. Branch to MULT. The linkage is illustrated below:

```
          MCW (Multiplicand)    MULTD
          MCW (Multiplier)      MULTR
          MCW RETURN - 1        MULTX + 3
          B   MULT
          B   RETURN
RETURN    (next instruction in program)
```

After completion of the operation, the product of m + n digits will be in PROD. Both the multiplicand and multiplier remain in their respective areas.

WORD MARKS: Word marks are placed in locations 081, 091, and 181 with DCW's during assembly and care must be exercised that they are not cleared during execution of the main program.

CLEARING: Initially all three work areas will be zero, further clearing is left to the programmer. Blanking or zeroing of the multiplicand and multiplier areas will only be necessary if the new values contain fewer digits than the previous quantities which utilized these areas. Zeroing of the product accumulator will always be necessary unless it is desired to sum products.

SCALING: Decimal alignment is the responsibility of the programmer. The number of decimal places in the product is equal to the sum of the number of decimal places in the multiplicand and the multiplier.

MULTIPLY SUBROUTINE

```
PG LIN    LABEL  OP  A-OPERAND      B-OPERAND      COMMENTS
M1 010    MULT   MCW M16            M3    + 3
M1 020           MCW M17            M6    + 6
M1 030    M3     MCW MULTR - 7      M19   - 1
M1 040    M4     C   M19   - 1      M18   - 1
M1 050           B   M9                          U
M1 060    M6     A   MULTD          PROD  - 7
M1 070           S   M18            M19
M1 080           B   M4
M1 090    M9     SW  M3    + 1      M6    + 4
M1 100           A   M18   - 1      M3    + 3
M1 110           A   M18   - 1      M6    + 6
M1 120           CW  M3    + 1      M6    + 4
M1 130           C   M3    + 3      M16   - 2
M1 140           B   M3                      /
M1 150    MULTX  B   0000
M1 160 03 M16    DCW *       092
M1 170 02 M17    DCW *       89
M1 180 02 M18    DCW 0198    10
M1 190 02 M19    DCW 0200    00
M1 200 09 MULTD  DCW 0089    000000000
M1 210 09 MULTR  DCW 0099    000000000
M1 220 16 PROD   DCW 0196    0000000000000000
```

THE DIVIDE SUBROUTINE

TITLE: Divide

MNEMONIC: DIV

PURPOSE: To provide the capability of dividing a number containing up to 16 digits by a second number containing up to 8 digits to form a quotient of up to 8 digits.

STORAGE REQUIREMENTS:

Dividend	(DIVD)	181 through 196
Divisor	(DIVR)	081 through 089
Quotient	(QUOT)	091 through 099
Program		154 additional locations as assigned by assembler

LINKAGE: Move the dividend of m digits to DIVD. This field will then occupy storage positions (197 - m) through 196. Move the divisor of n digits to DIVR. This field will then occupy storage positions (090 - n) through 089. Move the return Branch instruction to DIVX + 3. Branch to DIV.

```
          MCW (Dividend)   DIVD
          MCW (Divisor)    DIVR
          MCW RETURN - 1   DIVX + 3
          B   DIV
          B   RETURN
RETURN    (next instruction in program)
```

After completion of the operation, the quotient will be located at QUOT and the remainder at DIVD. The divisor remains in DIVR but the dividend is lost.

WORD MARKS: Word marks are placed in locations 081, 091, and 181 with DCW's during assembly and care must be taken that they are not cleared during execution of the main program.

CLEARING: Initially all three work areas will contain zeroes, further clearing is left to the programmer. Zeroing of the dividend and divisor areas will be necessary if new values contain fewer digits then previous quantities which utilized these areas. The high order position (081) of the divisor must contain zero. Zeroing of the quotient accumulator will always be necessary unless it is desired to sum quotients.

SCALING: Decimal alignment is the responsibilty of the programmer. The rules to follow are:listed on the next page.

1. Multiply dividend and divisor by the appropriate power of ten to clear decimals from divisor.
2. Muliply dividend and expected quotient by the same power of ten to obtain greater accuracy.
3. Upper eight digits (181 through 188) of dividend must be less than divisor.

The following examples illustrate scaling in the divide subroutine:

$$\frac{38}{1.2} = \frac{380}{12}$$

		Number	Location of low order position
1.	Before division	380	DIVD
		12	DIVR
	After division	31	QUOT
		8 (remainder)	DIVD
2.	Before division	380^0	DIVD
		12	DIVR
	After division	31^6	QUOT
		0^8 (remainder)	DIVD
3.	Before division	380^00	DIVD
		12	DIVR
	After division	31^66	QUOT
		0^08 (remainder)	DIVD

DIVIDE SUBROUTINE

```
PG LIN     LABEL  OP   A-OPERAND      B-OPERAND      COMMENTS
D1 010     DIV    MCW  D24            D7   + 3
D1 020            MCW  D25            D11  + 6
D1 030            MCW  D24            D13  + 6
D1 040            C    DIVR           DIVD - 8
D1 050            B    D7                            T
D1 060            H    DIVX
D1 070     D7     MCW  DIVD - 7       D26  - 1
D1 080     D8     C    D26  - 1       DIVR
D1 090            B    D13                           U
D1 100            S    DIVR + 1       D26
D1 110     D11    A    D24  - 2       QUOT - 7
D1 120            B    D8
D1 130     D13    MCW  D26  - 1       DIVD - 7
D1 140            SW   D7   + 1       D11  + 6
D1 150            SW   D13  + 4
D1 160            C    D24            D11  + 6
D1 170            A    D24  - 2       D7   + 3
D1 180            A    D24  - 2       D11  + 6
D1 190            A    D24  - 2       D13  + 6
D1 200            CW   D7   + 1       D11  + 6
D2 010            CW   D13  + 4
D2 020            B    D7                            /
D2 030     DIVX   B    0000
D2 040 03  D24    DCW  *      189
D2 050 01  D25    DCW  *      2
D2 060 10  D26    DCW  *      0000000000
D2 070 09  DIVR   DCW  0089   000000000
D2 080 09  QUOT   DCW  0099   00000000
D2 090 16  DIVD   DCW  0196   0000000000000000
```


THE SUPPRESS ZERO SUBROUTINE

TITLE: Suppress Zero

MNEMONIC: SUPZR

PURPOSE: Given a numeric field of 9 digits or fewer, to suppress leading zeroes (that is change high order zeroes to blanks).

STORAGE REQUIREMENTS:

Work area	(SZARG)	091 through 099
Program		82 additional locations as assigned by assembler

LINKAGE: Move the numeric field of m digits to SZARG. The field will then occupy storage positions (100 - m) through 099. For example, a three digit field would occupy positions 097 through 099. Move the return Branch instruction to SUPZRX + 3. Branch to SUPZR.

```
          MCW (Argument)   SZARG
          MCW RETURN - 1   SUPZRX + 3
          B   SUPZR
          B   RETURN
RETURN    (next instruction in program)
```

After completion of the operation, the field with leading zeroes suppressed will remain in its original location. If the entire field is zero, then one zero will remain.

WORD MARKS: A word mark is set at location 091 during processing by the assembler. If cleared during execution of the main program it should be reset.

CLEARING: Initially the work area will be zero, further clearing is left to the programmer. Zeroing will always be necessary if the new field contains fewer digits than the previous quantity which utilized this area.

SUPPRESS ZERO SUBROUTINE

```
PG LIN     LABEL   OP   A-OPERAND      B-OPERAND      COMMENTS
S1 010     SUPZR   MCW  SZ15           SZ3   + 3
S1 020             MCW  SZ15           SZ5   + 6
S1 030     SZ3     C    SZARG - 8      SZ13  - 1
S1 040             B    SUPZRX                      T
S1 050     SZ5     MCW  SZ14           SZARG - 8
S1 060             SW   SZ3    + 1     SZ5   + 4
S1 070             A    SZ13           SZ3   + 3
S1 080             A    SZ13           SZ5   + 6
S1 090             CW   SZ3    + 1     SZ5   + 4
S1 100             C    SZ3    + 3     SZ15  - 1
S1 110             B    SZ3                         /
S1 120     SUPZRX  B    0000
S1 130 02  SZ13    DCW  *        01
S1 140 01  SZ14    DCW  *
S1 150 02  SZ15    DCW  *        91
S1 160 09  SZARG   DCW  0099     000000000
```


THE EDIT SUBROUTINE

TITLE: Edit

MNEMONIC: EDIT

PURPOSE: To provide the capability to edit a field of up to 8 digits consisting of dollars and cents. Leading zeroes are suppressed and a decimal point, a comma (if needed) and a floating dollar sign are placed in appropriate positions of the field.

STORAGE REQUIREMENTS:

Input field	(EDIN)	081 through 089
Output field	(EDOUT)	181 through 191
Program		127 additional locations as assigned by assembler

LINKAGE: Move the field of m digits to be edited to EDIN. This field will then occupy positions (090 - m) through 089. Move the return Branch instruction to EDITX + 3. Branch to EDIT.

```
          MCW (Argument)   EDIN
          MCW RETURN - 1   EDITX + 3
          B   EDIT
          B   RETURN
RETURN    (next instruction in program)
```

After completion of the operation, the edited field will be located at EDOUT. The original field remains in EDIN.

WROD MARKS: Word marks are placed in locations 081 and 191 with DCW's during assembly and care must be taken that they are not cleared during execution of the main program.

CLEARING: Initially both work areas will be zero, further clearing is left up to the programmer. Zeroing of the input area (EDIN) will be necessary if the new argument contains fewer digits than previous quantities which utilized this area. The output area (EDOUT) is self clearing.

SCALING: Quantities which are edited must consist of a dollar and cent amount. The following examples illustrate scaling in the edit subroutine:

Input field	Output field
12345678	$123,456.78
12345	$123.45
123	$1.23
12	$0.12

EDIT SUBROUTINE

```
PG LIN     LABEL  OP  A-OPERAND      B-OPERAND      COMMENTS
E1 010     EDIT   MCW ED20           ED9   + 3
E1 020            MCW ED20           ED11  + 6
E1 030            MCW EDIN           EDOUT
E1 040            MCW ED21           EDOUT - 2
E1 050            MCW EDIN   - 2     EDOUT - 3
E1 060            MCW ED21   - 1     EDOUT - 6
E1 070            MCW EDIN   - 5     EDOUT - 7
E1 080            MCW ED21   - 2     EDOUT - 10
E1 090     ED9    C   EDOUT - 9     ED19  - 1
E1 100            B   EDITX                    T
E1 110     ED11   MCW ED21   - 2     EDOUT - 9
E1 120            SW  ED9    + 1     ED11  + 4
E1 130            A   ED19           ED9   + 3
E1 140            A   ED19           ED11  + 6
E1 150            CW  ED9    + 1     ED11  + 4
E1 160            C   ED9    + 3     ED20  - 1
E1 170            B   ED9                      /
E1 180     EDITX  B   0000
E1 190 02  ED19   DCW *        01
E1 200 02  ED20   DCW *        82
E1 210 04  ED21   DCW *          $,.
E1 220 11  EDOUT  DCW 0191     00000000000
E1 230 09  EDIN   DCW 0089     000000000
```


OPERATING PROCEDURES

Four versions of the 141 SPS Assembler and the 141 Simulator are available in order to permit maximum utilization of the computer hardware. These are identified as:

Non-Monitor Versions

Version A - Basic 1620
Version B - 1620 with 1443 Printer

Monitor Versions

Version C - 1620 with 1311 Disk Storage Drive and indirect addressing
Version D - 1620 with 1443 Printer, 1311 Disk Storage Drive, and indirect addressing

Letters preceeding each procedure statement below identify the versions to which they apply.

141 SPS ASSEMBLER

Prepare Console

Version	
A C	1) Set left typewriter margin at 10 and right margin at 95.
A B C D	2) Set Parity Switch and I/O Switch to STOP.
A B C D	3) Set O'Flow Switch to PROGRAM.
C D	4) Set Disk Switch to PROGRAM.
A B C D	5) Set Program Switches 1 and 2 according to the options listed below.

Assemble SPS Programs

Version		
A B	1)	Place the 141 SPS Assembler deck in the reader hopper in the 9-edge face-down position.
C D	2)	Place the following Monitor cards in the reader hopper: "COLD START", ≠ ≠ JOB, and ≠ ≠ XEQ 141SPS.
A B C D	3)	Place SPS source program decks in the reader hopper. Any number of programs may be stacked for assembly. The last card of each deck must be an END statement.
A B C D	4)	With the machine in MANUAL mode, press the LOAD key on the 1622 Reader-Punch unit.

Program Switch Options

Version		
A B C D	1)	Switch 1 and 2 off - Object deck will be punched and program will be listed.
A B C D	2)	Switch 1 off and Switch 2 on - Object deck will be punched but program listing will be suppressed except for incorrect statements. A program listing can be prepared from the object program cards on an IBM 407 Accounting Machine. This option will greatly reduce assembly time for versions A and C.
A B C D	3)	Switch 1 on and Switch 2 off - Object deck will be suppressed and program will be listed on the console typewriter (or printer).
A B C D	4)	Switch 1 and 2 on - Object deck and program listing will be suppressed. This combination can be used as an edit run. Programs from an entire class can quickly be scanned for errors with only incorrect statements being listed. The particular op-code or address that is erroneous will appear as the symbol =. For easy recognition, be sure that the source cards are numbered in columns 1 through 5 and that the IDENTIFICATION field, columns 76 through 80, is punched.

