File Number 1401-37
Form F20-0234-2

IBM Systems Reference Library

Auto-Test for IBM 1401: Specifications and Operating Procedures

Program Number 1401-AT-072

This publication contains the specifications and operating procedures for the IBM 1401 Auto-Test program.

Included are sections describing the machine requirements for using the Auto-Test program, Auto-Test considerations, Auto-Test specifications, Auto-Test features, and program operating instructions.

Minor Revision (November 1963)

This edition, F20-0234-2, is a minor revision of and obsoletes F20-0234-1.

The information from Technical Newsletter N24-0120 is included in this edition. In addition:

1. The format is changed to conform to that of the System Reference Library.
2. A paragraph is added after the paragraph titled General Description on page 32.
3. A paragraph is added to the note at the bottom of page 35.
4. The note at the end of page 49 is changed.

TABLE OF CONTENTS

INTRODUCTION

Auto-Test is a testing program for the IBM 1401 Data Processing System. It effectively uses the power of the 1401 system to aid in the testing of Autocoder, SPS, and FARGO programs. Auto-Test provides the ability to stack programs and produce documentation to evaluate the tested programs. Coupled with the ability to generate operating instructions and automatically print core storage and tapes, the Auto-Test program provides the following features which may be selected by the user:

1. Tape File Generator

2. RAMAC File Generator

3. Automatic Patching — The ability to automatically correct the object program without reassembling or manually calculating patching addresses.

4. RAMAC File Trace — Data which is read or written in the disk file during the execution of the object program will be traced and printed.

5. Snapshot core prints may be obtained during the execution of the object program.

6. An 80-80 listing, identified by program, may be obtained for all punched output.

These features of the Auto-Test program may be used in any desired combination as tools for obtaining efficient and successful testing of 1401 programs. The program can handle only program input from cards and cannot be run on the 1410.

MACHINE REQUIREMENTS

The Auto-Test program can test programs for IBM 1401 systems if the test machine has at least:

1. 4,000 positions of core storage.
2. One IBM 729 II, 729 IV, 729 V or 7330 Magnetic Tape Unit. Two tape units are required if RAMAC File Trace and/or Snapshot features are utilized.
3. IBM 1403 Printer, model 2.
4. IBM 1402 Card Read Punch.
5. Advanced Programming feature.
6. High-Low-Equal Compare feature.
7. Sense Switches.

AUTO-TEST CONSIDERATIONS

A minimum amount of core storage is required to use the Auto-Test program, and linkages need not be assembled in the object program to be tested. One control card is required for each card or RAMAC program to be tested. Two control cards are required for each object program which uses tape for input/output or uses the optional RAMAC and/or Snapshot Print features provided in the Auto-Test program.

Core Storage Considerations

Core storage within the 1401 may be thought of as being divided into two sections, one used by the object program under test, and one available to Auto-Test for the necessary linkages, patch area and trace routines. The area of core storage that is available to Auto-Test for these three routines must be one consecutive block of core. This single area will be referred to as the Auto-Test storage area.

1. Auto-Test requires a minimum of 55 positions of core storage. To this must be added 12 positions for each last-card test in the object program. However, the first two last-card tests are included within the basic 55 storage positions. For example, a program with three last-card tests would need 55 positions plus 12 for the third last-card test or a total of 67 positions of core storage.

2. If the programmer should decide to use the RAMAC File Trace feature of Auto-Test, 122 positions of core storage would have to be added to the Auto-Test storage area. See "RAMAC File Trace," page 36, for detailed information.

3. The Snapshot Print, when used, requires additional core storage in the Auto-Test storage area. This snapshot, which stores a picture of core on the tape and prints it out at the end of the test, requires 278 positions of core storage in addition to the basic 55 positions required by the monitor.

4. The Patch feature of Auto-Test is a very simple way of making corrections to an object program. However, the amount of core storage available for automatic patching is limited by the amount of core storage that remains in the Auto-Test storage area after the linkages and selected routines have been loaded.

5. Normally, the Auto-Test storage area is located in the high positions of core storage. The core storage remaining from the highest core storage position used by the object program to the upper limit of core storage on the test machine will automatically become the Auto-Test storage area. For example, if the highest core position used by the object program on the test machine is 2350, the core storage positions between 2351 and 3999, 7999, 11999, or 15999 will be used as the Auto-Test storage area. The programmer may elect to relocate the Auto-Test storage area. If the area is relocated, the area limits of core storage available must be specified. Assume that 100 positions of core storage are available between 2501 and 2600. The low (2501)

and the high (2600) positions must be specified. The core storage positions between these limits will automatically become the Auto-Test storage area.

Note: The card read area and print area must not be specified as part of a relocated Auto-Test storage area.

Tape Drive Considerations

The Auto-Test program is contained on a systems tape. It always occupies one tape drive, and that drive must be specified as drive 6. Any instruction which refers to tape drive number 6 in the object program will be automatically changed to a NOP instruction.

Miscellaneous Considerations

1. Programs that utilize the Punch-Feed-Read feature of the 1401 cannot be tested with Auto-Test.

2. Programs that contain a word separator character (0-5-8) as a constant cannot be tested. These characters will cause a machine process error during the loading phase of Auto-Test.

3. Programs that contain a last-card test (B xxx A) as part of an execute routine (within the EX routine) cannot be tested unless they have been properly modified. (See "Testing Hints," page 48, paragraph 3.)

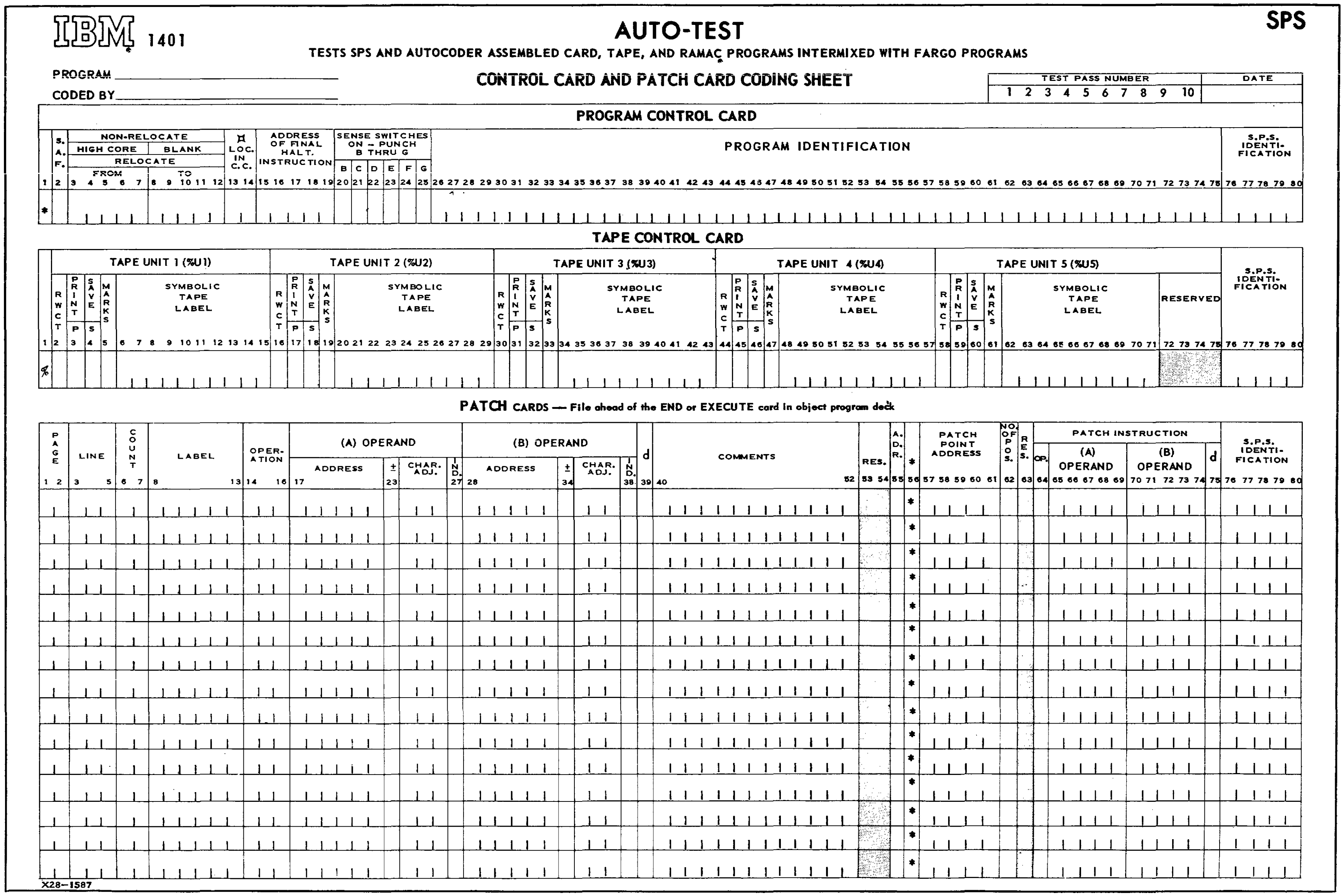
4. An SPS <u>condensed</u> object program cannot be tested.

5. Those programs that do not have the minimum amount of core storage available as specified above cannot be tested. Note: The read area and print area <u>must not</u> be specified as part of a relocated Auto-Test storage area.

6. When testing <u>patched</u> FARGO programs, it must be realized that Auto-Test automatically inserts a 29-position program linkage into core storage positions 333-361. Therefore, the FARGO-generated program being tested by Auto-Test begins loading into storage position 362, instead of the normal location of 333. In order for patched FARGO programs which have been tested with Auto-Test to be run independent of Auto-Test, it will be necessary to insert a constant control card, punched C29000 in columns 1-6, in front of all other constant control cards.

Auto-Test Forms

The forms in Figures 1, 2 and 3 are shown at this point to familiarize the reader with them. They will be discussed in subsequent sections.