Long Programs

Version		
A B	1)	An SPS assembly is a two pass operation but the 141 SPS assembler only requires that the cards be fed through once if the number of cards in the source program does not exceed 100. This reduces the amount of card handling and permits the stacking of of programs. If the number of cards in a source program is greater than 100, images of the first 100 cards are held in storage and copies of the remaining cards are punched for a second pass. These cards are removed from the PUNCH stacker and placed

		in the READ hopper at the end of PASS I. Only those statements in excess of 100 need be processed twice.
C D	2)	Images of the source cards are stored on the disk and therefore the length of the program does not effect the operating procedures.

141 SIMULATOR

Prepare Console

Version		
A C	1)	Set left margin at 10 and right margin at 95.
A B C D	2)	Set Parity Switch to STOP.
A B C D	3)	Set O'Flow Switch to PROGRAM.
C D	4)	Set Disk Switch to PROGRAM.
A B C D	5)	Set Program Switches 1,2,3, and 4 according to the options listed at the end of this section.

Load Simualtor

Version		
A B	1)	Place 141 Simulator deck in the reader hopper in the 9-edge face-down position.
C D	2)	Place the following Monitor cards in the reader hopper: "COLD START", ≠ ≠ JOB, and ≠ ≠ XEQ 141SIM
A B C D	3)	With the machine in MANUAL mode, press the LOAD key on the 1622 Reader-Punch unit. When the Simulator is loaded the typewriter will automatically begin typing a list of the functions that the simulator will perform and the request words that will initiate these functions.

Functions Performed	Request by Typing
Load Program From Card Reader	LOAD
Clear 141 Storage	CLEAR
Alter Storage From Typewriter	ALTER
Dump Contents of 141 Storage	DUMP
Begin Execution of Program	EXECUTE
Return to 1620 Monitor	EXIT (C & D only)

Select the Desired Function

Each function, except EXIT, is available in all versions.

a) The typewriter will type the words REQUESTED FUNCTION IS and then stop.
b) The operator then types the word LOAD, CLEAR, ALTER, DUMP, EXECUTE or EXIT and presses the RELEASE and START keys on the console or the RS key on the typewriter.
c) If a function runs to completion the simulator will automatically request the next function. If the function is interrupted by turning on Program Switch 1, the operator may return to the request statement by pressing, in order, the RESET, INSERT, RELEASE, and START keys on the console.

The LOAD Function

Programs that have been assembled by SPS can be loaded with this function.

a) Place the SPS object deck, including the two clear storage cards and the bootstrap card, in the hopper.
b) Type the request word LOAD and press the RELEASE and START keys.
c) Press READER START, if necessary.

The CLEAR Function

The 141 storage can be cleared (set to blanks) with this function.

a) Type the request word CLEAR and press the RELEASE and START keys.
b) When the clearing operation is completed the typewriter will request the next function.

The ALTER Function

Instructions and data, including word marks, in the 141 storage can be altered with this function. This may be used for debugging a program or entering complete small demonstration programs directly in machine language.

a) Type the request word ALTER and press the RELEASE and START keys.
b) The typewriter will type BEGINNING AT.
c) Type the three digit 141 location at which the alteration is desired and press the RELEASE and START keys.
d) The typewriter will repeat this location to verify it.
e) Type the instructions and data in machine language, disregarding word marks. This is the only instance where the operator will have to use the typewriter shift key. For all other entries the typewriter will automatically be in the proper alphabetic or numeric shift. At any convenient place, at least one character before the end of the line, cease typing and press the RELEASE and START keys.

f) The typewriter carriage will return for a second line. This line will indicate the presence or absence of word marks. If the character above requires a word mark type a 1, if it does not, strike the space bar. Continue to type 1's and spaces until the carriage has moved across the entire line above. In the first position after completion of the word mark line, type a record mark, and then press the RELEASE and START keys.
g) The typewriter will now type the address of the next storage location that will be altered if steps c) and f) are repeated.
h) When altering is completed press, in order, the RESET, INSERT, RELEASE, and START keys. The EXECUTE function can be used to start the program.

The DUMP Function

When a 141 program is stopped either by a programmed halt or by an error condition, it is desirable to be able to "DUMP" the Instruction Register (I-REG), the Operation Register (OP-REG) and the storage. The DUMP function will list the contents of the I-REG, which will be the address of the next character to be accessed, the contents of the OP-REG, which is the operation code of the last instruction to be executed, and the contents of the 141 storage as it stood when the program stopped.

a) Type the request word DUMP and press the RELEASE and START keys.
b) When the entire storage is dumped the typewriter will request the next function.

The EXECUTE Function

Execution of 141 programs can be started with this function.

a) Type the request word EXECUTE and press the RELEASE and START keys.
b) The typewriter will type BEGINNING AT.
c) Type the three-digit 141 location of the first instruction to be executed and press the RELEASE and START keys.

The EXIT Function

In versions C and D this function returns control to the 1620 Monitor.

a) Type the request word EXIT.
b) Press the RELEASE and START keys.

Program Switch Options

a) Program Switch 1 - Turning Program Switch 1 on will cause the program to halt at the end of the execution of the current 141 instruction. The operator may either press START to continue with the next 141 instruction or he may press RESET, INSERT, RELEASE and START to request a new function.

b) Program Switch 2 - When Program Switch 2 is off the DUMP function will use the typewriter or printer. When it is on the DUMP function will use the card punch. These cards can be listed on an IBM 407 Accounting Machine.

c) Program Switch 3 - Cards punched by the DUMP function can be reloaded with the LOAD function with Program Switch 3 on. With Program Switch 3 off SPS self-loading cards can be loaded.

d) Program Switch 4 - If Program Switch 4 is on at the time the simulator is loaded the typing of the list of functions will be omitted.

Special Notes

a) Restarting Programs - 141 programs can be stopped, dumped, and later restarted by the following procedure:
   1) Stop the program by turning Program Switch 1 on.
   2) Dump the program on cards using the DUMP function with Program Switch 2 on.
   3) Later re-load the program using the LOAD function with Program Switch 3 on.

b) Console Lights - When a 141 program is stopped by a program halt, an error halt, or by turning on Program Switch 1, the operation code of the instruction just completed can be determined from the DIGIT AND BRANCH lights on the console. The 1620 display can be converted to a 141 operation code by using the following table:

DIGIT AND BRANCH	141 OP-CODE	SPS OP-CODE	DIGIT AND BRANCH	141 OP-CODE	SPS OP-CODE
03	. or)	H	53	L	LCA
04	¤	CW	54	M	MCW
21	/	CS	55	N	NOP
23	,	SW	62	S	S
41	A	A	71	1	R
42	B	B	72	2	W
43	C	C	74	4	P

The address of the next instruction to be executed can be determined by pressing the DISPLAY MAR key with the MEMORY ADDRESS REGISTER SELECTOR rotated to the OR-2 position. The 141 address of the next instruction will be displayed by the lights of the MEMORY ADDRESS REGISTER.

c) Loading Machine Language Programs - Machine language programs can be loaded either by typing them under the control of the ALTER function or by key punching them in the Card Dump format and loading them using the LOAD function with Program Switch 3 on.

Card Dump Format - Cards in this format must be sequentially numbered with the odd numbered cards containing the program and data characters and the even numbered cards containing the word marks.

COLUMNS	ODD	EVEN
1 - 2	Card Number	Card Number
4 - 6	Blank	Blank except for last card
9 - 11	Load address	Blank
20 - 69	Program or Data	1's for word marks

In an odd numbered card, up to fifty characters to be loaded are punched starting in column 20. In columns 9 through 11 is punched the address of the location in storage where the character in column 20 is to be stored. In columns 20 through 69 of an even numbered card are punched 1's for the word marks associated with the characters in columns 20 through 69 of the previous card. In columns 4 through 6 of the last card (even numbered) is punched the address at which execution is to begin.

d) Monitor END OF JOB cards - In versions C and D, ‡ ‡ ‡ ‡ END OF JOB cards may be used to facilitate continuous operation. In an SPS Assembly, if the last source program deck is followed by an END OF JOB card control is automatically returned to the 1620 Monitor and the next program, such as the 141 Simulator, can be called into storage for execution.

During the execution of a 141 program using the 141 Simulator, an END OF JOB card following the data cards will automatically cause a return to request a new function. This may be any 141 Simulator function, including the EXIT function which will return control to the 1620 Monitor.

e) 1443 Carriage Control - In versions B and D, no provisions are made for control of the 1443 printer carriage except for an automatic detection of a channel 12 punch which will skip the paper form to the channel 1 position.

SOURCE PROGRAM LISTING

13.0.017

- VERSION C -

1620 With 1311 Disk Storage Drive

and Indirect Addressing

```
00 001 *                          - 141 -  ASSEMBLER
00 002 *                                FOR 1620 - 1311
00 003 *
00 004 * INITIALIZATION AND STORE PROGRAM ROUTINE
00 005 *
00 006 ASMBLY TFM   CDCNT,0
00 007        TF    11,INIT+11
00 008        BLC   *+12
00 009        TFM   ERRCNT,0
00 010        TFM   IMAGE+5,0
00 011        SEEK  DCTL
00 012        TFM   ICTR,0333,8
00 013        TFM   MADDR+6,LABEL-15
00 014        TFM   MLABEL+6,LABEL-18
00 015        TDM   OVERSW,0
00 016        CF    IDENT-1
00 017 LC     BLC   NOEND
00 018        RACD  LAREA
00 019        BNR   *+36,LAREA
00 020        BNR   *+24,LAREA+2
00 021        B     796
00 022        AM    CDCNT,1,10
00 023        C     END+4,LAREA+30
00 024        BE    MOD
00 025        C     AST,LAREA+14
00 026        BE    MOD
00 027        C     CCTL,LAREA+30
00 028        BE    MOD
00 029        C     CEX,LAREA+30
00 030        BE    MOD
00 031        C     CDCW-2,LAREA+28
00 032        BE    DCDSR
00 033        C     CDSA-2,LAREA+28
00 034        BE    DCDSR
00 035        C     CORG,LAREA+30
00 036        BE    ORGR
00 037        C     CB,LAREA+30
00 038        BNE   *+60
00 039        C     BLANK,LAREA+64
00 040        BE    *+36
00 041        TFM   CNT,8,9
00 042        B     REPL
00 043        TFM   CNT,0,9
00 044        BD    INCR,LAREA+75
00 045        BD    INCR,LAREA+76
00 046        B     *+24
00 047 INCR   AM    CNT,1,10
00 048        C     BLANK,LAREA+64
00 049        BNE   P7
00 050        C     BLANK,LAREA+42
00 051        BNE   P4
00 052        AM    CNT,1,10
00 053        B     REPL
00 054 P7     AM    CNT,7,10
00 055        B     REPL
00 056 P4     AM    CNT,4,10
00 057 REPL   TD    LAREA+12,CNT
```

```
00 058          TDM   LAREA+11,7
00 059          TD    LAREA+10,CNT-1
00 060          C     BLANK,LAREA+24
00 061          BE    REPLIM
00 062          TF    LOC,ICTR
00 063          BTM   LTABLE
00 064 REPLIM   A     ICTR,CNT
00 065          BD    MOD,OVERSW
00 066          BD    OVERR,ICTR-3
00 067 MOD      PUT   DCTL
00 068          AM    IMAGE+5,2,10
00 069          C     END+4,LAREA+30
00 070          BE    PASS2
00 071          B     LC
00 072          DC    5,0
00 073 LTABLE   AM    MLABEL+6,15,10
00 074          AM    MADDR+6,15,10
00 075          CM    MADDR+6,LABEL+15*90
00 076          BNL   LBLERR
00 077          SF    LAREA+13
00 078 MLABEL   TF    0,LAREA+24
00 079          SF    LOC-2
00 080 MADDR    TF    0,LOC
00 081          CF    LAREA+13
00 082          BB
00 083 LBLERR   RCTY
00 084          WATY  LBLMSG
00 085          B     OVERR+36
00 086 LBLMSG   DAC   18,LABEL TABLE FULL.@,
00 087 ORGR     TD    ICTR,LAREA+38
00 088          TD    ICTR-1,LAREA+36
00 089          TD    ICTR-2,LAREA+34
00 090          B     MOD
00 091 DCDSR    TD    CNT,LAREA+12
00 092          TD    CNT-1,LAREA+10
00 093          C     CDSA,LAREA+30
00 094          BNE   *+48
00 095          TF    CNT,C3
00 096          TFM   LAREA+12,0073,8
00 097          CF    LAREA+9
00 098          C     AST,LAREA+32
00 099          BNE   ABSLT
00 100          A     ICTR,CNT
00 101          C     BLANK,LAREA+24
00 102          BE    REPLIM+12
00 103          TF    LOC,ICTR
00 104          SM    LOC,1,10
00 105          BTM,  LTABLE
00 106          B     REPLIM+12
00 107 ABSLT    TD    LOC,LAREA+38
00 108          TD    LOC-1,LAREA+36
00 109          TD    LOC-2,LAREA+34
00 110          BTM   LTABLE
00 111          B     MOD
00 112 NOEND    RCTY
00 113          WATY  ENDMSG
00 114          RCTY
```

```
00 115          H
00 116          B     LC+12
00 117 OVERR    TDM   OVERSW,1
00 118          RCTY
00 119          WATY  OVMSG
00 120          RCTY
00 121          WATY  LAREA
00 122          RCTY
00 123          B     MOD
00 124 LAREA    DAC   50,
00 125          DS    10
00 126 LDIN     DAC   20,L       1056          ,
00 127 IDENT    DAC   10,        000@,
00 128          DS    30
00 129          DGM
00 130 ADDRAR   DC    4,0
00 131 ICTR     DC    4,0
00 132 BLANK    DC    12,0
00 133 LOC      DC    4,0
00 134 CNT      DC    3,0
00 135 LABEL    DSB   15,90
00 136 CDCNT    DC    5,0
00 137          DC    1,@
00 138 ERRCNT   DC    5,0
00 139          DC    1,@
00 140 DCTL     DDW   ,IMAGE,,,A
00 141 IMAGE    DDA   ,0,0,2,LAREA-1
00 142          DC    1,@
00 143 END      DAC   3,END,
00 144 ENDMSG   DAC   48,END CARD MISSING. LOAD END CARD AND PUSH START.@,
00 145 OVMSG    DAC   22,PROGRAM EXCEEDS CORE.@,
00 146 *
00 147 * PASS2
00 148 *
00 149 PASS2    BD    ASMBLY,OVERSW
00 150          TFM   ICTR,0332,8
00 151          TFM   CDCNT,0
00 152          TFM   IMAGE+5,0
00 153          SF    LAREA+149
00 154          TF    CS1+158,IDENT+8
00 155          TF    CS2+158,IDENT+8
00 156          TF    BS+158,IDENT+8
00 157          BC2   PCS
00 158          RCTY
00 159          WATY  CS1
00 160          RCTY
00 161          WATY  CS2
00 162          RCTY
00 163          WATY  BS
00 164          RCTY
00 165          RCTY
00 166 PCS      BC1   PULIM
00 167          WACD  CS1
00 168          WACD  CS2
00 169          WACD  BS
00 170 PULIM    GET   DCTL
00 171          SF    LDIN+7
```

```
00 172          SF    LDIN+23
00 173          TF    IDENT+8,BS+158
00 174          TFM   IDENT+15,70707
00 175          TF    LDIN+38,CLDIN+38
00 176          AM    CDCNT,1,10
00 177          TD    CNT,LAREA+12
00 178          TD    CNT-1,LAREA+10
00 179          TDM   ERRSW,0
00 180          C     AST,LAREA+14
00 181          BE    ORGR2+48
00 182          C     END+4,LAREA+30
00 183          BE    ENDCD
00 184          C     CORG,LAREA+30
00 185          BE    ORGR2
00 186          C     CCTL,LAREA+30
00 187          BE    ORGR2+48
00 188          C     CEX,LAREA+30
00 189          BE    EXR2
00 190          TF    ADDRAR,ICTR
00 191          AM    ADDRAR,1,10
00 192          TD    IDENT+16,ADDRAR
00 193          TD    IDENT+14,ADDRAR-1
00 194          TD    IDENT+12,ADDRAR-2
00 195          A     ICTR,CNT
00 196          TFM   LDIN+11,70707
00 197          TD    LDIN+12,ICTR
00 198          TD    LDIN+10,ICTR-1
00 199          TD    LDIN+8,ICTR-2
00 200          C     CDCW,LAREA+30
00 201          BE    DCWR2
00 202          C     CDC,LAREA+30
00 203          BE    DCWR2-12
00 204          C     CDS,LAREA+30
00 205          BE    DSR
00 206          C     CDSA,LAREA+30
00 207          BE    DSAR
00 208          SF    LAREA+10
00 209          TF    WA,CNT
00 210          AM    WA,66,10
00 211          TFM   LDIN+5,70707
00 212          TD    LDIN+6,WA
00 213          TD    LDIN+4,WA-1
00 214          TD    LDIN+2,WA-2
00 215          BTM   TABLE,0
00 216          C     C8,CNT
00 217          BNE   *+84
00 218          TFM   DMOD+6,LDIN+36
00 219          TFM   DMOD+18,LDIN+35
00 220          BTM   DMOD,0
00 221          BTM   BADD,0
00 222          BTM   AADD,0
00 223          B     TESTSW
00 224          C     C7,CNT
00 225          BNE   *+48
00 226          BTM   BADD
00 227          BTM   AADD
00 228          B     TESTSW
```

```
00 229          C     C5,CNT
00 230          BNE   *+60
00 231          TFM   DMOD+6,LDIN+30
00 232          TFM   DMOD+18,LDIN+29
00 233          BTM   DMOD
00 234          B     *-84
00 235          C     C4,CNT
00 236          BE    *-108
00 237          C     C2,CNT
00 238          BNE   TESTSW
00 239          TFM   DMOD+6,LDIN+24
00 240          TFM   DMOD+18,LDIN+23
00 241          BTM   DMOD,0
00 242          B     TESTSW
00 243          TFM   LDIN,54,10
00 244 DCWR2    C     TT,CNT
00 245          BL    *+36
00 246          C     BLANK-10,CNT
00 247          BL    *+72
00 248          AM    ERRCNT,1,10
00 249          TDM   ERRSW,1
00 250          TF    LDIN+12,LBS
00 251          TF    LDIN+6,LBS
00 252          B     TESTSW
00 253          SF    LAREA+43
00 254          CM    LAREA+44,20,10
00 255          BNE   MINUS+12
00 256          SF    LAREA+10
00 257          TFM   MINUS+6,LAREA+43
00 258          A     MINUS+6,CNT
00 259          A     MINUS+6,CNT
00 260 MINUS    TDM   0,5
00 261          TFM   T24,23,9
00 262          A     T24,CNT
00 263          TF    LDIN+6,ZERO
00 264          TD    LDIN+6,T24
00 265          TD    LDIN+4,T24-1
00 266          TD    LDIN+2,T24-2
00 267          C     AST,LAREA+32
00 268          BE    AAA
00 269          S     ICTR,CNT
00 270          TD    LDIN+12,LAREA+38
00 271          TD    LDIN+10,LAREA+36
00 272          TD    LDIN+8,LAREA+34
00 273 AAA      TF    IDENT+16,LDIN+12
00 274          B     TESTSW
00 275 DSR      TFM   LDIN,55,10
00 276          TF    LDIN+28,LDIN+12
00 277          TF    LDIN+12,BRRD
00 278          TFM   LDIN+22,70,10
00 279          C     AST,LAREA+32
00 280          BE    *+48
00 281          S     ICTR,CNT
00 282          SF    LAREA+33
00 283          TF    LDIN+28,LAREA+38
00 284          TF    IDENT+16,LDIN+28
00 285          B     TESTSW
```

```
00 286 DSAR   TFM   LDIN+6,7276,8
00 287        BTM   BADD
00 288        CF    LDIN+30
00 289        TF    LAREA+50,LDIN+34
00 290        TF    LDIN+34,BLANK-6
00 291        B     MINUS+72
00 292        DC    5,0
00 293 TABLE  C     CMCW,LAREA+30
00 294        BE    INM
00 295        C     CR,LAREA+30
00 296        BE    IN1
00 297        C     CW,LAREA+30
00 298        BE    IN2
00 299        C     CP,LAREA+30
00 300        BE    IN4
00 301        C     CSW,LAREA+30
00 302        BE    INCOM
00 303        C     CCW,LAREA+30
00 304        BE    INLOZ
00 305        C     CA,LAREA+30
00 306        BE    INA
00 307        C     CS,LAREA+30
00 308        BE    INS
00 309        C     CC,LAREA+30
00 310        BE    INC
00 311        C     CH,LAREA+30
00 312        BE    INH
00 313        C     CB,LAREA+30
00 314        BE    INB
00 315        C     CCS,LAREA+30
00 316        BE    INSLH
00 317        C     CLCA,LAREA+30
00 318        BE    INL
00 319        C     CNOP,LAREA+30
00 320        BE    INN
00 321        TF    LDIN+22,LBS-4
00 322        B     INLBS+12
00 323 INM    TFM   LDIN+22,54,10
00 324        BB
00 325 IN1    TFM   LDIN+22,71,10
00 326        BB
00 327 IN2    TFM   LDIN+22,72,10
00 328        BB
00 329 IN4    TFM   LDIN+22,74,10
00 330        BB
00 331 INCOM  TFM   LDIN+22,23,10
00 332        BB
00 333 INLOZ  TFM   LDIN+22,04,10
00 334        BB
00 335 INA    TFM   LDIN+22,41,10
00 336        BB
00 337 INS    TFM   LDIN+22,62,10
00 338        BB
00 339 INC    TFM   LDIN+22,43,10
00 340        BB
00 341 INH    TFM   LDIN+22,03,10
00 342        BB
```

```
00 343 INB     TFM   LDIN+22,42,10
00 344         BB
00 345 INSLH   TFM   LDIN+22,21,10
00 346         BB
00 347 INL     TFM   LDIN+22,53,10
00 348         BB
00 349 INN     TFM   LDIN+22,55,10
00 350         BB
00 351 * DMOD ROUTINE
00 352         DC    5,0
00 353 DMOD    TD    LDIN+36,LAREA+76
00 354         TD    LDIN+35,LAREA+75
00 355         BB
00 356 *B ADDRESS ROUTINE
00 357         DC    5,0
00 358 BADD    BD    *+36,LAREA+54
00 359         BD    *+24,LAREA+53
00 360         B     INLBS
00 361         C     S9,LAREA+54
00 362         BL    BINACT
00 363         C     AST,LAREA+54
00 364         BNE   *+84
00 365         TF    WA,ICTR
00 366         TF    LDIN+34,ZERO
00 367         TD    LDIN+34,WA
00 368         TD    LDIN+32,WA-1
00 369         TD    LDIN+30,WA-2
00 370         B     BCADJ
00 371         TFM   LEXIT+6,BCADJ
00 372         TF    LDIN+34,ZERO
00 373         TFM   LOOK+23,LAREA+64
00 374         TFM   XX+6,LDIN+34
00 375         B     LOOK
00 376 BCADJ   C     BLANK-6,LAREA+74
00 377         BNE   ADJB
00 378         BB
00 379 INLBS   TF    LDIN+34,LBS
00 380         AM    ERRCNT,1,10
00 381         TDM   ERRSW,1
00 382         BB
00 383 BINACT  TF    LDIN+34,ZERO
00 384         TD    LDIN+34,LAREA+60
00 385         TD    LDIN+32,LAREA+58
00 386         TD    LDIN+30,LAREA+56
00 387         B     BCADJ
00 388 ADJB    TD    WA1,LAREA+72
00 389         TD    WA1-1,LAREA+70
00 390         TD    WA1-2,LAREA+68
00 391         SF    WA1-2
00 392         TD    WA2,LDIN+34
00 393         TD    WA2-1,LDIN+32
00 394         TD    WA2-2,LDIN+30
00 395         SF    WA2-2
00 396         C     BSIGN,LAREA+66
00 397         BNE   *+36
00 398         S     WA2,WA1
00 399         B     *+24
```

```
00 400           A     WA2,WA1
00 401           CF    WA2
00 402           TD    LDIN+34,WA2
00 403           TD    LDIN+32,WA2-1
00 404           TD    LDIN+30,WA2-2
00 405           BB
00 406 *A ADDRESS ROUTINE
00 407           DC    5,0
00 408 AADD      BD    *+36,LAREA+32
00 409           BD    *+24,LAREA+31
00 410           B     INLBSA
00 411           C     S9,LAREA+32
00 412           BL    AINACT
00 413           C     AST,LAREA+32
00 414           BNE   *+84
00 415           TF    WA,ICTR
00 416           TF    LDIN+28,ZERO
00 417           TD    LDIN+28,WA
00 418           TD    LDIN+26,WA-1
00 419           TD    LDIN+24,WA-2
00 420           B     ACADJ
00 421           TFM   LEXIT+6,ACADJ
00 422           TF    LDIN+28,ZERO
00 423           TFM   LOOK+23,LAREA+42
00 424           TFM   XX+6,LDIN+28
00 425           B     LOOK
00 426 ACADJ     C     BLANK-6,LAREA+52
00 427           BNE   ADJA
00 428           BB
00 429 INLBSA    TF    LDIN+28,LBS
00 430           B     INLBS+12
00 431 AINACT    TF    LDIN+28,ZERO
00 432           TD    LDIN+28,LAREA+38
00 433           TD    LDIN+26,LAREA+36
00 434           TD    LDIN+24,LAREA+34
00 435           B     ACADJ
00 436 ADJA      TD    WA1,LAREA+50
00 437           TD    WA1-1,LAREA+48
00 438           TD    WA1-2,LAREA+46
00 439           SF    WA1-2
00 440           TD    WA2,LDIN+28
00 441           TD    WA2-1,LDIN+26
00 442           TD    WA2-2,LDIN+24
00 443           SF    WA2-2
00 444           C     BSIGN,LAREA+44
00 445           BNE   *+36
00 446           S     WA2,WA1
00 447           B     *+24
00 448           A     WA2,WA1
00 449           CF    WA2
00 450           TD    LDIN+28,WA2
00 451           TD    LDIN+26,WA2-1
00 452           TD    LDIN+24,WA2-2
00 453           BB
00 454 * LABEL TABLE LOOK UP
00 455 LOOK      TFM   *+18,LABEL-3
00 456           C     0,0
```

```
00 457           BE    MVADDR
00 458           C     MLABEL+6,LOOK+18
00 459           BE    INSLB
00 460           AM    LOOK+18,15,10
00 461           B     LOOK+12
00 462 MVADDR    TF    XX+11,LOOK+18
00 463           AM    XX+11,3,10
00 464           TF    XX+23,XX+11
00 465 XX        TD    0,0
00 466           BNF   *+24
00 467 LEXIT     B     0
00 468           SM    XX+6,2,10
00 469           SM    XX+11,1,10
00 470           SM    XX+23,1,10
00 471           B     XX
00 472 INSLB     TDM   ERRSW,1
00 473           AM    ERRCNT,1,10
00 474           TF    *+18,XX+6
00 475           TF    0,LBS
00 476           B     LEXIT
00 477 TESTSW    BD    PRINT,ERRSW
00 478           BNC2  PRINT
00 479           BC1   *+24
00 480           WACD  LAREA
00 481           AM    IMAGE+5,2,10
00 482           B     PULIN
00 483 ENDCD     TF    LDIN+12,ENDC
00 484           BTM   AADD
00 485           TF    LDIN+6,LDIN+28
00 486           TF    LDIN+28,BLANK
00 487           TF    LDIN+16,BLANK-8
00 488           TF    IDENT+16,BLANK-6
00 489           BD    *+24,ERRSW
00 490           BC2   *+36
00 491           RCTY
00 492           WATY  LAREA
00 493           BC1   *+24
00 494           WACD  LAREA
00 495           RCTY
00 496           RCTY
00 497           TD    CNTMSG+6,CDCNT
00 498           TD    CNTMSG+4,CDCNT-1
00 499           TD    CNTMSG+2,CDCNT-2
00 500           TD    CNTMSG+28,ERRCNT
00 501           TD    CNTMSG+26,ERRCNT-1
00 502           TD    CNTMSG+24,ERRCNT-2
00 503           WATY  CNTMSG
00 504           BNLC  ASMBLY
00 505           H
00 506 INIT      B     ASMBLY,0,0
00 507 ORGR2     TD    ICTR,LAREA+38
00 508           TD    ICTR-1,LAREA+36
00 509           TD    ICTR-2,LAREA+34
00 510           SM    ICTR,1,10
00 511           TFM   LDIN,55,10
00 512           TF    LDIN+12,BRRD
00 513           TF    IDENT+16,BLANK-6
```

```
00 514           B     TESTSW
00 515 EXR2      BTM   AADD
00 516           TF    LDIN+6,LDIN+28
00 517           TF    LDIN+28,BLANK-6
00 518           TFM   LDIN,42,10
00 519           TF    LDIN+12,BLANK-6
00 520           TF    IDENT+16,BLANK-6
00 521           B     TESTSW
00 522 PRINT     WATY  LAREA
00 523           RCTY
00 524           B     TESTSW+24
00 525 CS1       DAC   50,,008015,022026,030034,041,045,053,0570731026      ,
00 526           DAC   31,                                                  @,
00 527 CS2       DAC   50,L072116,110106,105117B101/999,027A074028)027B00102,
00 528           DAC   31,70B026/0991,001/00111710                          @,
00 529 BS        DAC   50,,008015,022029,056063/056029                      ,
00 530           DAC   31,     ,0240671056                                  @,
00 531 CLDIN     DAC   20,L0010561056          ,
00 532 AST       DAC   1,*,
00 533 CNTMSG    DAC   23, 000 CARDS  000 ERRORS@,
00 534 ENDC      DC    14,21707070707870
00 535 BRRD      DC    8,71707576
00 536 ZERO      DC    6,707070
00 537 CMCW      DC    6,544366
00 538 CR        DC    6,590000
00 539 CW        DC    6,660000
00 540 CP        DC    6,570000
00 541 CSW       DC    6,626600
00 542 CCW       DC    6,436600
00 543 CA        DC    6,410000
00 544 CS        DC    6,620000
00 545 CC        DC    6,430000
00 546 CH        DC    6,480000
00 547 CB        DC    6,420000
00 548 CCS       DC    6,436200
00 549 CLCA      DC    6,534341
00 550 CNOP      DC    6,555657
00 551 CDCW      DC    6,444366
00 552 CDSA      DC    6,446241
00 553 CDC       DC    6,444300
00 554 CDS       DC    6,446200
00 555 CORG      DC    6,565947
00 556 CCTL      DC    6,436353
00 557 CEX       DC    6,456700
00 558 LBS       DC    6,333333
00 559 TT        DC    3,32
00 560 T24       DC    3,23
00 561 S9        DC    2,69
00 562 BSIGN     DC    2,20
00 563 WA        DC    3,0
00 564 C8        DC    3,8
00 565 C7        DC    3,7
00 566 C5        DC    3,5
00 567 C4        DC    3,4
00 568 C3        DC    3,3
00 569 C2        DC    3,2
00 570 WA1       DC    3,0
```

```
00 571 WA2      DC    3,0
00 572 OVERSW   DC    1,0
00 573 ERRSW    DC    1,0
00 574          DEND  ASMBLY
```