IBM 1401

AUTO-TEST

SPS

TESTS SPS AND AUTOCODER ASSEMBLED CARD, TAPE, AND RAMAC PROGRAMS INTERMIXED WITH FARGO PROGRAMS

PROGRAM ______________________

CODED BY ______________________

CONTROL CARD AND PATCH CARD CODING SHEET

TEST PASS NUMBER	DATE
1 2 3 4 5 6 7 8 9 10	

PROGRAM CONTROL CARD

	S.A.F.	NON-RELOCATE HIGH CORE / RELOCATE FROM	NON-RELOCATE BLANK / RELOCATE TO	⌑ LOC. IN C.C.	ADDRESS OF FINAL HALT. INSTRUCTION	SENSE SWITCHES ON – PUNCH B THRU G: B	C	D	E	F	G	PROGRAM IDENTIFICATION	S.P.S. IDENTI-FICATION
1	2	3 4 5 6 7	8 9 10 11 12	13 14	15 16 17 18 19	20	21	22	23	24	25	26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75	76 77 78 79 80
*													

TAPE CONTROL CARD

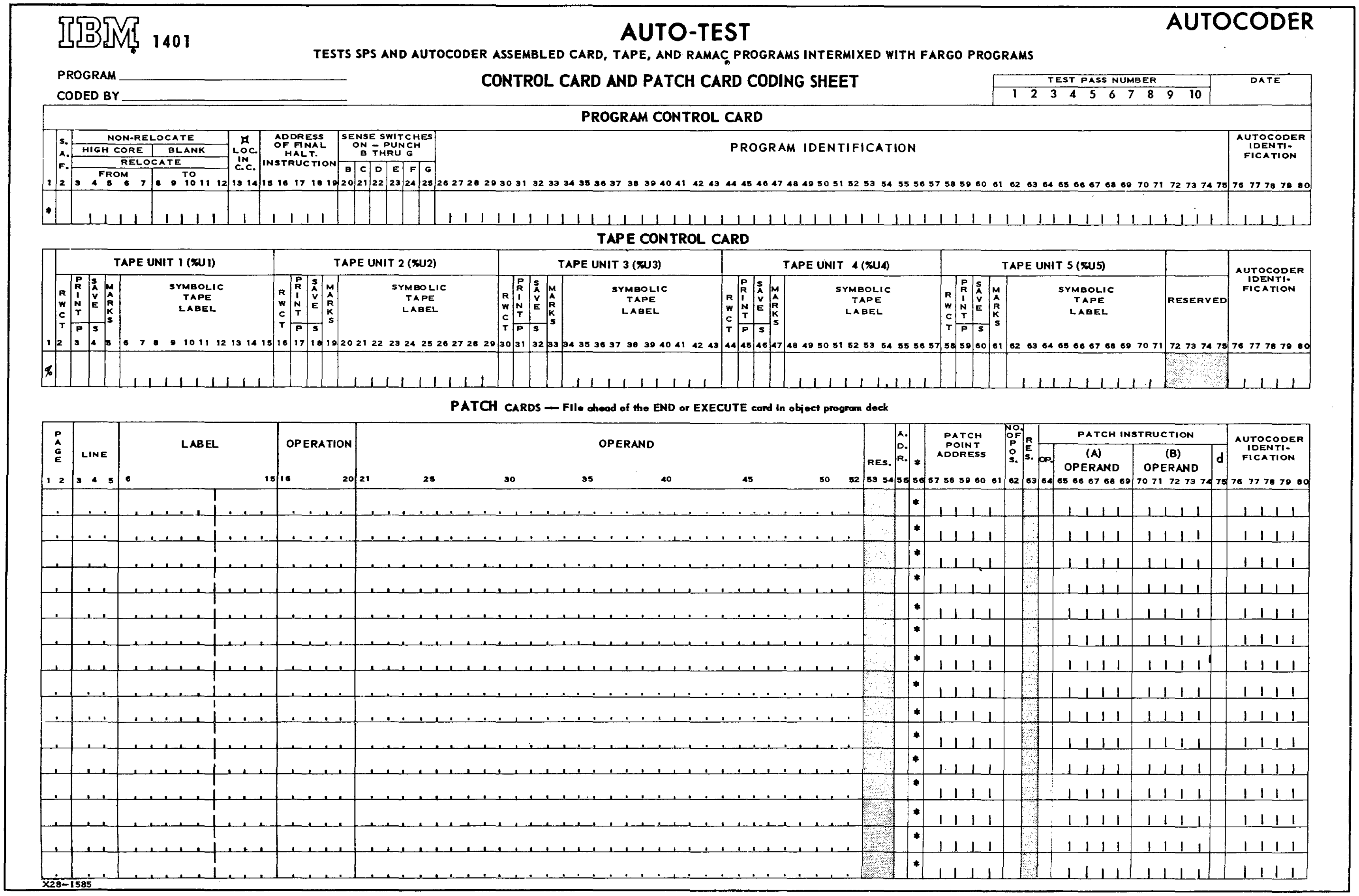
	TAPE UNIT 1 (%U1)					TAPE UNIT 2 (%U2)					TAPE UNIT 3 (%U3)					TAPE UNIT 4 (%U4)					TAPE UNIT 5 (%U5)						S.P.S. IDENTI-FICATION
	RWCT	PRINT P	SAVE S	MARKS	SYMBOLIC TAPE LABEL	RWCT	PRINT P	SAVE S	MARKS	SYMBOLIC TAPE LABEL	RWCT	PRINT P	SAVE S	MARKS	SYMBOLIC TAPE LABEL	RWCT	PRINT P	SAVE S	MARKS	SYMBOLIC TAPE LABEL	RWCT	PRINT P	SAVE S	MARKS	SYMBOLIC TAPE LABEL	RESERVED	
1	2	3	4	5	6 7 8 9 10 11 12 13 14 15	16	17	18	19	20 21 22 23 24 25 26 27 28 29	30	31	32	33	34 35 36 37 38 39 40 41 42 43	44	45	46	47	48 49 50 51 52 53 54 55 56 57	58	59	60	61	62 63 64 65 66 67 68 69 70 71	72 73 74 75	76 77 78 79 80
%																										▨	

PATCH CARDS — File ahead of the END or EXECUTE card in object program deck

PAGE	LINE	COUNT	LABEL	OPER-ATION	(A) OPERAND ADDRESS	±	CHAR. ADJ.	IND.	(B) OPERAND ADDRESS	±	CHAR. ADJ.	IND.	d	COMMENTS	RES.	A.D.R.	*	PATCH POINT ADDRESS	NO. OF POS.	RES.	OP.	PATCH INSTRUCTION (A) OPERAND	PATCH INSTRUCTION (B) OPERAND	d	S.P.S. IDENTI-FICATION
1 2	3 5	6 7	8 13	14 16	17	23		27	28	34		38	39	40 52	53 54	55	56	57 58 59 60 61	62	63	64	65 66 67 68 69	70 71 72 73 74	75	76 77 78 79 80
																	*								
																	*								
																	*								
																	*								
																	*								
																	*								
																	*								
																	*								
																	*								
																	*								
																	*								
																	*								
																	*								
																	*								
																	*								

X28-1587

Figure 1. SPS Control Card and Patch Card Coding Sheet



IBM 1401

AUTO-TEST

AUTOCODER

TESTS SPS AND AUTOCODER ASSEMBLED CARD, TAPE, AND RAMAC® PROGRAMS INTERMIXED WITH FARGO PROGRAMS

PROGRAM ______________

CODED BY ______________

CONTROL CARD AND PATCH CARD CODING SHEET

TEST PASS NUMBER	DATE
1 2 3 4 5 6 7 8 9 10	

PROGRAM CONTROL CARD

	S.A.F.	NON-RELOCATE HIGH CORE / RELOCATE FROM	NON-RELOCATE BLANK / RELOCATE TO	¤ LOC. IN C.C.	ADDRESS OF FINAL HALT. INSTRUCTION	SENSE SWITCHES ON – PUNCH B THRU G: B	C	D	E	F	G	PROGRAM IDENTIFICATION	AUTOCODER IDENTI-FICATION
1	2	3 4 5 6 7	8 9 10 11 12	13 14	15 16 17 18 19	20	21	22	23	24	25	26–75	76 77 78 79 80
*													

TAPE CONTROL CARD

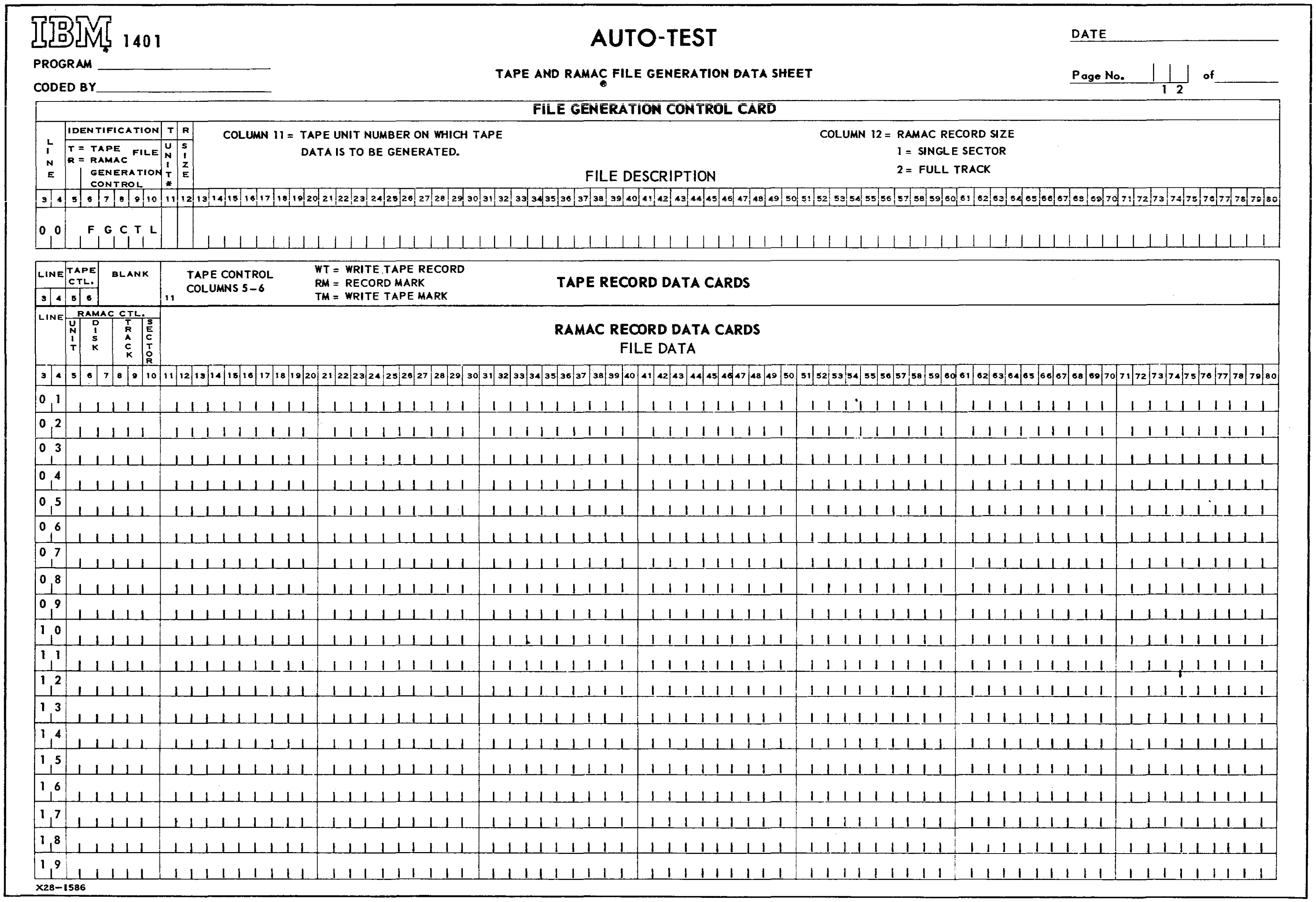
	TAPE UNIT 1 (%U1)				TAPE UNIT 2 (%U2)				TAPE UNIT 3 (%U3)				TAPE UNIT 4 (%U4)				TAPE UNIT 5 (%U5)					
	RWCT	PRINT P	SAVE S	MARKS	SYMBOLIC TAPE LABEL	RWCT	PRINT P	SAVE S	MARKS	SYMBOLIC TAPE LABEL	RWCT	PRINT P	SAVE S	MARKS	SYMBOLIC TAPE LABEL	RWCT	PRINT P	SAVE S	MARKS	SYMBOLIC TAPE LABEL	RESERVED	AUTOCODER IDENTI-FICATION
1	2	3	4	5	6–15	16	17	18	19	20–29	30	31	32	33	34–43	44	45	46	47	48–57	58	59	60	61	62–71	72 73 74 75	76 77 78 79 80
%																						