SYMBOL TABLE
141SPS-C

TESTSW	09496	REPLIM	03110	OVERSW	10843	MVADDR	09316	MLABEL	03284
LTABLE	03224	LBLMSG	03381	LBLERR	03344	INLBSA	08932	ERRCNT	05448
ENDMSG	05481	CNTMSG	10601	BINACT	08398	ASMBLY	02402	AINACT	08956
ADDRAR	04064	AAA	07120	AADD	08680	ABSLT	03656	ACADJ	08896
ADJA	09016	ADJB	08458	AST	10599	BADD	08098	BCADJ	08314
BLANK	04080	BRRD	10667	BS	10397	BSIGN	10815	CA	10715
CB	10739	CC	10727	CCS	10745	CCTL	10793	CCW	10709
CDC	10775	CDCNT	05442	CDCW	10763	CDS	10781	CDSA	10769
CEX	10799	CH	10733	CLCA	10751	CLDIN	10559	CMCW	10679
CNOP	10757	CNT	04087	CORG	10787	CP	10697	CR	10685
CS	10721	CSW	10703	CS1	10073	CS2	10235	CW	10691
C2	10836	C3	10833	C4	10830	C5	10827	C7	10824
C8	10821	DCDSR	03464	DCTL	05450	DCWR2	06772	DMOD	08056
DSAR	07276	DSR	07144	END	05475	ENDC	10659	ENDCD	09568
ERRSW	10844	EXR2	09952	ICTR	04068	IDENT	04011	IMAGE	05458
INA	07858	INB	07954	INC	07906	INCOM	07810	INCR	02906
INH	07930	INIT	09844	INL	08002	INLBS	08350	INLOZ	07834
INM	07714	INN	08026	INS	07882	INSLB	09436	INSLH	07978
IN1	07738	IN2	07762	IN4	07786	LABEL	04102	LAREA	03861
LBS	10805	LC	02546	LDIN	03971	LEXIT	09376	LOC	04084
LOOK	09232	MADDR	03308	MINUS	06964	MOD	03146	NOEND	03716
ORGR	03416	ORGR2	09856	OVERR	03776	OVMSG	05577	PASS2	05620
PCS	05824	PRINT	10036	PULIM	05872	P4	03014	P7	02990
REPL	03026	S9	10813	TABLE	07354	TT	10808	T24	10811
WA	10818	WA1	10839	WA2	10842	XX	09352	ZERO	10673

```
00 001 *                          - 141 -  SIMULATOR
00 002 *                           FOR 1620 - 1311
00 003 *
00 004 * INITIALIZER ROUTINE
00 005 *
00 006 BEGIN   TR    19998,ASK+41
00 007         TF    11,PRELD+11
00 008         SF    17982
00 009         BC4   CLEAR+24
00 010         RCTY
00 011         WATY  HEADG
00 012         RCTY
00 013         BTM   WRT,WORD
00 014         BTM   WRT,WORD+10
00 015         BTM   WRT,WORD+22
00 016         BTM   WRT,WORD+34
00 017         BTM   WRT,WORD+44
00 018         BTM   WRT,WORD+60
00 019         B     CLEAR+24
00 020 WRT     BC4   CLEAR+24
00 021         RCTY
00 022         WATY  FUNCT,,2
00 023         BC4   CLEAR+24
00 024         WATY  -WRT+1
00 025         AM    WRT+30,80,10
00 026         BB
00 027 INITZR  RCTY
00 028         RCTY
00 029         WATY  ASK
00 030         RATY  TESTL
00 031         SF    TESTL-1
00 032         C     TESTL+6,WORD+6
00 033         BE    START
00 034         C     TESTL+8,WORD+18
00 035         BE    CLEAR
00 036         C     TESTL+8,WORD+30
00 037         BE    ALTER
00 038         C     TESTL+6,WORD+40
00 039         BE    DSTART
00 040         C     TESTL+12,WORD+56
00 041         BE    INBRCH
00 042         C     TESTL+6,WORD+66
00 043         BE    796
00 044         WATY  INERR
00 045         RCTY
00 046         RCTY
00 047         B     INITZR
00 048 INBRCH  WATY  BGMSG
00 049         RNTY  TESTL-1
00 050         TD    17985,TESTL-1
00 051         TD    17987,TESTL
00 052         TD    17989,TESTL+1
00 053         RCTY
00 054         RCTY
00 055         SF    17990
00 056         B     B
00 057 TESTL   DAC   10,          ,
```

```
00 058 HEADG  DAC  36,FUNCTIONS PERFORMED                    ,
00 059        DAC  18,REQUEST BY TYPING@,
00 060 FUNCT  DAC  40,   LOAD PROGRAM FROM CARD READER       @,
00 061        DAC  40,   CLEAR 141 STORAGE                   @,
00 062        DAC  40,   ALTER STORAGE FROM TYPEWRITER       @,
00 063        DAC  40,   DUMP CONTENTS OF 141 STORAGE        @,
00 064        DAC  40,   BEGIN EXECUTION OF PROGRAM          @,
00 065        DAC  40,   RETURN TO 1620 MONITOR              @,
00 066 WORD   DAC  5,LOAD@,
00 067        DAC  6,CLEAR@,
00 068        DAC  6,ALTER@,
00 069        DAC  5,DUMP@,
00 070        DAC  8,EXECUTE@,
00 071        DAC  5,EXIT@,
00 072 ASK    DAC  23,REQUESTED FUNCTION IS @,
00 073 BGMSG  DAC  15, BEGINNING AT @,
00 074 INERR  DAC  24,  INVALID REQUEST WORD.@,
00 075 *
00 076 * LOADER ROUTINE
00 077 *
00 078 START  RCTY
00 079        RCTY
00 080        BC3  LDUMP
00 081        TF   18161,BLANKS
00 082        TF   18141,BLANKS
00 083        TF   18121,BLANKS
00 084        TF   18101,BLANKS
00 085        TF   18081,BLANKS
00 086        TF   18061,BLANKS
00 087        TF   18041,BLANKS
00 088        TF   18021,BLANKS-1
00 089        RACD 18003
00 090        TFM  FTEST+11,18002
00 091        B    NEXTIN
00 092 *
00 093 * INSTRUCTION ACCESS ROUTINE
00 094 *
00 095 NEXTIN BNC1 *+60
00 096        BTM  CVTREG,0,10
00 097        TF   *+35,17983
00 098        TF   *+18,IREG-1
00 099        H    0,0
00 100        AM   FTEST+11,2,10
00 101        BT   TESTHI,FTEST+11
00 102 FTEST  BNF  *+36,0,7
00 103        TF   17984,-FTEST-11
00 104        B    TABLE
00 105        AM   FTEST+11,6,10
00 106        BT   TESTHI,FTEST+11
00 107        TF   17990,-FTEST-11
00 108        BNF  *+24,-FTEST-11
00 109        B    TABLE
00 110        CM   17983,42,10
00 111        BNE  *+72
00 112        BD   *+60,17990
00 113        TF   *+35,FTEST+11
00 114        AM   *+23,1,10
```

```
00 115          BD    *+24,0
00 116          B     8+12
00 117          AM    FTEST+11,2,10
00 118          BT    TESTHI,FTEST+11
00 119          BNF   *+36,-FTEST-11
00 120          TF    17992,-FTEST-11
00 121          B     TABLE+288
00 122          AM    FTEST+11,4,10
00 123          BT    TESTHI,FTEST+11
00 124          TF    17996,-FTEST-11
00 125          BNF   *+24,-FTEST-11
00 126          B     TABLE+96
00 127          CM    17983,23,10
00 128          BE    SW
00 129          CM    17983,21,10
00 130          BE    CS-60
00 131          AM    FTEST+11,2,10
00 132          BT    TESTHI,FTEST+11
00 133          BNF   *-24,-FTEST-11
00 134          TF    17998,-FTEST-11
00 135          B     TABLE+288
00 136 * TEST FOR WRAP-AROUND OFF HIGH END OF CORE.
00 137          DC    5,0
00 138 TESTHI   CM    *-1,20000
00 139          BNL   *+24
00 140          BB
00 141          RCTY
00 142          WATY  HIMSG
00 143          RCTY
00 144          H
00 145          B     DSTART
00 146 HIMSG    DAC   47,HI LIMIT OF CORE EXCEEDED. PUSH START TO DUMP.a,
00 147 *
00 148 * TABLE SEARCH FOR OPERATIONAL SUBROUTINE
00 149 *    TABLE ORDER - R,W,P,H,SW,A,S,CS,CW,MCW,C,LCA,B,NOP.
00 150 *
00 151 TABLE    CM    17983,71,10
00 152          BE    R
00 153          CM    17983,72,10
00 154          BE    W
00 155          CM    17983,74,10
00 156          BE    P
00 157          CM    17983,03,10
00 158          BE    H
00 159          CM    17983,23,10
00 160          BE    SW
00 161          CM    17983,41,10
00 162          BE    A
00 163          CM    17983,62,10
00 164          BE    S
00 165          CM    17983,21,10
00 166          BE    CS-84
00 167          CM    17983,04,10
00 168          BE    CW
00 169          CM    17983,54,10
00 170          BE    MCW
00 171          CM    17983,43,10
```

```
00 172          BE    C
00 173          CM    17983,53,10
00 174          BE    LCA
00 175          CM    17982,42,10
00 176          BE    B
00 177          CM    17982,55,10
00 178          BE    NEXTIN
00 179 * INVALID OP CODE ROUTINE
00 180 ERROR1   RCTY
00 181          WATY  OPMSG
00 182          RCTY
00 183          B     CORLIM+36
00 184 OPMSG    DAC   41,INVALID INSTRUCTION. PUSH START TO DUMP.a,
00 185 *
00 186 * OPERATIONAL SUBROUTINES
00 187 *
00 188 * WRITE SUBROUTINE
00 189 W        TFM   *+23,18561
00 190          C     ZEROES-38,0
00 191          BNE   RE
00 192          SM    W+23,2,10
00 193          CM    W+23,18401
00 194          BNE   W+12
00 195          B     SECL
00 196 RE       AM    W+23,2,10
00 197          TD    *+47,-W-23
00 198          TD    -W-23,400
00 199          WATY  18403
00 200          TDM   -W-23,0
00 201 SECL     RCTY
00 202          TD    *+59,18562
00 203          BV    *+12
00 204          SF    18562
00 205          C     18601,ZEROES
00 206          TDM   18562,0
00 207          BNE   *+36
00 208          BV    *+24
00 209          B     B-24
00 210          TD    *+47,18603
00 211          TD    18603,400
00 212          WATY  18563
00 213          TDM   18603,0
00 214          RCTY
00 215          B     B-24
00 216 ZEROES   DC    40,0
00 217 * READ A CARD  SUBROUTINE
00 218 R        RACD  18003
00 219          BNR   B-24,18003
00 220          BNR   B-24,18005
00 221          B     796
00 222 * PUNCH A CARD SUBROUTINE
00 223 P        WACD  18203
00 224          B     B-24
00 225 * HALT SUBROUTINE
00 226 H        BTM   CVTREG,0,10
00 227          TF    *+35,17983
00 228          TF    *+18,IREG-1
```

```
00 229          H     0,0
00 230          B     B-24
00 231 * SET WORD MARK SUBROUTINE
00 232 SW       BTM   CONVTA
00 233          TF    *+30,17989
00 234          SM    *+18,1,10
00 235          SF    0
00 236          BNF   *+24,17990
00 237          B     NEXTIN
00 238          BTM   CONVTB
00 239          TF    *+30,17995
00 240          SM    *+18,1,10
00 241          SF    0
00 242          B     NEXTIN
00 243 * CLEAR WORD MARK SUBROUTINE
00 244 CW       BTM   CONVTA
00 245          TF    *+30,17989
00 246          SM    *+18,1,10
00 247          CF    0
00 248          BNF   *+24,17990
00 249          B     NEXTIN
00 250          BTM   CONVTB
00 251          TF    *+30,17995
00 252          SM    *+18,1,10
00 253          CF    0
00 254          B     NEXTIN
00 255 * MOVE CHARACTER TO A OR B FIELD WORD MARK SUBROUTINE
00 256 MCW      BTM   CONVTA
00 257          TF    MOVE+11,17989
00 258          TF    MOVE+23,17989
00 259          SM    MOVE+23,1,10
00 260          BTM   CONVTB
00 261          TF    MOVE+6,17995
00 262          TF    MOVE+18,17995
00 263          SM    MOVE+18,1,10
00 264          BNF   MOVE,-MOVE-18
00 265          TDM   SFCF+1,2
00 266          TDM   MOVE+25,9
00 267 MOVE     TD    0,0
00 268          TD    0,0
00 269          NOP   SFCF-12
00 270          BNF   SFCF+24,-MOVE-18
00 271          TDM   SFCF+1,3
00 272          TDM   MOVE+25,1
00 273 SFCF     SF    -MOVE-18,0
00 274          B     NEXTIN
00 275          SM    MOVE+6,2,10
00 276          SM    MOVE+11,2,10
00 277          SM    MOVE+18,2,10
00 278          SM    MOVE+23,2,10
00 279          CM    MOVE+18,18000
00 280          BL    CORLIM
00 281          CM    MOVE+23,18000
00 282          BNL   MOVE-36
00 283 CORLIM   RCTY
00 284          BNR   B-24,18003
00 285          B     796
```

```
00 286          WATY CORMSG
00 287          RCTY
00 288          BTM  CVTREG,0,10
00 289          TF   *+35,17983
00 290          TF   *+18,IREG-1
00 291          H    0,0
00 292          B    DSTART
00 293 CORMSG   DAC  48,LOW LIMIT OF CORE EXCEEDED. PUSH START TO DUMP.@,
00 294 * COMPARE SUBROUTINE
00 295 C        BTM  CONVTA
00 296          BTM  CONVTB
00 297          C    -17995,-17989
00 298          BNH  *+36
00 299 HIGH     SF   HIGH
00 300          B    *+24
00 301          CF   HIGH
00 302          BNE  *+36
00 303 EQUAL    SF   EQUAL
00 304          B    *+24
00 305          CF   EQUAL
00 306          B    NEXTIN
00 307 * BRANCH SUBROUTINE
00 308          BNF  B+12,17984
00 309          B    NEXTIN
00 310 B        BNF  DMOD,17990
00 311          BTM  CONVTA
00 312          TF   FTEST+11,17989
00 313          SM   FTEST+11,1,10
00 314          BNF  ERROR1,-FTEST-11
00 315          B    NEXTIN
00 316 DMOD     BNF  BCE,17992
00 317          SF   17990
00 318          CM   17991,21,10
00 319          BE   SLASH
00 320          CM   17991,62,10
00 321          BE   SAME
00 322          CM   17991,63,10
00 323          BE   TINY
00 324          CM   17991,64,10
00 325          BE   UPPER
00 326          B    ERROR1
00 327 SLASH    BNF  B+12,EQUAL
00 328          B    NEXTIN
00 329 SAME     BNF  NEXTIN,EQUAL
00 330          B    B+12
00 331 TINY     BNF  *+24,EQUAL
00 332          B    NEXTIN
00 333          BNF  B+12,HIGH
00 334          B    NEXTIN
00 335 UPPER    BNF  *+24,EQUAL
00 336          B    NEXTIN
00 337          BNF  NEXTIN,HIGH
00 338          B    B+12
00 339 BCE      SF   17996
00 340          BTM  CONVTB