PATCH CARDS — File ahead of the END or EXECUTE card in object program deck

PAGE	LINE	LABEL	OPERATION	OPERAND	RES.	A.D.R.	*	PATCH POINT ADDRESS	NO. OF OPS.	RES.	OP.	PATCH INSTRUCTION (A) OPERAND	(B) OPERAND	d	AUTOCODER IDENTI-FICATION
1 2	3 4 5	6 15	16 20	21 25 30 35 40 45 50 52	53 54	55	56	57 58 59 60 61	62	63	64	65 66 67 68 69	70 71 72 73 74	75	76 77 78 79 80
							*								
							*								
							*								
							*								
							*								
							*								
							*								
							*								
							*								
							*								
							*								
							*								
							*								
							*								
							*								
							*								

X28–1585

Figure 2. Autocoder Control Card and Patch Card Coding Sheet



IBM 1401

AUTO-TEST

TAPE AND RAMAC® FILE GENERATION DATA SHEET

PROGRAM ____________

CODED BY ____________

DATE ____________

Page No. 1 2 of ______

FILE GENERATION CONTROL CARD

LINE	IDENTIFICATION (T = TAPE, R = RAMAC; FILE GENERATION CONTROL)	T UNIT #	R SIZE	FILE DESCRIPTION
3 4	5 6 7 8 9 10	11	12	13 – 80
0 0	F G C T L			

COLUMN 11 = TAPE UNIT NUMBER ON WHICH TAPE DATA IS TO BE GENERATED.

COLUMN 12 = RAMAC RECORD SIZE
1 = SINGLE SECTOR
2 = FULL TRACK

TAPE RECORD DATA CARDS

LINE	TAPE CTL.	BLANK	
3 4	5 6		11

TAPE CONTROL COLUMNS 5–6
WT = WRITE TAPE RECORD
RM = RECORD MARK
TM = WRITE TAPE MARK

RAMAC RECORD DATA CARDS

FILE DATA

LINE	RAMAC CTL. UNIT	DISK	TRACK	SECTOR	FILE DATA							
3 4	5	6 7	8 9	10	11–20	21–30	31–40	41–50	51–60	61–70	71–80
0 1											
0 2											
0 3											
0 4											
0 5											
0 6											
0 7											
0 8											
0 9											
1 0											
1 1											
1 2											
1 3											
1 4											
1 5											
1 6											
1 7											
1 8											
1 9											

X28-1586

Figure 3. Tape and RAMAC File Generation Data Sheet

AUTO-TEST SPECIFICATIONS

Setup Instructions

Auto-Test will accept (1) SPS-assembled programs in uncondensed form, (2) Autocoder-assembled programs, and (3) FARGO programs and control cards. These programs may be stacked in any sequence for testing.

1. The first thing a programmer must do is to decide whether the object program can be tested (see "Auto-Test Considerations," page 7). Secondly, if it can be tested, the programmer should decide whether the Auto-Test storage area needs to be relocated.

2. The first card of each program must be an Auto-Test program control card. This card may be followed by the optional instruction card and the console status card.

3. If the program to be tested utilizes tape, or is a RAMAC program which will use the RAMAC File Trace feature, or is a program which will use the Snapshot feature, a tape control card must be included.

4. Instructions for punching Auto-Test control cards may be found under "Control Card, Formats," page 13. The control card coding sheet is an aid in filling out these cards.

5. If it is desired to create tapes for input to a program, or to create a RAMAC file, the Tape and RAMAC File Generation Data Sheet should be used. Instructions for filling out these sheets may be found under "Tape File Generation," page 18, and "RAMAC File Generation," page 20.

6. The last input card of each program to be tested must contain a 12-8-4 (⌑ lozenge) punch. This punch may be placed in any column desired. It is used as a substitution for the last-card test in the object program. If the program does not use cards as input, and therefore would not contain a last-card test instruction, this Auto-Test requirement may be ignored.

7. Each program must contain the normal clear storage, bootstrap and END cards. The Autocoder assembly creates two cards after each execute (EX) instruction. These two cards must not be removed from the object program.

8. A last-card test must be given prior to a Start Read Feed or Start Punch Feed command if the program under test is utilizing this feature. (This last-card test may be patched by using the automatic patching feature of the Auto-Test program.)

9. If the programmer decides to relocate the Auto-Test storage area he must not include the card read area or the print area within this Auto-Test storage area.

Exhibits showing the sequence of control cards, file generation control cards, object programs, input cards, etc., are contained in the appendix.

Control Card Formats

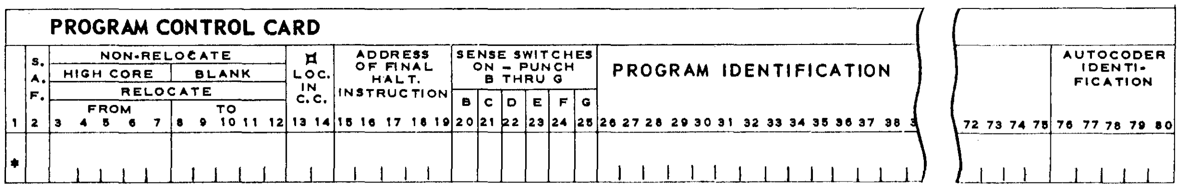
PROGRAM CONTROL CARD

Each program to be tested must have a program control card (Figure 4).

Card Column	Description
1	11-8-4 (* asterisk)
2	S — if the program is assembled in SPS single instruction per card A — if the program is assembled in Autocoder F — if this is a FARGO program
3-7	Punch the address of either (a) the highest position of core storage that the object program uses, if the Auto-Test storage area is not to be relocated, or (b) the low limit or beginning point in core for the Auto-Test storage area if this area is to be relocated.
8-12	Either (a) leave blank if the Auto-Test storage area is not to be relocated, or (b) punch the address of the high limit or ending point in core for the Auto-Test storage area if this area is to be relocated.
13-14	Punch the number of the card column in which the lozenge appears in the last data card. If the program under test does not read cards as input, it will not contain a last-card test. Therefore, these two columns may be left blank.
15-19	Punch the core storage address of the final end-of-job halt in the object program. (Note: This is the address of the OP code of the halt instruction.) If these card columns are blank, or if the final halt is a one-position instruction, linkage to Auto-Test will have to be effected through the console at the Auto-Test restart point.
20-25	Punch the B through G alphabetic punches in the proper columns to indicate the sense switches that must be on during the test. Leave blank for off.
26-75	Punch the programmer's identification for the program. Example: Payroll Register — The Smith Company.
76-80	The five-character identification as used in the source program.

Note: The addresses requested in this card may be specified by using a four-position address, left-justified in the field, or a five-position address. Three-position addresses must not be used. (Example: 3529 or 03529 may be used; E29 must not be used.)

When specifying FARGO programs on the program control card, it is only necessary to punch an 11-8-4 (*) in column 1, an F in column 2, and the program identification in columns 26-80. The FARGO data cards must be followed by two special cards, the first with a ¤ (12-8-4) in column 1 and the second with B333 in columns 1-4.



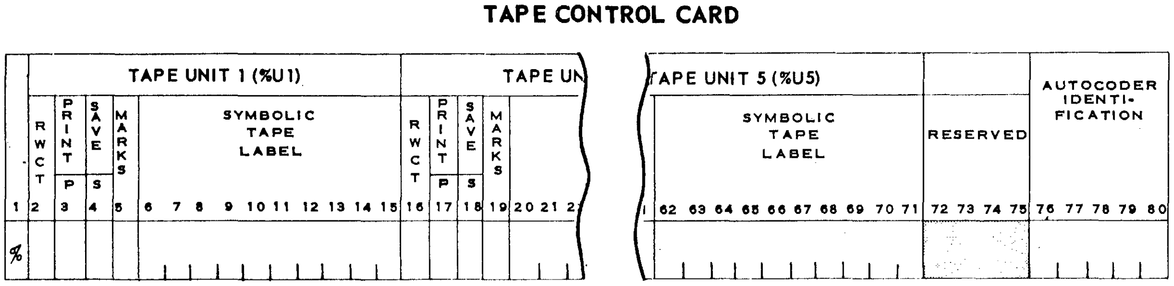
PROGRAM CONTROL CARD

	S. A. F.	NON-RELOCATE HIGH CORE	NON-RELOCATE BLANK	¤ LOC. IN C.C.	ADDRESS OF FINAL HALT. INSTRUCTION	SENSE SWITCHES ON - PUNCH B THRU G						PROGRAM IDENTIFICATION		AUTOCODER IDENTI-FICATION
		RELOCATE FROM	RELOCATE TO			B	C	D	E	F	G			
1	2	3 4 5 6 7	8 9 10 11 12	13 14	15 16 17 18 19	20	21	22	23	24	25	26 27 28 29 30 31 32 33 34 35 36 37 38	72 73 74 75	76 77 78 79 80
*														

Figure 4. Auto-Test Program Control Card Layout

TAPE CONTROL CARD

One tape control card (Figure 5) must be punched for each object program that utilizes tape for either input or output, or for a program that uses the Snapshot or RAMAC File Trace features of Auto-Test. File this single card as illustrated in the appendix. This card is divided into five sections of 14 columns each, one section for each possible tape drive on line. (This excludes drive 6, which is always the Auto-Test systems tape drive.) The fields in this card are punched according to the corresponding tape drives used in the object program. For example, if tape drives 1 and 5 are used in the object program, those sections labeled Tape Unit 1 (% U1) and Tape Unit 5 (% U5) must be punched.