00 341          C    17997,-17995
00 342          BE   B+12
```

```
00 343          B     NEXTIN
00 344 * ADD SUBROUTINE
00 345 A        TFM   ADD+1,21,10
00 346          B     *+24
00 347 * SUBTRACT SUBROUTINE
00 348 S        TFM   ADD+1,22,10
00 349          BTM   CONVTA
00 350          BNF   *+36,17990
00 351          TF    17995,17989
00 352          B     *+24
00 353          BTM   CONVTB
00 354          TFM   STRIPA+6,FIELDA-1
00 355          TF    STRIPA+11,17989
00 356          TF    STRIPA+35,17989
00 357          SM    STRIPA+35,1,10
00 358          TF    TSIGNA+11,STRIPA+35
00 359          TFM   STRIPB+6,FIELDB-1
00 360          TF    STRIPB+11,17995
00 361          TF    STRIPB+35,17995
00 362          SM    STRIPB+35,1,10
00 363          TF    TSIGNB+11,STRIPB+35
00 364          TF    SN-25,STRIPB+35
00 365          TF    SN+6,STRIPB+35
00 366          TFM   SN+47,FIELDB-2
00 367          TF    SN-6,17995
00 368          TF    SN+42,17995
00 369          SM    SN+42,2,10
00 370          TF    SN+59,17995
00 371          SM    SN+59,3,10
00 372          TF    SN+90,SN+59
00 373 TSIGNB   TD    *+22,0
00 374          CM    *+9,5000,8
00 375          BE    STRIPB-12
00 376          C     *+22,-17995
00 377          BE    STRIPB-12,,9
00 378          TDM   FIELDB,0
00 379          B     *+24
00 380          TDM   FIELDB,0,11
00 381 STRIPB   TD    0,0
00 382          SM    STRIPB+6,1,10
00 383          BNF   *+60
00 384          TF    POSCNT,17995
00 385          S     POSCNT,STRIPB+11
00 386          AM    POSCNT,1,10
00 387          B     TSIGNA-12
00 388          SM    STRIPB+11,2,10
00 389          SM    STRIPB+35,2,10
00 390          CM    STRIPB+6,FIELDB-33
00 391          BE    ERROR2
00 392          B     STRIPB
00 393          TDM   -STRIPB-6,0,11
00 394 TSIGNA   TD    *+22,0
00 395          CM    *+9,5000,8
00 396          BE    STRIPA-12
00 397          C     *+22,-17989
00 398          BE    STRIPA-12,,9
00 399          TDM   FIELDA,0
```

```
00 400          B     *+24
00 401          TDM   FIELDA,0,11
00 402 STRIPA   TD    0,0
00 403          SM    POSCNT,1,10
00 404          BNF   *+24
00 405          B     ADD-12
00 406          CM    POSCNT,0
00 407          BE    ADD-12
00 408          SM    STRIPA+6,1,10
00 409          SM    STRIPA+11,2,10
00 410          SM    STRIPA+35,2,10
00 411          B     STRIPA
00 412          SF    -STRIPA-6,0
00 413 ADD      H     FIELDB,FIELDA
00 414          BNF   *+36,FIELDB
00 415          TDM   SN+11,5
00 416          B     *+24
00 417          TDM   SN+11,7
00 418          BNF   *+24,0
00 419          SF    SN+11
00 420          TD    0,FIELDB-1
00 421 SN       TDM   0,0
00 422          BNF   *+24,SN+11
00 423          B     NEXTIN
00 424          TD    0,0
00 425          BNF   *+36,0
00 426          TDM   *+30,7,611,  INDIRECT  ADDRESS  ON  *+30
00 427          B     NEXTIN
00 428          TDM   0,7
00 429          SM    SN+42,2,10
00 430          SM    SN+47,1,10
00 431          SM    SN+59,2,10
00 432          SM    SN+90,2,10
00 433          B     SN+36
00 434 ERROR2   RCTY
00 435          WATY  AMSG
00 436          RCTY
00 437          B     CORLIM+36
00 438 AMSG     DAC   47,B-FIELD OF ADD OR SUB INSTR OVER 32 POSITIONS. ,
00 439          DAC   20,PUSH START TO DUMP.@,
00 440 POSCNT   DC    5,0
00 441 FIELDA   DS    33,
00 442 FIELDB   DS    34,
00 443 * CLEAR STORAGE SUBROUTINE
00 444          BNF   *+24,17990
00 445          B     CS
00 446          BTM   CONVTA
00 447          TF    FTEST+11,17989
00 448          SM    FTEST+11,1,10
00 449          SF    17990
00 450          TF    17989,17995
00 451 CS       TFM   CS+210,18000
00 452          TD    CS+248,17985
00 453          A     CS+208,CS+248
00 454          A     CS+208,CS+248
00 455          TFM   CS+234,18000
00 456          TD    CS+249,17987
```

```
00 457          A     CS+233,CS+249
00 458          A     CS+233,CS+249
00 459          TFM   CS+191,BLANKS-19
00 460          TD    CS+248,17989
00 461          A     CS+191,CS+248
00 462          A     CS+191,CS+248
00 463          TD    CS+248,17987
00 464          BTM   CONVTA
00 465          TF    CS+186,17989
00 466          TF    0,0
00 467          BD    CS+228,CS+248
00 468          CF    0,0
00 469          B     NEXTIN
00 470          TF    0,BLANKS
00 471          SM    *+8,1,710
00 472          SM    CS+234,20,10
00 473          B     CS+192
00 474 * LOAD CHARACTERS TO A-FIELD WORD MARK SUBROUTINE
00 475 LCA      BTM   CONVTA,0
00 476          BTM   CONVTB
00 477          TF    -17995,-17989
00 478          B     NEXTIN
00 479 * CONVERT A SUBROUTINE TO CONVERT FROM 141 TO 1620 ADDRESSING
00 480          DC    5,0
00 481 CONVTA   TD    17988,17987
00 482          TD    17987,17985
00 483          TFM   17986,0,10
00 484          A     17989,17989
00 485          AM    17989,18001
00 486          BB
00 487 * CONVERT B SUBROUTINE TO CONVERT FROM 141 TO 1620 ADDRESSING
00 488          DC    5,0
00 489 CONVTB   TD    17994,17993
00 490          TD    17993,17991
00 491          TFM   17992,0,10
00 492          A     17995,17995
00 493          AM    17995,18001
00 494          BB
00 495 *
00 496 * CLEAR ROUTINE
00 497 *
00 498 CLEAR    RCTY
00 499          RCTY
00 500          TFM   CLEAR+42,19999
00 501          TF    19999,BLANKS,2
00 502          SM    CLEAR+42,20,10
00 503          CM    CLEAR+42,17999
00 504          BNE   CLEAR+36
00 505 PRELD    B     INITZR,,0
00 506 BLANKS   DC    21,0
00 507 *
00 508 * DUMP ROUTINE
00 509 *
00 510 DSTART   RCTY
00 511          RCTY
00 512          TF    OPREG+10,17983
00 513          BTM   CVTREG,0,10
```

```
00 514          RCTY
00 515          WATY  TITLE
00 516          RCTY
00 517          SPTY
00 518          WNTY  IREG-3
00 519          WATY  OPREG
00 520          RCTY
00 521          RCTY
00 522          CF    BLNKS-49
00 523          CF    BLNKS-99
00 524          TFM   CARDNO,0,10
00 525          TFM   ADDR1,0,9
00 526          TFM   ADDR2,49,9
00 527          TFM   SAVC+11,18101
00 528          TFM   INSRM+6,18101
00 529          TFM   IN+18,18101
00 530          TFM   INSRM+23,18000
00 531 SAVC     TD    IN+23,0
00 532 INSRM    TD    0,400
00 533          TR    BANDA+37,0
00 534          AM    CARDNO,01,10
00 535          TD    BANDA,CARDNO-1
00 536          TD    BANDA+2,CARDNO
00 537          AM    CARDNO,01,10
00 538          TD    BANDB,CARDNO-1
00 539          TD    BANDB+2,CARDNO
00 540          TDM   BANDA+137,0
00 541          TD    BANDA+18,ADDR1-1
00 542          TD    BANDA+16,ADDR1-2
00 543          TD    BANDA+30,ADDR2-1
00 544          TD    BANDA+28,ADDR2-2
00 545          BNC2  *+48
00 546          TDM   BANDA+138,0
00 547          WACD  BANDA
00 548          B     PWM
00 549          BNC1  *+24
00 550          H
00 551 TYPE     TFM   *+23,BANDA+136
00 552          C     ZEROES-38,0
00 553          BNE   *+36
00 554          SM    TYPE+23,2,10
00 555          B     TYPE+12
00 556          AM    TYPE+23,2,10
00 557          TD    -TYPE-23,400
00 558          WATY  BANDA+16
00 559          TDM   -TYPE-23,0
00 560          RCTY
00 561 PWM      TF    BANDB+138,BLNKS
00 562          TFM   TEST5+11,BANDA+37
00 563          TFM   TEST5+18,BANDB+38
00 564          TFM   INSRM2+6,BANDB+16
00 565 TEST5    BNF   INCR,0,27
00 566          TFM   0,71,10
00 567          TF    INSRM2+6,*-6
00 568 INCR     AM    TEST5+11,2,10
00 569          AM    TEST5+18,2
00 570          CM    TEST5+11,BANDA+137
```

```
00 571          BNE   TEST5
00 572 WRITE    BNC2  *+120
00 573          CM    CARDNO,40,10
00 574          BNE   *+60
00 575          TFM   BANDB+9,70707
00 576          TD    BANDB+10,IREG-1
00 577          TD    BANDB+8,IREG-2
00 578          TD    BANDB+6,IREG-3
00 579          WACD  BANDB
00 580          TF    BANDB+10,ZEROES-34
00 581          B     *+60
00 582          AM    INSRM2+6,2,10
00 583 INSRM2   TD    0,400
00 584          WATY  BANDB+16
00 585          RCTY
00 586          BD    OUT,SWENDD
00 587 IN       AM    SAVC+11,100,9
00 588          TDM   0,0
00 589          TF    INSRM+6,SAVC+11
00 590          TF    IN+18,SAVC+11
00 591          AM    INSRM+23,100,9
00 592          AM    ADDR1,50,10
00 593          AM    ADDR2,50,10
00 594          CM    SAVC+11,20001
00 595          BNE   SAVC
00 596          TD    1,400
00 597          TDM   SWENDD,1
00 598          TR    BANDA+37,19900
00 599          B     INSRM+24
00 600 OUT      TDM   1,9
00 601          TDM   SWENDD,0
00 602          B     INITZR
00 603 BANDA    DAC   50,01       000 - 049
00 604          DAC   30,
00 605 BANDB    DAC   50,02
00 606          DAC   30,
00 607          DC    22,0
00 608          DC    50,0
00 609 BLNKS    DC    50,0
00 610 SWENDD   DC    1,0
00 611 ADDR1    DC    3,0
00 612 ADDR2    DC    3,49
00 613 CARDNO   DC    2,0
00 614 * PRINT REGISTERS SUBROUTINE
00 615 TITLE    DAC   14,I-REG  OP-REG@,
00 616 IREG     DC    6,@,
00 617 OPREG    DAC   7,      @,
00 618 DIV      DC    6,0
00 619          DC    5,0
00 620 CVTREG   TF    IREG-1,FTEST+11
00 621          SM    IREG-1,18000
00 622          TF    DIV-1,IREG-1
00 623          S     DIV,IREG-1
00 624          S     DIV,IREG-1
00 625          S     DIV,IREG-1
00 626          S     DIV,IREG-1
00 627          S     DIV,IREG-1
```

```
00 628          TF    IREG-1,DIV-1
00 629          BB
00 630 *
00 631 * ALTER ROUNTINE AND
00 632 * LOAD DUMP CARDS ROUTINE
00 633 *
00 634 ALTER    WATY  BG,SC
00 635          RNTY  TESTL-1
00 636          SF    TESTL-1
00 637          TF    FIRST+2,TESTL+1
00 638          TDM   ALTSW,1
00 639 NEXTL    RCTY
00 640          RCTY
00 641          CF    FIRST
00 642          WNTY  FIRST
00 643          SF    FIRST
00 644          RCTY
00 645          TFM   READ1+6,18001
00 646          A     READ1+6,FIRST+2
00 647          A     READ1+6,FIRST+2
00 648 READ1    RATY  0
00 649          RCTY
00 650          RNTY  WMS+19
00 651          TF    STFLG+6,READ1+6
00 652          SM    STFLG+6,1,10
00 653 COMMON   TFM   TDIG+11,WMS+19
00 654          TFM   TRM+11,WMS+19
00 655 TRM      BNR   *+24,0
00 656          B     RM
00 657 TDIG     BD    *+36,0
00 658          TDM   STFLG+1,3
00 659          B     *+24
00 660          TDM   STFLG+1,2
00 661 STFLG    SF    0
00 662          AM    STFLG+6,2,10
00 663          AM    TDIG+11,1,10
00 664          AM    TRM+11,1,10
00 665          CM    TDIG+11,WMS+119
00 666          BNE   TRM
00 667 RM       SM    TDIG+11,WMS+19
00 668          SF    TDIG+9
00 669          A     FIRST+2,TDIG+11
00 670          BD    NEXTL,ALTSW
00 671          BD    EXEC,WMS+3
00 672 LDUMP    SF    BANDC+16
00 673          RACD  BANDC
00 674          TD    BANDC+138,400
00 675          BD    *+24,BANDC+19
00 676          B     CDERR
00 677          TD    FIRST+2,BANDC+20
00 678          TD    FIRST+1,BANDC+18
00 679          TD    FIRST,BANDC+16
00 680          TFM   TR+6,18000
00 681          A     TR+6,FIRST+2
00 682          A     TR+6,FIRST+2
00 683 TR       TR    0,BANDC+37
00 684          RNCD  WMS
```

```
00 685          TD    WMS+69,400
00 686          BD    CDERR,WMS+16
00 687          TF    STFLG+6,TR+6
00 688          SF    WMS
00 689          CM    WMS+1,40,10
00 690          BE    *+48
00 691          TF    *+30,TR+6
00 692          AM    *+18,101,9
00 693          TDM   0,0
00 694          TDM   ALTSW,0
00 695          B     COMMON
00 696 EXEC     TD    17985,WMS+3
00 697          TD    17987,WMS+4
00 698          TD    17989,WMS+5
00 699          SF    17990
00 700          TFM   1,49,10
00 701          B     B
00 702 CDERR    WATY  CDMSG
00 703          RCTY
00 704          H
00 705          B     START
00 706 ALTSW    DC    1,0
00 707 CDMSG    DAC   38,SEQUENCE ERROR. PUSH START TO RE-LOAD@,
00 708 FIRST    DSC   4,000@,
00 709 WMS      DSS   120
00 710 BANDC    DAC   50,
00 711          DAC   30,
00 712          DEND  BEGIN
```