TAPE CONTROL CARD

	TAPE UNIT 1 (%U1)					TAPE U					TAPE UNIT 5 (%U5)		
	RWCT	PRINT P	SAVE S	MARKS	SYMBOLIC TAPE LABEL	RWCT	PRINT P	SAVE S	MARKS		SYMBOLIC TAPE LABEL	RESERVED	AUTOCODER IDENTI-FICATION
1	2	3	4	5	6 7 8 9 10 11 12 13 14 15	16	17	18	19	20 21 2	62 63 64 65 66 67 68 69 70 71	72 73 74 75	76 77 78 79 80
%													

Figure 5. Auto-Test Tape Control Card Layout

Card Column	Description
1	0-8-4 (% percent)
2	R — if this tape will be read as input (e.g., Precreated Master File) W — if this tape will be written as output C — if the Auto-Test program will create this tape or any portion thereof using the Tape File Generator feature T — if this drive has been specified for the RAMAC File Trace and/or Snapshot Print feature Blank — if the drive is not being used for this test.
3	P — to print out this tape at the end of the test Blank — if this tape is not to be printed
4	S — to save this tape after the program test Blank — do not save this tape
5	1 through 9 — to indicate the number of tape marks that will be found on this tape if it has been properly created. Note: Each tape should always end with a tape mark.
6-15	Symbolic identification for this tape. Example: PAYMASTER.
16-71	These columns are a duplication of columns 2-15 above for each tape unit that is used by the object program (maximum of five tape units)
72-75	Blank
76-80	The five-character identification as used in the source program

Optional Cards

INSTRUCTION CARDS

This feature allows the programmer to insert special instructions, for example, MOUNT THE DETAIL TAPE ON THE 1011 PAPER TAPE READER. The presence of these cards will cause the machine to halt so that the operator may perform the necessary instructions. Any number of instruction (I) cards may be used for a given program test. The special instructions will be printed at the same time as the machine-generated instructions. (See "Operator Instruction Generation," page 17.)

Card Column	Description
1	"I" (the letter I)
2-80	Instructions that the programmer wishes to be printed

CONSOLE STATUS CARD

This optional unpunched card may be placed in sequence immediately after the program control card. If used, it is selected to the 8/2 stacker for ready availability. This card is used to record the status of the console in case of a program process error; the back of this card space is used to list any halts that might occur during the testing of a program and the action to be taken at the time. This card must not contain any punches. (See appendix for example of the console status card.)

Automatic Modification to the Object Program Being Tested

There is no need to make changes of any type to an object program, or to assemble it in any special way, to be able to test with Auto-Test except when using the optional RAMAC File Trace feature.

While Auto-Test loads the object program into core, it makes certain changes automatically to the program. These changes will appear on the core storage printout. These modifications are:

1. All tape unload commands are changed to rewinds. Tape units are unloaded only under Auto-Test control when necessary.

2. Every reference to tape 6, in the object program under test, will be NOP'd. Tape 6 is the Auto-Test systems tape.

3. Each last-card test (B xxx A) command is changed to an unconditional branch to the Auto-Test storage area where the command is re-created as an eight-position test for the 12-8-4 (lozenge) punch in the last data card. This area is located just beyond the highest position core specified in columns 3-7 of the program control card.

4. The SPS and Autocoder rules for using an EX command specify that the last instruction of this execute structure must be a read card and branch back to loading. These commands are modified to create an unconditional branch to the Auto-Test load monitor so that the Auto-Test program may continue loading the object program. Note: The command format must be a Read and Branch, not a Read command followed by a Branch command (1056, not 1B056).

5. If the final halt in the object program is a four- or seven-position instruction, it will be changed to a branch to the Auto-Test restart monitor so that no halt will occur, and testing will continue without manual intervention into the storage print phase. If it is a one-position final halt, no change can be made and the program will halt. This will necessitate a manual console restart to return to the Auto-Test program.

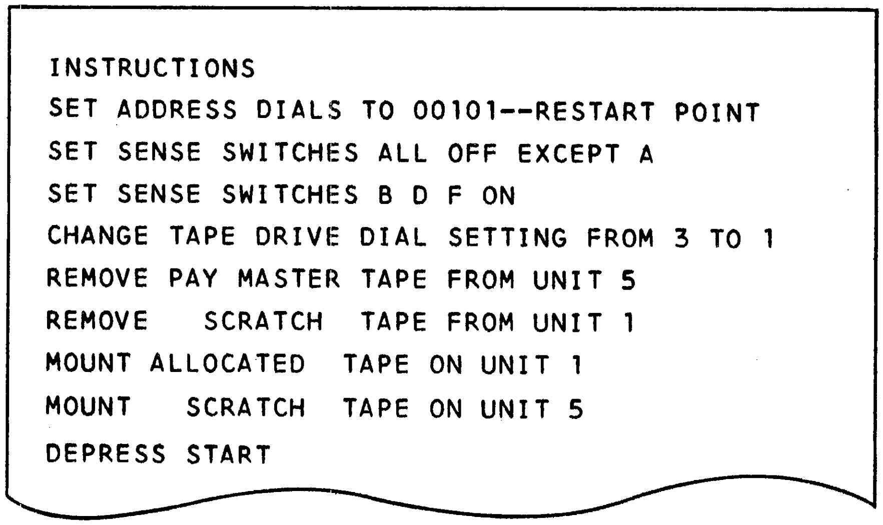
AUTO-TEST FEATURES

Auto-Test will automatically generate complete operating instructions and core storage prints for every object program. In addition to these automatic functions, the user may specify any combination of the optional features provided by Auto-Test to obtain an efficient and thoroughly documented test of all object programs.

Operator Instruction Generation

The Auto-Test program will automatically generate any necessary operator instructions to properly test a card, tape or RAMAC program. These instructions are generated by a comparison of the status of the machine during the previous test with the requirements of the program about to be tested. The instructions will be printed on the printer.

Examples of typical generated instructions are shown in Figure 6.



```
INSTRUCTIONS
SET ADDRESS DIALS TO 00101--RESTART POINT
SET SENSE SWITCHES ALL OFF EXCEPT A
SET SENSE SWITCHES B D F ON
CHANGE TAPE DRIVE DIAL SETTING FROM 3 TO 1
REMOVE PAY MASTER TAPE FROM UNIT 5
REMOVE    SCRATCH  TAPE FROM UNIT 1
MOUNT ALLOCATED  TAPE ON UNIT 1
MOUNT    SCRATCH  TAPE ON UNIT 5
DEPRESS START
```

Figure 6. Typical Generated Instructions

Instructions should be executed consecutively beginning with the top instruction. If no operator intervention is necessary between program tests, the following message will print on the 1403:

```
INSTRUCTIONS
NONE
```

The 1401 will not halt between program tests unless some operator intervention is necessary.

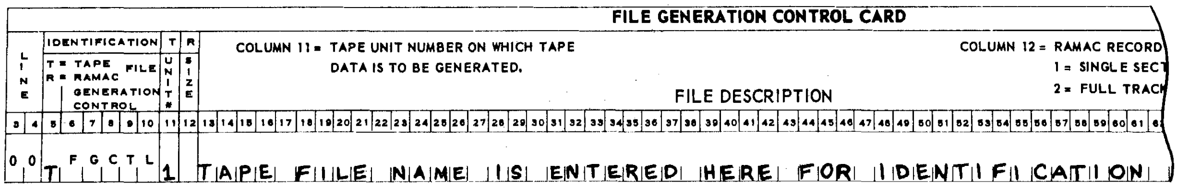
Tape File Generation

The Auto-Test Tape File Generator provides for the creation of tape files prior to the test of each individual program. The tape files can be created with header and trailer labels. The data tape records may be created with or without wordmarks in blocked or unblocked, fixed or variable length format. Maximum record length that can be created is 2,000 for a 4K machine, and 6,000, 10,000, 14,000 for 8, 12, and 16K machines respectively. To generate tape files, two types of cards are necessary, as illustrated in Figure 3.

TAPE FILE GENERATION CONTROL CARD

A tape file generation control card (Figure 7) is required for each file to be created, and has the following format:

Card Column	Description
1-2	Page number
3-4	Line Number
5-10	TFGCTL
11	Tape unit number on which the file is to be created
12	Blank
13-80	Tape File Description


FILE GENERATION CONTROL CARD

LINE | IDENTIFICATION: T = TAPE, R = RAMAC; FILE GENERATION CONTROL | T UNIT # | R SIZE

COLUMN 11 = TAPE UNIT NUMBER ON WHICH TAPE DATA IS TO BE GENERATED.

COLUMN 12 = RAMAC RECORD
1 = SINGLE SECT
2 = FULL TRACK

FILE DESCRIPTION

3	4	5	6	7	8	9	10	11	12	13–62
0	0	T	F	G	C	T	L	1		TAPE FILE NAME IS ENTERED HERE FOR IDENTIFICATION

Figure 7. File Generation Control Card Layout

TAPE RECORD DATA CARD

Card Column	Description
1-2	Page number
3-4	Line number
5-6	WT — Write tape record RM — Record mark

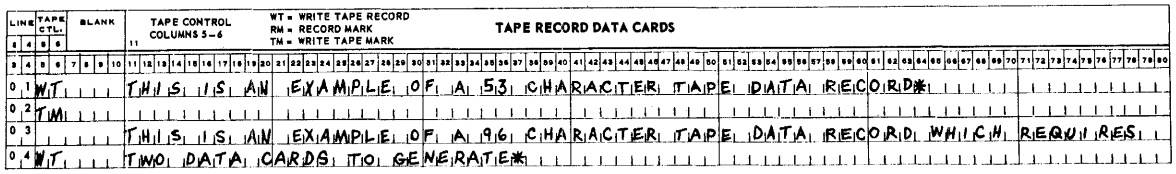
Card Column	Description
5-6 cont.	TM — Write tape mark Blank -- Record length exceeds this card capacity
7-10	Blank
11-80	Data to be created on the tape

This card is used to create the tape file specified by the TFG control card. The data to be created on tape is punched in columns 11-80 in the tape image format. Multiple data cards are used to create larger tape records. Each new record must start in column 11 of a new data card. If it is desired to create a tape with word marks, a word separator character (0-5-8) is punched one column before the character with which it is associated. The end of each record is specified by punching an 11-8-4 (asterisk) one position to the right of the last data character. The use of the asterisk in the control card does not prevent its use within the record as a character or part of the tape record to be created. WT (write tape) must also be punched in columns 5-6 of the last data card for each record. This causes the record to be written on the tape unit specified in the TFG control card.

A tape mark may be generated on the tape by punching TM (tape mark) in columns 5-6 and leaving the rest of the card blank. For example, if a tape mark is desired between the data file and a trailer label, a TM card would be inserted before the first data card of the trailer label.

When a file is completely generated, a tape mark is automatically written on the tape and the tape is rewound.

For an example of tape record data cards, see Figure 8.



LINE	TAPE CTL.	BLANK	TAPE RECORD DATA CARDS
3-4	5-6	7-10	11-80
01	WT		THIS IS AN EXAMPLE OF A 53 CHARACTER TAPE DATA RECORD*
02	TM		
03			THIS IS AN EXAMPLE OF A 96 CHARACTER TAPE DATA RECORD WHICH REQUIRES
04	WT		TWO DATA CARDS TO GENERATE*

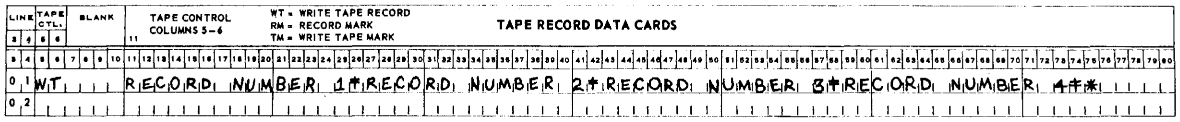
TAPE CONTROL COLUMNS 5-6
WT = WRITE TAPE RECORD
RM = RECORD MARK
TM = WRITE TAPE MARK

Figure 8. Tape Record Data Cards

When generating blocked records which contain record marks (0-8-2) as record separators, the user may elect to begin each individual record in the block on a new data card starting in column 11, rather than punch the complete block continuously. Punching RM (record mark) in columns 5-6 indicates that the record mark which is used as the record separator is the last character contained in this card. All card columns to the right of the last record mark will be ignored and only the record mark and the characters to the left will be used to build the block. The next individual record of the block must begin in column 11 of the next data card. The last card of the block, as in single records, must have the WT in columns 5-6 and an 11-8-4 (asterisk) punch following the last character of the block.

For an example of a blocked record punched continuously in data cards, see Figure 9.

For an example of a blocked record punched in separate data cards by records, see Figure 10.

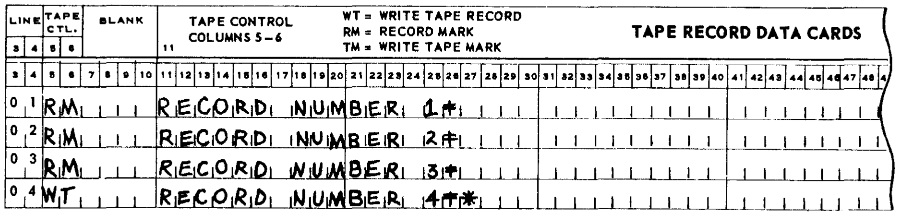


TAPE RECORD DATA CARDS

TAPE CONTROL COLUMNS 5–6: WT = WRITE TAPE RECORD, RM = RECORD MARK, TM = WRITE TAPE MARK

LINE (3-4)	TAPE CTL. (5-6)	BLANK (7-10)	Columns 11-80
01	WT		RECORD NUMBER 1ǂRECORD NUMBER 2ǂRECORD NUMBER 3ǂRECORD NUMBER 4ǂ*
02			

Figure 9. Tape Record Data Cards — Blocked Record Continuously Punched



TAPE RECORD DATA CARDS

TAPE CONTROL COLUMNS 5–6: WT = WRITE TAPE RECORD, RM = RECORD MARK, TM = WRITE TAPE MARK

LINE (3-4)	TAPE CTL. (5-6)	BLANK (7-10)	Columns 11-
01	RM		RECORD NUMBER 1ǂ
02	RM		RECORD NUMBER 2ǂ
03	RM		RECORD NUMBER 3ǂ
04	WT		RECORD NUMBER 4ǂ*

Figure 10. Tape Record Data Cards — Blocked Record Separately Punched

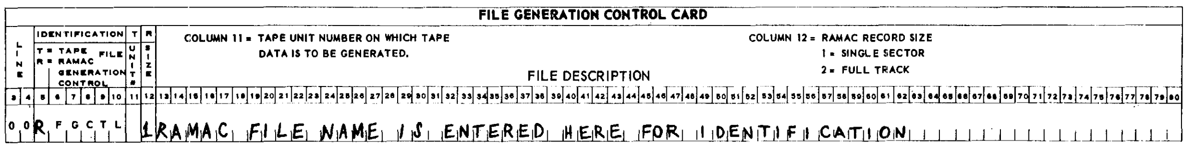
RAMAC File Generation

The Auto-Test RAMAC File Generator provides for the creation of the disk file records prior to the test of each individual program. The file records are written in the MOVE mode only. The record size may be single sector and/or full track. If a tape drive number is specified for the RAMAC Trace feature on the tape control card, the file records will also be written on tape, as they are created on the file, and printed at the end of the program test. For a detailed explanation of this feature, see "RAMAC File Trace," page 36.

RAMAC FILE GENERATION CONTROL CARD

A RAMAC file generation control card (Figure 11) is required for either record size to be created. If only single-sector or full-track records are to be created, only one control card is needed; if single-sector and full-track records are to be created, two control cards are needed. Any time a new record size is desired, an RFG control card specifying the new size must precede the data cards which will create the records.

Card Column	Description
1-2	Page number
3-4	Line number
5-10	RFGCTL
11	Blank
12	1 — for single-sector records 2 — for full-track records
13-80	File description



FILE GENERATION CONTROL CARD

LINE	IDENTIFICATION T = TAPE R = RAMAC FILE GENERATION CONTROL	T UNIT #	R SIZE	COLUMN 11 = TAPE UNIT NUMBER ON WHICH TAPE DATA IS TO BE GENERATED. FILE DESCRIPTION COLUMN 12 = RAMAC RECORD SIZE 1 = SINGLE SECTOR 2 = FULL TRACK
3 4	5 6 7 8 9 10	11	12	13–80
0 0	R F G C T L		1	RAMAC FILE NAME IS ENTERED HERE FOR IDENTIFICATION

Figure 11. RAMAC File Generation Control Card

RAMAC RECORD DATA CARD

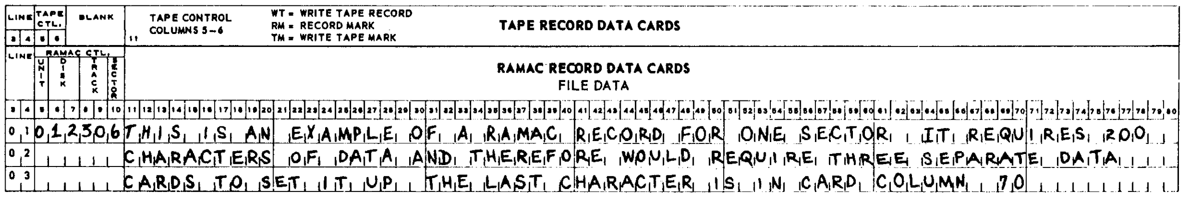
This card is used to create records on the disk file. The data to be created is punched in columns 11-80 in the record image format. Records can be created in the move mode only. Multiple data cards are required for the generation of either single sector or full tracks. If a single sector were desired, three data cards would be needed. Card 1 would contain the disk address of the records in columns 5-10 and the first 70 data characters of a sector in columns 11-80. Card 2 would have the next 70 data characters in columns 11-80. Card 3 would have the last 60 data characters in columns 11-70. The creation of a full track would require 15 data cards. Cards 1-14 would contain data in columns 11-80. Card 15 would have the last 20 data characters in 11-30. Note: Only the first data card of each sector or track contains the disk address.

Card Column	Description
1-2	Page number
3-4	Line number
5	Unit number (0)
6-7	Disk number
8-9	Track number

Card Column	Description
10	Sector number
11-80	Data to be created on the file

For an example of RAMAC record data cards, see Figure 12.

The generation of a series of single-sector records may be followed by the generation of a series of full-track records by placing a second RFG control card and sets of 15 full-track record data cards behind the last data card for the single-sector data file.



LINE 3 4 | TAPE CTL. 5 6 | BLANK | TAPE CONTROL COLUMNS 5-6 | WT = WRITE TAPE RECORD | RM = RECORD MARK | TM = WRITE TAPE MARK | TAPE RECORD DATA CARDS

RAMAC RECORD DATA CARDS
FILE DATA

LINE (3-4)	RAMAC CTL. UNIT (5)	DISK (6-7)	TRACK (8-9)	SECTOR (10)	FILE DATA (11-80)
01	0	12	30	6	THIS IS AN EXAMPLE OF A RAMAC RECORD FOR ONE SECTOR IT REQUIRES 200
02					CHARACTERS OF DATA AND THEREFORE WOULD REQUIRE THREE SEPARATE DATA
03					CARDS TO SET IT UP THE LAST CHARACTER IS IN CARD COLUMN 70

Figure 12. RAMAC Record Data Cards

Automatic Patching

The Patch feature of Auto-Test facilitates changes and corrections to assembled object programs, making it possible to eliminate program reassemblies. The Patch feature provides for deleting any object instruction, adding instructions, replacing instructions and adding multiple-instruction subroutines to be branched to from several points within the object program. Only one patch card need be coded to specify any instruction or constant. It is not necessary for the programmer to compute three-position alphameric addresses to locate the patch in core, or to provide the necessary patch linkage between the object program and the patch area. The Patch feature computes all necessary linkage and inserts patches in the Auto-Test area of core storage. (See "Core Storage Considerations," page 7.)

Auto-Test control card and patch card coding sheets provide space for coding up to 15 patch cards per sheet.