SYMBOL TABLE
141SIM-C

ZEROES	05501	TSIGNB	07338	TSIGNA	07590	TESTHI	04482	SWENDD	10564
STRIPB	07434	STRIPA	07686	POSCNT	08256	NEXTIN	03984	INSRM2	09882
INITZR	02654	INBRCH	02906	FIELDB	08323	FIELDA	08289	ERROR2	08070
ERROR1	05008	DSTART	09006	CVTREG	10634	CORMSG	06343	CORLIM	06222
CONVTB	08816	CONVTA	08738	COMMON	10982	CARDNO	10572	BLANKS	09004
A	07014	ADD	07818	ADDR1	10567	ADDR2	10570	ALTER	10754
ALTSW	11618	AMSG	08119	ASK	03693	B	06606	BANDA	10123
BANDB	10283	BANDC	11821	BCE	06954	BEGIN	02402	BGMSG	03739
BLNKS	10563	C	06438	CDERR	11570	CDMSG	11621	CLEAR	08888
CS	08408	CW	05766	DIV	10627	DMOD	06678	EQUAL	06534
EXEC	11498	FIRST	11696	FTEST	04068	FUNCT	03143	H	05574
HEADG	03035	HIGH	06486	HIMSG	04579	IN	09930	INCR	09702
INERR	03769	INSRM	09270	IREG	10607	LCA	08684	LDUMP	11210
MCW	05898	MOVE	06030	NEXTL	10814	OPMSG	05057	OPREG	10609
OUT	10086	P	05550	PRELD	08972	PWM	09618	R	05502
RE	05222	READ1	10922	RM	11150	S	07038	SAME	06834
SAVC	09258	SECL	05282	SFCF	06102	SLASH	06810	SN	07914
START	03816	STFLG	11078	SW	05634	TABLE	04672	TDIG	11030
TESTL	03015	TEST5	09666	TINY	06858	TITLE	10575	TR	11342
TRM	11006	TYPE	09498	UPPER	06906	W	05138	WMS	11700
WORD	03623	WRITE	09750	WRT	02570				

DR. JOHN MANIOTES
COMPUTER TECHNOLOGY DEPT.
PURDUE UNIVERSITY
CALUMET CAMPUS
HAMMOND, IN 46323

THE 141 DATA PROCESSING SYSTEM

Description

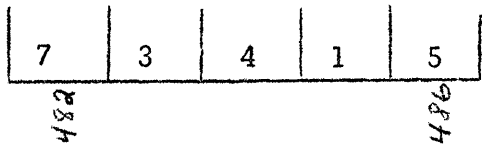
The 141 Data Processing System is an abbreviated version of the IBM 1401 Data Processing System. It is an internally stored program machine with the following features:

1. Input: IBM Card Reader
2. Output: IBM Card Punch and 100 character per line Printer
3. Storage: 1000 positions of core storage with three digit numerical addresses
4. Instruction length and date length: Variable.

Each position is designated by a three digit address in the range of 000 through 999 and is capable of storing one character: ≠ a letter of the alphabet, a numeric digit or a special character such as . , / or ¤ . A group of consecutive storage positions make up a field. Both data and instruction fields are variable in length so that no storage space need be wasted with meaningless blanks or zeros.

Data

If a field is used for data it is referred to by the address of its low-order (rightmost) digit position. A special indicator called a word mark is placed in the high-order (leftmost) digit position to indicate the length of the field. The machine reads a data field from right to left starting at the addressed low-order position and continuing until a character with a word mark is met. For example, a 5 digit data field in storage locations 482 through 486 would have a word mark (indicated by an underline) in position 482 and would be addressed by 486.



Instructions

When a field contains an instruction it is addressed by the high-order position (left-most position) of the field. This position contains the operation code character and a word mark. In addition to the operation code character, an instruction may also contain one or two 3-digit addresses and/or a modifying character. An instruction may therefore contain 1,4,5, or 7 characters. The machine reads instructions from left to right starting with the addressed high-order position and continuing until it meets the word mark in the high-order position of the next instruction.

Character Coding

Each position of storage is made up of eight ferromagnetic cores - each of which can hold one "yes-or-no" bit. Four bits are needed to represent the digit portion of the Hollerith code of the characters; two bits are needed to represent the zone portion of the Hollerith code of the characters; one bit is needed to indicate the presence or absence of a word mark; and one bit is used for checking.

The digit portion is coded in the Binary Code Decimal (BCD) system in which the four bits have the values of 1,2,4, and 8 respectively. The sum of the "yes" bits is equal to the value of the digit.

Digit	8	4	2	1
1				X
2			X	
3			X	X
4		X		
5		X		X
6		X	X	
7		X	X	X
8	X			
9	X			
0	X		X	

(1)

The 0 digit is represented by the 8 bit and the 2 bit rather than no bits at all. In representing alphabetic and special characters the zone is represented by the A and the B bits as follows:

Zone	B	A
12 (A-I)	X	X
11 (J-R)	X	
0 (S-Z)		X
no (0-9)		

The C bit is used for parity checking and is chosen so that the character will contain the odd number of "yes" bits. For example the letter C which is given by the 12 and 3 punches on the IBM is represented in the seven-bit alphameric code as:

		Seven-bit alphameric code						
Character	Card Code	C	B	A	8	4	2	1
C	12-3	X	X	X			X	X

TABLE I

Character	Card Code	Seven-bit Alphameric Code C*	B	A	8	4	2	1
Blank	Blank	X						
.	12 - 3 - 8		X	X	X		X	X
¤	12 - 4 - 8	X	X	X	X	X		
&	12	X	X	X				
-	11		X					
/	0 - 1	X		X				X
,	0 - 3 - 8	X		X	X		X	X
A	12 - 1		X	X				X
B	12 - 2		X	X			X	
C	12 - 3	X	X	X			X	X
D	12 - 4		X	X		X		
E	12 - 5	X	X	X		X		X
F	12 - 6	X	X	X		X	X	
G	12 - 7		X	X		X	X	X
H	12 - 8		X	X	X			
I	12 - 9	X	X	X	X			X
J	11 - 1	X	X					X
K	11 - 2	X	X				X	
L	11 - 3		X				X	X
M	11 - 4	X	X			X		
N	11 - 5		X			X		X
O	11 - 6		X			X	X	
P	11 - 7	X	X			X	X	X
Q	11 - 8	X	X		X			
R	11 - 9		X		X			X
S	0 - 2	X		X			X	
T	0 - 3			X			X	X
U	0 - 4	X		X		X		
V	0 - 5			X		X		X
W	0 - 6			X		X	X	
X	0 - 7	X		X		X	X	X
Y	0 - 8	X		X	X			
Z	0 - 9			X	X			X
0	0	X			X		X	
1	1							X
2	2						X	
3	3	X					X	X
4	4					X		
5	5	X				X		X
6	6	X				X	X	
7	7					X	X	X
8	8				X			
9	9	X			X			X

*Check bit to produce odd-parity. Table shows values for positions with no word mark. Reverse if word mark is present.