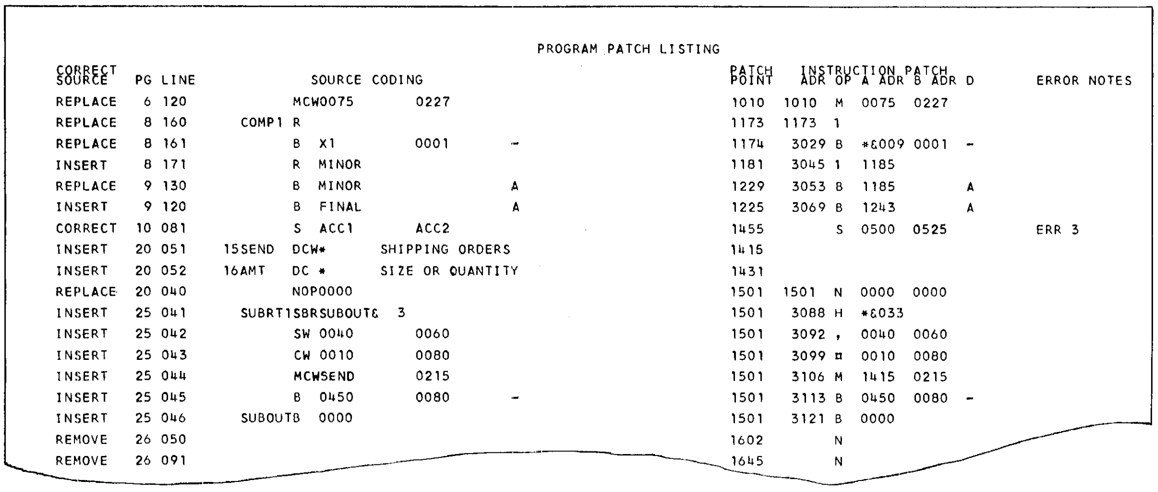
These code sheets are available in the following two forms:

SPS sheets (see Figure 1, page 9) are designed so that the source SPS coding can be entered in the first 52 columns, and the "actual" patch coding can be entered in the remaining columns.

Autocoder sheets (see Figure 2, page 10) are designed so that the source Autocoder coding can be entered in the first 52 columns, and the "actual" patch coding can be entered in the remaining columns.

Patch cards are inserted in the object program as a group, just ahead of the END card, or, if patches are being made to an execute routine, just ahead of the Execute card. The processing of patch cards takes place during the loading of the object program under Auto-Test control. The patches are written on tape 6 (Auto-Test systems tape) and then loaded back to core storage along with the object program for execution.

Each time the patch cards are processed, a program patch listing (Figure 13) is printed, which provides full documentation of all changes being made to an object program. The left-hand column of this listing gives an indication of how the "source" program should be corrected.



```
                                                             PROGRAM PATCH LISTING
CORRECT                                                                           PATCH    INSTRUCTION PATCH
SOURCE   PG LINE          SOURCE CODING                                           POINT     ADR OP A ADR B ADR D        ERROR NOTES
REPLACE   6 120          MCW0075      0227                                        1010  1010 M  0075  0227
REPLACE   8 160     COMP1 R                                                       1173  1173 1
REPLACE   8 161          B  X1        0001     -                                  1174   3029 B *&009 0001  -
INSERT    8 171          R  MINOR                                                 1181   3045 1 1185
REPLACE   9 130          B  MINOR              A                                  1229   3053 B 1185        A
INSERT    9 120          B  FINAL              A                                  1225   3069 B 1243        A
CORRECT  10 081          S  ACC1      ACC2                                        1455        S 0500  0525              ERR 3
INSERT   20 051    15SEND DCW*     SHIPPING ORDERS                                1415
INSERT   20 052    16AMT  DC *     SIZE OR QUANTITY                               1431
REPLACE  20 040          NOP0000                                                  1501  1501 N  0000  0000
INSERT   25 041     SUBRT1SBRSUBOUT&  3                                           1501   3088 H *&033
INSERT   25 042          SW 0040      0060                                        1501   3092 , 0040  0060
INSERT   25 043          CW 0010      0080                                        1501   3099 ¤ 0010  0080
INSERT   25 044          MCWSEND      0215                                        1501   3106 M 1415  0215
INSERT   25 045          B  0450      0080     -                                  1501   3113 B 0450  0080  -
INSERT   25 046     SUBOUTB 0000                                                  1501   3121 B 0000
REMOVE   26 050                                                                   1602        N
REMOVE   26 091                                                                   1645        N
```

Figure 13. Typical Program Patch Listing

Upon successfully completing the testing of any given program, the programmer has the following options:

1. Using the final program patch listing as a guide, to file the patch cards in the source program for final reassembly.

Note: Patch cards must be reproduced, dropping out columns 55-75, prior to insertion in the source program deck which is to be reassembled.

2. To place the program control card in front of the patch cards and allow the Auto-Test program to punch object self-loading patch cards, which, when placed in the object program, provide a complete self-loading program.

Note: Loadable patch cards which affect an execute subroutine must be placed before the execute card. All other loadable patch cards would be placed before the END-START card of the object program.

Format of patch cards:

Card Column	Description
1-52	Source instruction in Autocoder or SPS format
53-54	Reserved (leave blank)

Card Column	Description
55	A — for adding an instruction or constant D — for deleting an instruction R — for replacing an instruction or constant
56	11-8-4 (asterisk) identifies this as a patch card
57-61	Patch point address
62	Number of positions in patch point instruction
63	(leave blank)
64-75	Actual instruction to be added or replaced
76-80	Program identification

Address Types. Five types of addresses are valid in the patch-point address field and the patch instruction A and B operand fields of the patch card:

1. Actual three-position alphameric left-justified.
2. Actual four-position numerical left- or right-justified.
3. Actual five-position numerical.
4. 11-8-4 (asterisk) left-justified.
5. 8-3 (# pound sign) left-justified.

For illustrations of these address types, see patching examples below.

Definitions

1. Patch-point address (columns 57-61 of the patch cards):
   a. When referring to an instruction patch, the patch-point address is the instruction (OP code) address of the instruction in the object program which is to be deleted or replaced, or after which added patch instructions are to be executed.
   b. When referring to a constant patch (DC or DCW), the patch-point address designates the core storage address where the units position or low-order position of the constant will be placed.
2. Number of positions (column 62 of the patch cards):
   a. For an instruction patch, the number of positions is the count of the instruction located at the patch-point address.
   b. For a constant patch, number of positions is left blank.

c. For insertion of a subroutine in the patch area, number of positions may be left blank.

PATCH RULES

1. Do not specify as a patch-point instruction any object program instruction which is being modified in core storage during execution, since this instruction will be relocated to the patch area if a branch-out linkage must be inserted to effect the patch.

2. Do not specify as a patch-point instruction for an ADD type of patch, any unconditional branch, or last-card test, as the patch will never be executed.

3. Patch instructions with indexed operands must utilize three-position actual address (e.g., 5A1).

4. Do not specify patch-point instructions in the middle of a chained instruction series.

5. The patch-point instruction specified for an add patch must be four positions or more in length.

PATCH ERROR NOTES

General Description. During the loading and patching of a program under Auto-Test control, patch cards are checked for the errors noted below. Error messages are printed on the patch listing and the error patch is bypassed.

Error Message	Error Made by Programmer
ERR 1	No patch-point address specified in columns 57-61
ERR 2	No OP code in column 64 of instruction patch card
ERR 3	Object program patch-point instruction is too small to insert branch linkage to the patch area. (See example of how this error is indicated in Figure 13.)
ERR 4	No count in columns 6-7 SPS DC or DCW patch card
ERR 5	Source coding error in Autocoder DC or DCW patch card
ERR 6	Tape unit 6 instruction patched in as a NOP
ERR 7	Core storage limits of patch area exceeded. No patch made.

DELETE PATCH

General Description. A delete patch card may be used to place a NOP (N) with word mark at any patch point specified in core storage. For example, if the programmer desires to delete the instruction located at 2505 (OP code address), the delete card includes source page and line number, patch code D in column 55 and 11-8-4 (asterisk) in column 56, and patch-point address in columns 57-61 (see Figure 14).

DELETE PATCH

PATCH CARDS — File ahead of the END or EXECUTE card in object program deck

PAGE 1-2	LINE 3-5	COUNT 6-7	LABEL 8-13	OPERATION 14-16	(A) OPERAND ADDRESS 17	± 23	CHAR. ADJ.	IND. 27	(B) OPERAND ADDRESS 28	± 34	CHAR. ADJ.	IND. 38	d 39	COMMENTS 40-52	RES. 53-54	A.D.R. 55	* 56	PATCH POINT ADDRESS 57-61	NO OF POS. 62	RES. 63	PATCH INSTRUCTION OP. 64	(A) OPERAND 65-69	(B) OPERAND 70-74	d 75	S.P.S. IDENTIFICATION 76-80
08	130															D	*	2505							

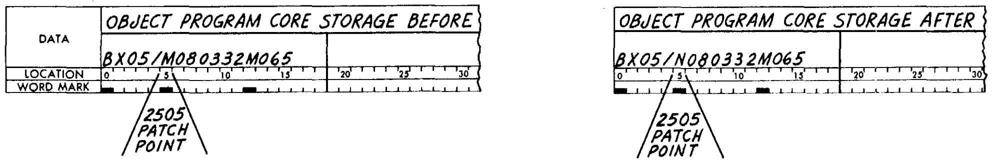


Figure 14. Delete Patch Card

ADD PATCH

General Description. An add patch card may be used to insert an additional instruction (or instructions) in an object program at the patch point specified so that it may be executed as part of the assembled program. Auto-Test sets up the necessary branch-out and branch-back linkages and inserts the additional instruction(s) in the patch area of core storage. The patch-point instruction specified for an add patch must be four positions or more in length.

For example, if the programmer desires to insert additional instructions which are to be executed after the instruction located at the patch-point specified in the object program:

1. The additional instructions are coded in the source language of the program (SPS or Autocoder).

2. The patch codes A in column 55 and 11-8-4 (asterisk) in column 56 are specified.

3. The patch point is specified as the instruction (OP code) address of the instruction after which the additional instruction(s) are to be executed. (For multiple add patches, the same patch-point address for each, in exactly the same columns of the card, is specified.)