Operation Codes

SW Set Word Mark ,aaa
The Set Word Mark instruction causes a word mark to be set at the A-address. Data at this address is undisturbed.

CW Clear Word Mark ⌐aaa
The Clear Word Mark instruction causes the word mark at the A-address to be cleared. Data at the A-address is not disturbed.

MCW Move Characters to A or B Word Mark instruction causes the data in the A-field to be moved to the B-field. The first word mark encountered stops the operation. The data at the A-address is unaffected by the move operation. Word Marks in both fields are undisturbed.

R Read a Card 1
The Read a Card instruction causes the information in columns 1 through 80 of an IBM card to be read into storage locations 001 through 080 respectively. The Hollerith code of each column is converted into six-bit alphameric code as it is read into its position of core storage, according to Table 1. Word marks are undisturbed. The C check bit of each position is automatically set to produce an odd parity.

P Punch a Card 4
The Punch a Card instruction causes the information in storage 101 through 180 to be punched into columns 1 through 80, respectively, of an IBM card. The six-bit alphameric code is converted into Hollerith code according to Table 1. The information is location 101 through 180, including word marks, is undisturbed.

W Write a Line 2
The Write a Line instruction causes the information in storage locations of 201 through 300 to print in the one hundred print positions of one line on the printer. The information in storage, including word marks, is undisturbed. The printer takes one automatic space after printing a line.

A Add A aaa bbb
The add instruction causes the data in the A-field to be added algebraically to the data in the B-field. The result is stored in the B-field. The defining word mark of the B-field stops the operation. If the A-field is shorter than the B-field, a word mark in the A-field will stop the transmission of data from the A-field, but any resulting carries will be added to the contents of the B-field until the word mark in the B-field is met.

S Subtract S aaa bbb
The subtract instruction causes the data in the A-field to be subtracted algebraically from the data in the B-field. Word marks control the operation in the same manner as in the Add operation.

C Compare C aaa bbb
The Compare instruction causes the information in fields A and B to be compared character to character. The six-bit configuration of each character in the two fields is compared. Word marks and check bits do not enter into the comparison. If two corresponding characters are not the same, the Unequal Compare Indicator is turned on; in addition, either the B Field High or B Field Low Indicators would be turned on according to the results of the comparison. Each of these indicators may be tested by a Branch if Indicator On instruction. The indicators are not turned off until the next Compare instruction. If all indicators are off after a Compare instruction has been executed, an equal condition may also be tested by a Branch if Indicator On instruction.

The first word mark encountered stops the operation. If the A-field is longer than the B-field, extra A-field positions beyond the length of the B-field will not be compared. If the B-field is no longer than the A-field, an unequal compare results and the Unequal and B-Field High Indicators are turned on.

B Branch B iii

The Branch instruction causes the program to branch to the instruction specified by the I-address.

B Branch if Indicator On B iii d

In the Branch If Indicator On instruction the d-character specifies the indicator tested. If the indicator is on the next instruction is taken from the I-address. If the indicator is off, the next sequential instruction is taken.

Indicator		d-Character	
Unequal Compare	(B≠A)	1	≠
Equal Compare	(B=A)	S	Same
Low Compare	(B<A)	T	Top
High Compare	(B>A)	U	Upper

H Halt •

The Halt instruction causes the program to start with the next instruction in sequence.

SECTION 2

141 Symbolic Programming System

The Symbolic Programming System, SPS, has been developed to facilitate logical, efficient programming with a minumum of actual coding effort. It almost completely relieves the programmer of the task of assigning actual storage location to the instructions and data of the program and allows him to refer to them by easy-to-remember names of his choice.

Instructions

Instructions written in SPS contain a Label, an operation code, an A-Operand, a B-operand, and a d-character. Any of the parts except the operation code may be left blank.

The Label is the symbolic representation of the location in memory that the instruction will be stored. It may have from one to six alphameric characters, the first of which must be alphabetic and must be placed in column 8 of the coding sheet. The Label may be left blank if no reference is made to the instruction in the rest of the program.

The Operation Code may be either Mnemonic or the Machine-language code. Mnemonic c odes are used from one to three characters starting in column 14. If the machine-language code is used, it must be placed in column 16.

If the instruction requires an A-Operand it is written in column 17 through 26. If no A-Operand is used, those columns may be left blank. The address of an operand may be expressed symbolically or actually. If it is written symbolically, it takes the same form as a Label. If it is written as an actual location, it must be a four-digit number with leading zeros where necessary. Although four digits are written on the coding sheet, only three characters will be used in memory.

Adjusted Address

The address of an operand may be adjusted by placing the number of characters of adjustment in columns 24 through 26 and the sign of the adjustment must be in column 23. Leading zeros may be omittedbut the units digit of the adjustment must be in column 26.

The indexing character (column 27) is not used with the 141.

If the instruction requires a B-Operand, its address is written in the same manner as the A-Operand. When an instruction requires a d-character, the actual machine code is placed in column 39.

DCW - Define Constant with Word Mark

The symbolic operation code DCW causes a constant to be loaded into the area specified and sets a word mark in the high-order position of the field. The nu mber of characters in the constant field is placed in the Count portion of the coding sheet (columns 6 and 7). The symbolic label by which the constant is referred is placed in columns 8-13. The code DCW is placed in columns 14-16. Columns 17-20 must contain the actual location of the units position of the field, or an asterisk must be placed in column 17 to indicate the SPS program is to choose the location of the field. The constant itself begins in column 24 and may extend through column 55, a maximum of 32 characters. If the constant is to be signed numeric constant, the sign may be placed in column 23.

End

The special symbols END placed in columns 14-15 of the coding sheet signifies to the SPS program that it is the last line of the program. In the A-operand field of this line must be placed either the symbolic or actual address of the first executable instruction of the program. The additional purpose of the END card is to provide a branch to the beginning of the object program at the end of the loading.

SECTION 3

Exercises

Exercise 1

Write a program that will reproduce a card, that is, will read a card and punch a card identical to the one read.

Exercise 2

Write a program that will read a card and punch a card with the information from columns 1-40 of the card read in columns 41-80 of the card punched and the information from columns 41-80 of the card read in columns 1-40 of the card punched.

Exercise 3

Write a program that will reproduce an entire deck of cards.

Exercise 4

Write a program that will read one card and will punch copy after copy of it until the machine is stopped by the operator.

Exercise 5

Write a program that will print a directory of telephone extensions from a deck of personnel cards. The cards and directory forms are as follows:

Card Columns	Field	Print Positions
1 - 18	Name	1 - 18
19	First Initial	20
20	Second Initial	22
21 - 60	Not used in this program	
61 - 64	Telephone Extension	28 - 31
65 - 80	Not used in this program	

Exercise 6

Write a program that will read cards containing numeric fields A, B, and C and will punch corresponding cards that contain fields A, B, C, and D, where D = A + B - C. The card columns are as follows:

Field	Card Columns	Card
A	1 - 6	Input and Output
B	7 - 11	Input and Output
C	12 - 14	Input and Output
D	75 - 80	Output only

Assume that no overflows will occur.

Exercise 7

Write a program that will check the sequence of employee numbers found in columns 75 - 80 of a deck of cards. The program should stop the machine if it finds any employee number that is not larger than the one in the previous card.

Exercise 8

Write a program that will punch consecutive numbers 001 through 015 in columns 78-80 of the first 15 blank cards in the punch hopper and stop automatically before punching a sixteenth card.

Exercise 9

Write a program that will calculate and punch D, where D = A + B - C. Provide for decimal alignment, rounding (half-adjustment), and overflow. The card columns and decimal form of each field is as follows:

Input Card		
A	Col. 5-8	XXX.X
B	9-12	XX.XX
C	13-14	XX.
Output Card		
D	Col. 7-10	XXXX.

Exercise 10

Write a program that will up-date a customer's charge account after a new purchase has been recorded. A new balance card is to be punched and a listing of each customer's name, new balance, and limit is to be printed. If the new balance exceeds the customer's limit the words OVER LIMIT are also to be printed on his entry. The card columns and print positions are as follows:

Field	Input Card	Output Card	Listing
Name	1-20	1-20	11-30
Balance	21-30	21-30	35-44
Charge	31-40	---	---
Limit	71-80	71-80	49-58
OVER LIMIT	---	---	63-72

SECTION 6

141 Simulator Operating Procedures

Adjust the Typewriter

a) Set left margin at 10
b) Set right margin at 94
c) Set a tab stop at 65 (clearing any other tab)

Set Console Switches

a) Set parity switch to STOP
b) Set I/O switch to STOP
c) Set O'Flow switch to PROGRAM
d) Program switches #1 and #2 will normally be OFF. Uses of these switches will be explained later. Program Switches #3 and #4 are not used.

Load Simulator

a) Place 141 Simulator deck in the hopper in the 9-edge face-down position.
b) With the machine in Manual press the Load button[1].
c) When the simulator is loaded the typewriter will automatically begin typing. It may be necessary to press the Reader Start button to enter the last two cards.

Select the Desired Function

a) The typewriter will type a list of the five functions that the simulator will perform and the five request words that will initiate these functions.

FUNCTIONS PERFORMED	REQUEST BY TYPING
LOAD PROGRAM FROM CARD READER	LOAD
CLEAR 141 MEMORY	CLEAR
ALTER MEMORY FROM TYPEWRITER	ALTER
DUMP CONTENTS OF 141 MEMORY	DUMP
BEGIN EXECUTION OF PROGRAM	EXECUTE

b) The typewriter will type the words REQUESTED FUNCTION IS and then stop.
c) The operator then types the word LOAD, CLEAR, ALTER, DUMP or EXECUTE and presses the RELEASE and START buttons on the console or the RS key on the typewriter.
d) If a function runs to completion the simulator will automatically request the next function. If the function is interrupted by pressing INSTANT STOP the operator may return to the request statement by pressing, in order, the RESET, INSERT, RELEASE, and START buttons on the console.

The LOAD function

Programs that have been assembled by SPS can be loaded with this function.
a) Place the SPS object deck, including the two clear storage cards and the bootstrap card, in the hopper.
b) Type the request word LOAD and pressthe RELEASE and START buttons.
c) Press READER START, if necessary.
d) When the program is loaded the typewriter will type PROGRAM LOADED. PUSH START TO EXECUTE.
e) Press the START button on the console to begin execution of the program.

The CLEAR function

The 141 memory can be cleared (set to blanks) with this function.
a) Type the request word CLEAR.
b) Press the RELEASE and START buttons
c) When the clearing operation is completed the typewriter will request the next function.

1. Clearing the 1620 memory before loading is unnecessary, since a clear routine is built into the simulator deck.

The ALTER function

Instruction and data, including word marks, in the 141 memory can be altered with this function. This may be used for debugging a program or entering complete small demonstration programs directly in machine language.

a) Type the request word ALTER.
b) Press the RELEASE and START buttons.
c) The typewriter will type BEGINNING AT.
d) Type the three digit 141 location at which the alteration is desired (such as 333).
e) The typewriter will repeat this location to verify it.
f) Type the instructions and data in machine language, disregarding word marks.[2]
g) At any convenient place, at least one character before the end of the typewriter line, cease typing and press the RELEASE and START buttons.
h) The typewriter carriage will return for a second line. This line will indicate the presence or absence of word marks. If the character above requires a word mark type a 1, if it should not have a word mark hit the space bar. Continue to type 1's and spaces until the carriage has moved across the entire line above. In the first character after the line type a record mark ≠ , then press the RELEASE and START buttons.
i) The typewriter will now type the address of the next memory location to be altered if the process (f thru i) is continued.
j) When altering is completed press, in order, the RESET, INSERT, RELEASE and START buttons. The EXECUTE function can be used to start the program.

SECTION 7

141 - SPS Assembler Operating Procedures

Adjust the typewriter

a) Set left margin at 10
b) Set right margin at 94

Set Console Switches

a) Set parity switch to STOP
b) Set I/O switch to STOP
c) Set O'Flow switch to Program
d) Program Switch #2 will normally be off. Program switches 1,3, and 4 are not used.

Assemble SPS Programs

a) Place the 141 - SPS ASSEMBLER deck in the reader hopper in the 9-edge face down position.
b) Place the SPS source program decks in the reader hopper. Any number of programs may be "stacked" for assembly. The last card of each deck must be an END statement.
c) With the machine in MANUAL press the LOAD button[3].
d) Press the PUNCH START button.
e) It will be necessary to press the READER START button to enter the last two cards of the last deck.

2. This is the only instance where the operator will have to use the typewriter shift key. For all other entires the typewriter will automatically be in the proper alphabetic or numeric shift.
3. Clearing the 1620 memory before loading is unnecessary, since a clear routine is built into the assembler deck.

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

IBM 1401 SYMBOLIC PROGRAMMING SYSTEM

CODING SHEET

FORM X24-1152-3
PRINTED IN U.S.A.

Program _____

Programmed by _____

Date _____

Page No. [1 2] of _____

Identification [76 80]

LINE 3 5	COUNT 6 7	LABEL 8 13	OPERATION 14 16	(A) OPERAND ADDRESS 17	(A) OPERAND ± 23	(A) OPERAND CHAR. ADJ.	(A) OPERAND IND. 27	(B) OPERAND ADDRESS 28	(B) OPERAND ± 34	(B) OPERAND CHAR. ADJ.	(B) OPERAND IND. 38	d 39	COMMENTS 40 55
010													
020													
030													
040													
050													
060													
070													
080													
090													
100													
110													
120													
130													
140													
150													
160													
170													
180													
190													
200													

AREA-DEFINITION CHARACTER COUNT → 1 5 10 15 20 25 30 32

6402250MSP

Use of Program Switch #2

If Program Switch #2 is turned ON the typing of the program listing will be eliminated. A program listing can be obtained by listing the object deck on the IBM 407. This alternative will greatly reduce the assembly time.