4. In column 62, the number of positions in the patch-point instruction is specified.

5. The actual machine-language OP code of the instruction to be added is written in column 64.

6. The A operand in columns 65-69, the B operand in 70-74, and the d modifier in column 75 are specified.

The Auto-Test system:

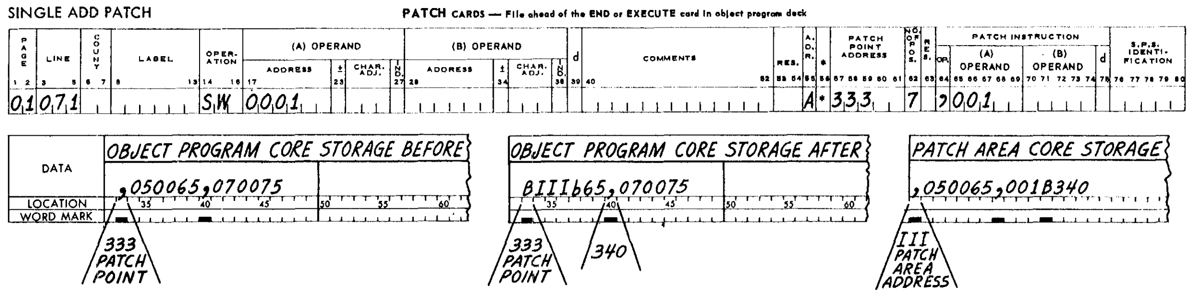
1. Loads the patch-point instruction from its location in the object program to a location in the patch area.

2. Loads an unconditional branch instruction over the patch-point instruction to effect linkage to the patch area.

3. Loads the additional instruction(s) to the patch area following the relocated patch-point instruction. The five-position patch-point address in columns 57-61 in each patch card is compared with the patch-point address of the preceding patch card. If the compare is equal, the patch instructions are located sequentially in the patch area of core storage.

4. Loads an unconditional branch instruction to the patch area following the added instruction(s), to effect linkage back to the next sequential instruction after the patch-point instruction in the object program core storage area.

See Figure 15 for an example of a "single add" type of patch coded on an SPS patch code sheet.



SINGLE ADD PATCH

PATCH CARDS — File ahead of the END or EXECUTE card in object program deck

PAGE 1 2	LINE 3 5	COUNT 6 7	LABEL 8 13	OPERATION 14 16	(A) OPERAND ADDRESS 17	± 23	CHAR. ADJ.	IND. 27	(B) OPERAND ADDRESS 28	± 34	CHAR. ADJ.	IND. 38	d 39	COMMENTS 40 52	RES. 53 54	A.D.R. 55	* 56	PATCH POINT ADDRESS 57 58 59 60 61	NO. OF POS. 62	RES. 63	OP. 64	PATCH INSTRUCTION (A) OPERAND 65 66 67 68 69	PATCH INSTRUCTION (B) OPERAND 70 71 72 73 74	d 75	S.P.S. IDENTIFICATION 76 77 78 79 80
01	071			SW	0001											A	*	333	7		,	001			

Figure 15. Single Add Type of Patch

The programmer wishes to add an instruction to set a word mark in storage position 001. This instruction is to follow the seven-position instruction located at core storage position 00333.

To accomplish this the programmer codes the patch as follows:

1. Source page and line number of 01071.

2. Source OP code of SW for set word mark.

3. Source A operand address of 0001.

4. Enters A in column 55 to specify an add type of patch.

5. 11-8-4 (asterisk) in column 56.

6. Specifies the patch-point address of 333 left-justified in columns 57-61.

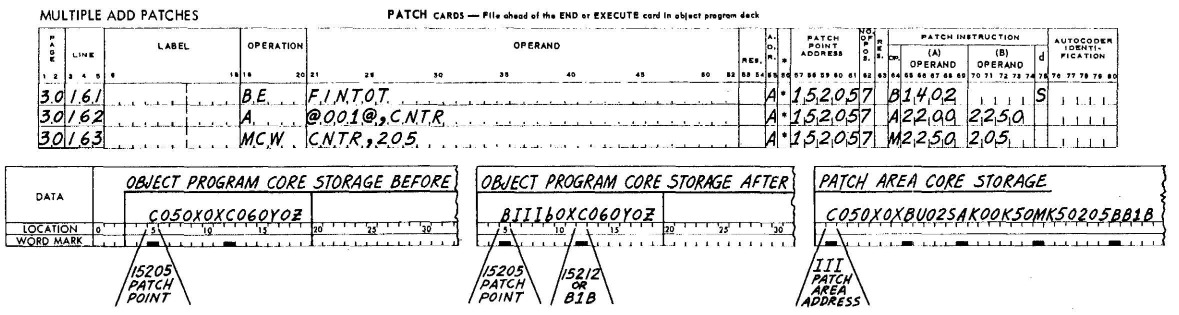
7. Specifies the length of the patch-point instruction. Seven positions in this example (, 050065).

8. Codes the actual machine-language OP code in column 64 (, in this case).

9. Codes the actual machine-language address of 001 in columns 65-69 (left-justified).

Figure 15 also shows the object program in storage: before the patch, after the patch, and the patch storage generated by Auto-Test.

See Figure 16 for an example of a multiple-add type of patch coded on an Autocoder patch code sheet.


MULTIPLE ADD PATCHES

PATCH CARDS — File ahead of the END or EXECUTE card in object program deck

PAGE 1 2	LINE 3 4 5	LABEL 6 15	OPERATION 16 20	OPERAND 21 52	RES. 53 54	A.D.R. 55	* 56	PATCH POINT ADDRESS 57-61	NO. OF POS. 62	RES. 63	OP. 64	(A) OPERAND 65-69	(B) OPERAND 70-74	d 75	AUTOCODER IDENTI-FICATION 76-80
30	161		BE	FINTOT		A	*	15205	7		B	1402		S	
30	162		A	@001@,CNTR		A	*	15205	7		A	2200	2250		
30	163		MCW	CNTR,205		A	*	15205	7		M	2250	205		

Figure 16. Multiple Add Type of Patch

REPLACE PATCH

General Description. Any instruction in object program core storage may be replaced by a replace patch. There are three types of replace patches possible:

1. Replace patch instruction with the same number of positions as the instruction at the patch point being replaced (Figure 17).

2. Replace patch instruction with a smaller number of positions than the instruction at the patch point being replaced (Figure 18).

3. Replace patch instruction with a larger number of positions than the instruction at the patch point being replaced. This type of replace patch requires that the patch-point instruction be four or more positions in length to allow for the insertion of a branch to the patch area (Figure 19).

For example, if the programmer desires to replace instructions at patch points specified in the object program:

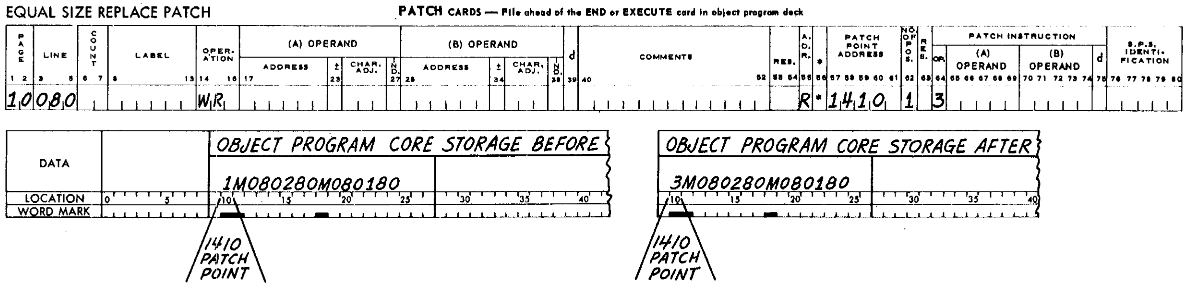
1. The replace instruction is coded in the source language of the program.

2. The patch code R in column 55 and an asterisk (11-8-4) in column 56 are specified.

3. The patch point is specified as the instruction (OP code) address of the instruction in the object program to be replaced.

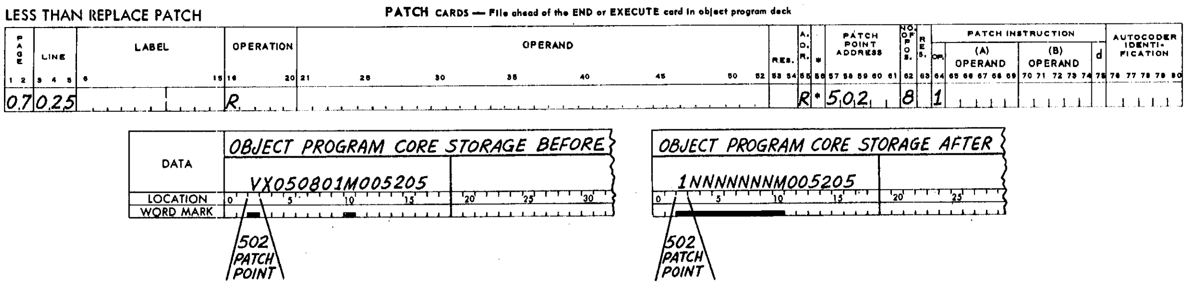
4. The number of positions in the patch-point instruction which is to be replaced is punched in column 62.

5. The actual machine-language OP code of the replace patch instruction is written in column 64, A operand in column 65-69, B operand in column 70-74, and the d modifier in column 75.



EQUAL SIZE REPLACE PATCH
PATCH CARDS — File ahead of the END or EXECUTE card in object program deck
10080 WR R * 1410 1 3
OBJECT PROGRAM CORE STORAGE BEFORE
1M080280M080180
OBJECT PROGRAM CORE STORAGE AFTER
3M080280M080180
DATA
LOCATION
WORD MARK
1410 PATCH POINT

Figure 17. Equal Size Replace Patch



LESS THAN REPLACE PATCH
PATCH CARDS — File ahead of the END or EXECUTE card in object program deck
07025 R R * 502 8 1
OBJECT PROGRAM CORE STORAGE BEFORE
VX050801M005205
OBJECT PROGRAM CORE STORAGE AFTER
1NNNNNNNM005205
DATA
LOCATION
WORD MARK
502 PATCH POINT

Figure 18. Less Than Replace Patch

GREATER THAN REPLACE PATCH

PATCH CARDS — File ahead of the END or EXECUTE card in object program deck

					(A) OPERAND				(B) OPERAND												PATCH INSTRUCTION				
PAGE 1 2	LINE 3 5	COUNT 6 7	LABEL 8 13	OPERATION 14 16	ADDRESS 17	± 23	CHAR. ADJ.	IND. 27	ADDRESS 28	± 34	CHAR. ADJ.	IND. 38	d 39	COMMENTS 40 52	RES. 53 54	A.D.R. 55	* 56	PATCH POINT ADDRESS 57 58 59 60 61	NO OF POS. 62	RES. 63	OP. 64	(A) OPERAND 65 66 67 68 69	(B) OPERAND 70 71 72 73 74	d 75	S.P.S. IDENTIFICATION 76 77 78 79 80
06	090			SW	0001				0005							R	*	333	4		,	001	005		

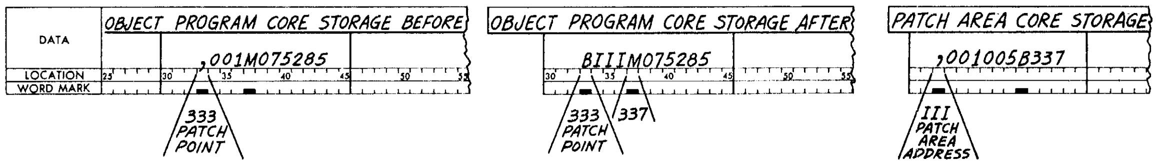


Figure 19. Greater Than Replace Patch

The Auto-Test system:

1. Determines the size of the replace patch instruction and compares it with the number of positions in the patch-point instruction which is to be replaced.