SECTION 8

Listing Panel for IBM 407

An IBM 407 Accounting Machine control panel for use with the 141 is diagrammed on the following pages. Three separate functions have been wired into the board, the choice of which is under the control of Alteration Switches #1 and #3, as follows:

FUNCTION	TRANSFERRED ALTERATION SWITCHES
80 - 80 List	1
DUMP List	1, 3
SPS Post List	None

In each case, single spacing occurs with Alteration Switch #2 normal and double spacing occurs with it transferred.

The 80-80 list option is useful for printing the results of 141 programs that have been punched on cards. Since the punch is very much faster than the simulated printer, teachers with large classes of students will wish them to use punched card output in preference to printer output whenever possible.

The DUMP list option is designed to list the cards that result from dumping the 141 memory with Program Switch #1 on. This is another way of increasing the efficiency of operations.

The SPS list option will list the output cards from an SPS assembly. Both the high-order and low-order addresses of each instruction are printed. The clear and bootstrap cards may be left on the deck but they will not print.

If more than one SPS program is to be listed the END card will cause a skip to channel 1 between programs.

The required 407 machine configuration is:

Ten pilot selectors
Eight co-selectors
One digit selector

IBM CARD CODING

The IBM card provides 80 vertical columns with twelve punching positions in each column. One or more punches in a single column represents a character. A group of columns used for the characters that make up a quantity, or a name, is called a field and is often indicated on the card by printed vertical lines.

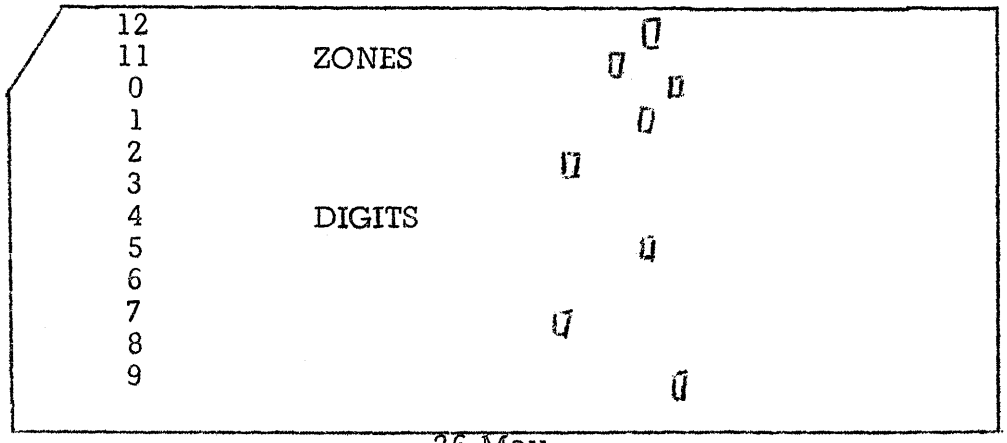
The standard IBM card code, often referred to as the Hollerith code, uses the twelve possible punching positions of a vertical column on the card to represent a numeric, alphabetic or special character. The twelve punching positions are divided into two groups: digits and zones. The first ten punching positions from the bottom edge of the card are the digit punching positions and have assigned numeric values of 9, 8, 7, 6, 5, 4, 3, 2, 1 and 0, respectively. The first three punching positions from the top of the card are the zone punching positions and have zone values of 12, 11 and 0, respectively. The 0 position is considered to be both a digit and a zone position.

The numeric characters 0 through 9 are represented by a single punch in a column. For example, 5 is represented by a single punch in the 5 position of the column.

Alphabetic characters are represented by two punches in a single column: one digit punch and one zone punch. The alphabetic characters A through I use the 12 zone punch and a digit punch 1 through 9 respectively; J through R use the 11 zone punch and a digit punch 1 through 9 respectively; and S through Z use the 0 zone punch and a digit punch 2 through 9 respectively.

Special characters are represented by one, two, or three punches in a single column in combinations that are not used for either numeric or alphabetic characters.

For example: '26May' would be represented as follows:



26 May

The 12 and 11 zone punches when used by themselves have several different meanings depending upon their usage in the specific problem. They may represent an and sign (&) and a hyphen (-), respectively, when used in an alphabetic field. They may also represent a plus (+) sign and a minus (-), respectively, when used as the algebraic sign of a numeric field. In this case they are usually punched in the same column as the units position digit. Still another use of these punches, especially the 11 punch, is that of a control punch to signify that a particular card is an exception of some kind. When used this way the 11 punch is usually referred to as an X punch and is placed in some pre-assigned column.

The following is a table of the IBM card code for numeric and alphabetic characters.

		ZONE			
		No	12	11	0
	0	0			
D	1	1	A	J	
I	2	2	B	K	S
G	3	3	C	L	T
I	4	4	D	M	U
T	5	5	E	N	V
	6	6	F	O	W
	7	7	G	P	X
	8	8	H	Q	Y
	9	9	I	R	Z

Number Systems

Almost all components used in building a ditital computer are best used as bi-state elements; that is, devices that have two and only two distinct conditions. Examples of this are: a switch that is either open or closed, a light that is either on or off, a punch position in an IBM card that is either punched or not, a vacuum tube that is conducting or cut off, a relay that is transferred or normal, and a magnetic core that has polarity of N-S or S-N. The use of bi-state elements encourages the use of a number system that has on ly two digits instead of the usual ten. These digits would be 0, usually corresponding to the off status of an element, and 1, corresponding to its on status. Such a number system is known as the binary system. It is probably easiest to understand a new number system by comparing it to the familiar decimal system.

There are three essential parts to any system. The first is the base of the system. In the decimal system the base is 10; in the binary system it is 2. The second essential part is a set of symbols or digits. In base 10 these digits are 0, 1, 2, 3, 4,5,6,7,8, and 9. In base 2 these symbols, often called bits (a contraction of binary and digits) are 0 and 1. Note that the number of digits in a system is equal to the base number and ranges from 9 to one less than the base number. The third essential part is place value. That is, the value of any digit depends upon its position within the number. Starting at the decimal point (or binary point) and proceeding to the left, the place values are always 1, the base, the base squared, the base cubed, etc. In base ten the place values are 1, 10, 100, 1000, etc. In base two they are 1, 2, 4, 8, etc.

The meaning of a number comes from combining these essential parts of its number system. Therefore, the number 5280 in the decimal system means

$$5280 = 5 \times 1000 + 2 \times 100 + 8 \times 10 + 0 \times 1$$

while the number 1101 in the binary system means

$$1101_2 = 1 \times 8 + 1 \times 4 + 0 \times 2 + 1 \times 1$$

or decimally 13. In the binary system, this reduces to the simple rule that "the value of a binary number is equal to the sum of the place values containing 1's". In the above case $1101_2 = 8 + 4 + 1 = 13$

The decimal equivalent of the first 17 binary integers is as follows:

Decimal	Binary	Decimal	Binary
0	0	8	1000
1	1	9	1001
2	10	10	1010
3	11	11	1011
4	100	12	1100
5	101	13	1101
6	110	14	1110
7	111	15	1111
		16	10000

RESTRICTIONS

The following restrictions pertain to the 141 S imulator program

1. The twelve executable instructions are: SW, CW, MCW, R, P, W, A, S, C, B, Bi, and H. The Branch if Indicator On instruction includes four d-modifiers: /, S, T, U.

2. One thousand positions of 141 memory are available, with addresses ranging from 000 to 999.

3. All instructions must be followed by a word mark.

4. No overflow arithmetic is provided. Carries out of the high-order of the B-field will be lost. They will not become zone bits.

5. In arithmetic operations zones are stripped from all B-field characters including the high-order character.

The following restrictions pertain to the 141 Assembly program.

1. Only the above-mentioned twelve operation codes and four d-modifiers can be assembled.

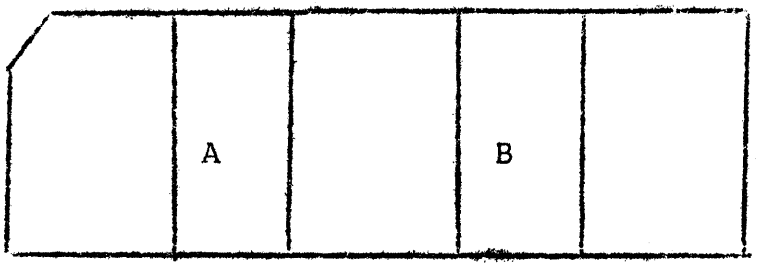
2. The DCW is the only declarative operation and END is the only processor operation recognized by the assembly program. All programs will automatically begin at location 333.

3. The maximum number of cards in any 141 SPS program is 100. Images of these are held in memory to reduce the amount of card handling and to permit stacking of programs.

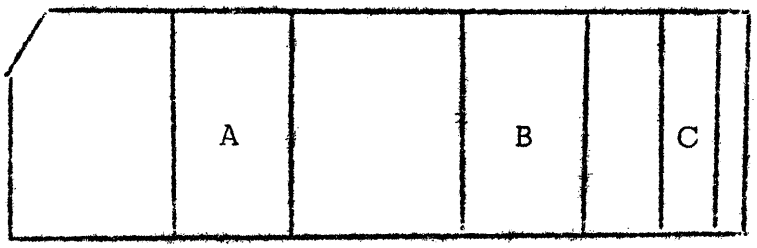
Problem #1

Calculate and punch C where C = A + B
A, B and C are all 5 column fields

Input Card



Output Card

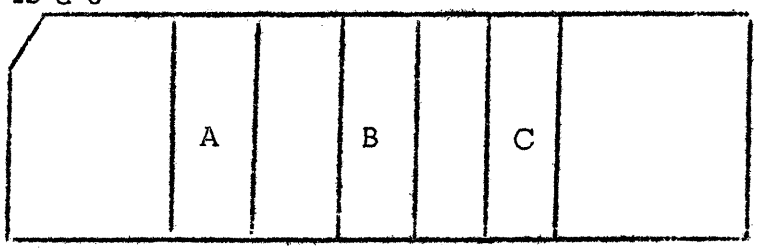


Input Cards 3
Output cards 3

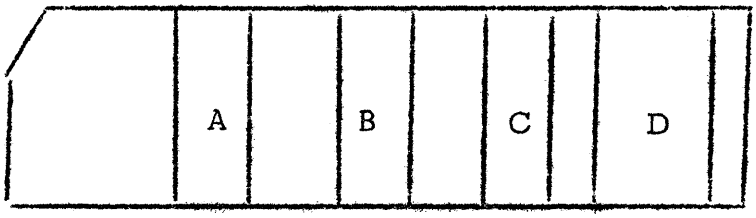
Problem No. 2

Calculate and punch D where D = A + B - C
A is a 5 column field
B is a 5 " "
C is a 6 " "

Input Card



Output Card



Problem No. 3

Calculate and punch E where, E = A + B + C - D

Input:	Column 4-8	A	xx.xxx
	9-12	B	x.xxx
	13-14	C	x.x
	15-17	D	.xxx
Output:	Column 4-8	A	xx.xxx
	9-12	B	x.xxx
	13-14	C	x.x
	15-17	D	.xxx
	21-24	E	xx.xx

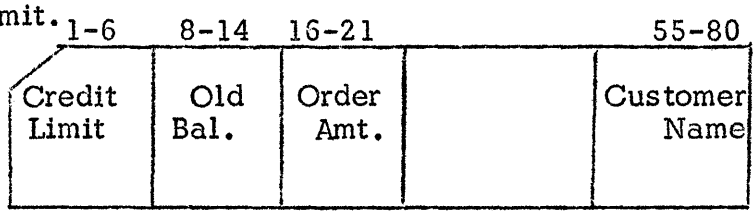
Note: The program must be written so that E is rounded off to two decimal places.

Input cards....5
Output cards...5

Problem No. 4

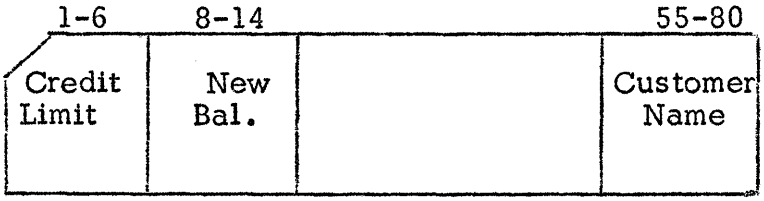
A program is to be written that will up-date a customer's balance, and compare it to his credit limit.

Input Card



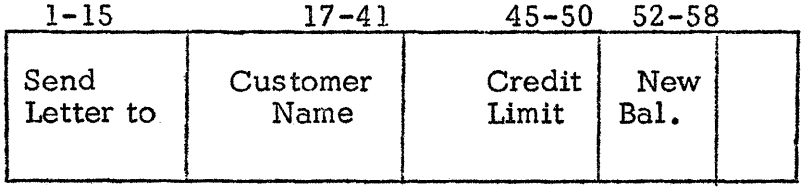
If the sum of the old balance card and the order amount are equal to or less than the credit limit, an up-dated card is to be punched as follows:

Output Card



If the sum of the old balance and the order amount exceed the credit limit, a printed output for the credit manager follows. He, in turn, will send a letter to the customer.

Printed Output



Input cards.....10
Output cards.....10
Printed output.....2 lines