2. Loads equal size replace patch instructions over the original instruction in object program core storage (Figure 17).

3. Loads replace patch instructions with a smaller number of positions than the original instruction to the patch-point area in core storage (Figure 18), and fills the remaining positions with one-position NOP (N) instructions.

4. Loads a replace patch instruction with a larger number of positions into the patch area of core storage, an unconditional branch to the patch area over the original object program instruction being replaced, and an unconditional branch back to the next sequential instruction in the object program (Figure 19).

DC OR DCW PATCH

General Description. Additional constants, DCW (with word mark in the high-order position) or DC (without word marks) may be added to any specified patch point in core storage not already being used either by the object program or by Auto-Test. However, constants cannot be corrected in, or added to, the original object program area.

For example, if the programmer desires to add several constants to the original object program (see Figures 20 and 21):

1. The constant is coded in the source language of the program (SPS or Autocoder). Note: Count must be placed in columns 6-7 for SPS.

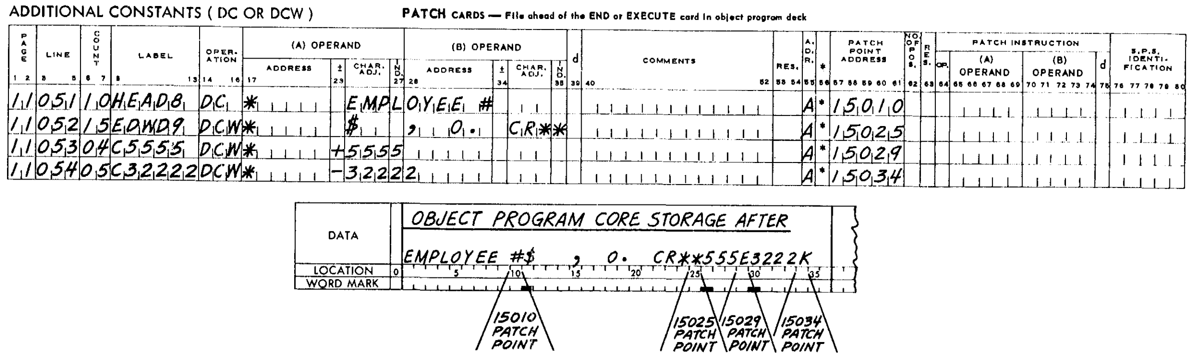
2. The patch code A or R in column 55 and the asterisk (11-8-4) in column 56 are specified.

3. The patch-point address is specified as the units position or low-order address desired for the constant.
4. Columns 62 and 75 are blank.
5. Columns 3-12 of the Auto-Test program control cards are changed. For programs in which the Auto-Test area of storage is not relocated, the highest positions of storage used by the object program <u>must</u> be increased by the number of constant positions being added. For programs in which the Auto-Test area of storage is relocated, the low limit <u>must</u> be increased or the high limit decreased to reflect the number of positions of constants added.

DC's or DCW's are limited to 32 positions. This is because the Auto-Test patch coding sheet has only 32 positions available.

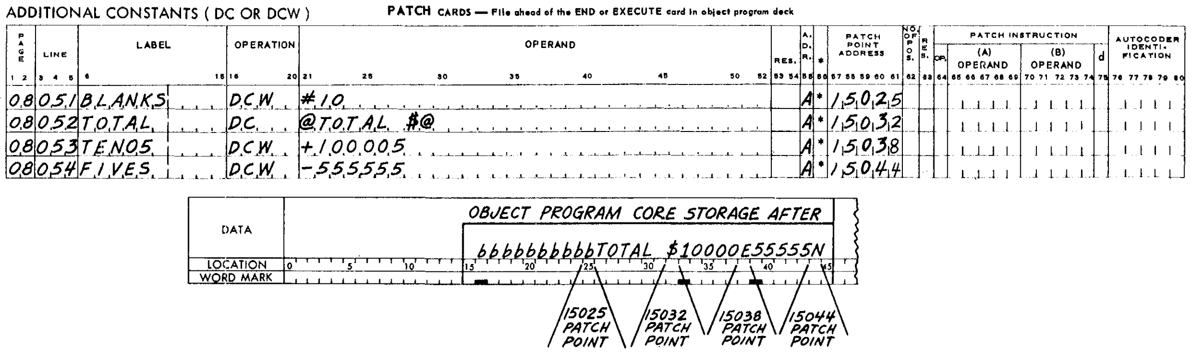
> Note: A larger constant can be patched by using a combination of DC's and DCW's.

Auto-Test cannot handle an unsigned numeric constant coded in numeric form (for example, DCW 5). However, an unsigned numeric constant can be obtained using Auto-Test by coding it as an alphameric constant (for example, DCW @5@).


ADDITIONAL CONSTANTS (DC OR DCW)   PATCH CARDS — File ahead of the END or EXECUTE card in object program deck

PAGE 1 2	LINE 3 5	COUNT 6 7	LABEL 8 13	OPERATION 14 16	(A) OPERAND ADDRESS 17	± 23	CHAR. ADJ.	IND. 27	(B) OPERAND ADDRESS 28	± 34	CHAR. ADJ.	IND. 38	d 39	COMMENTS 40 52	RES. 53 54	A.D.R. 55	* 56	PATCH POINT ADDRESS 57 58 59 60 61	NO. OF POS. 62	RES. 63	OP. 64	PATCH INSTRUCTION (A) OPERAND 65 66 67 68 69	(B) OPERAND 70 71 72 73 74	d 75	S.P.S. IDENTI-FICATION 76 77 78 79 80
11	051	10	HEAD8	DC	*		EMP	L	OYEE #							A	*	15010							
11	052	15	EDWD9	DCW	*		$		, 0.		CR*	*				A	*	15025							
11	053	04	C5555	DCW	*	+	555	5								A	*	15029							
11	054	05	C32222	DCW	*	-	322	2	2							A	*	15034							

Figure 20. Additional Constants (DC or DCW) Patch


ADDITIONAL CONSTANTS (DC OR DCW)   PATCH CARDS — File ahead of the END or EXECUTE card in object program deck

PAGE 1 2	LINE 3 4 5	LABEL 6 15	OPERATION 16 20	OPERAND 21 52	RES. 53 54	A.D.R. 55	* 56	PATCH POINT ADDRESS 57 58 59 60 61	NO. OF POS. 62	RES. 63	OP. 64	PATCH INSTRUCTION (A) OPERAND 65 66 67 68 69	(B) OPERAND 70 71 72 73 74	d 75	AUTOCODER IDENTI-FICATION 76 77 78 79 80
08	051	BLANKS	DCW	#10		A	*	15025							
08	052	TOTAL	DC	@TOTAL $@		A	*	15032							
08	053	TENOS	DCW	+100005		A	*	15038							
08	054	FIVES	DCW	-555555		A	*	15044							

Figure 21. Additional Constants (DC or DCW) Patch

The Auto-Test System:

1. Checks for DC or DCW on the source coding side of the patch card.
2. Loads DCW constants to the core storage address specified as the patch point.

3. Moves DC constants to the core storage address specified as the patch point.

4. For Autocoder constant statements, analyzes columns 21-52 and sets up the constant as specified in Autocoder language.

ASTERISK (11-8-4) OPERANDS

General Description. The use of an * for the A or B address of an actual patch instruction makes it possible to address a position of storage within the patch areas. As in SPS or Autocoder, the * is the equivalent of the units position or low-order position of the instruction in which it is used. The * operand may be character-adjusted plus (+) by specifying the number of positions desired (*+003) or character-adjusted minus (-) by using a 16K complement figure (*+I9I). Asterisk operands may not be indexed.

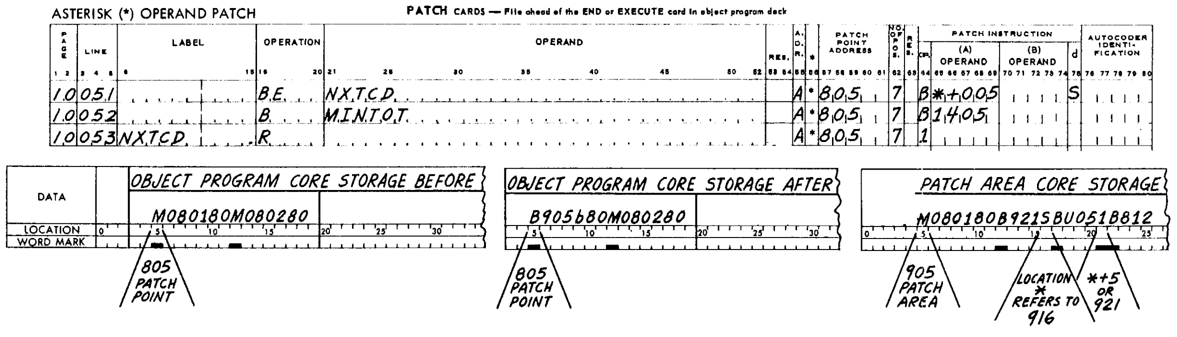
The character adjustment on asterisk operands must be three positions in length with leading zeros inserted if necessary. To character-adjust an operand by +5, punch *+005 in the appropriate columns. Do not punch *+5 which would result in an adjustment of +500. If a 4K machine is being used, a 4K complement figure must be used for negative character adjustment.

For example, if the programmer specifies multiple-add patch instructions such as subroutines and desires to branch to an instruction within this series of patch instructions, then an asterisk (11-8-4) in column 65, a + in column 66, and the character adjustment desired in columns 67-69 are specified (Figures 22 and 23).

The Auto-Test system:

1. Sets up the multiple patch and linkage from and back to the object program.

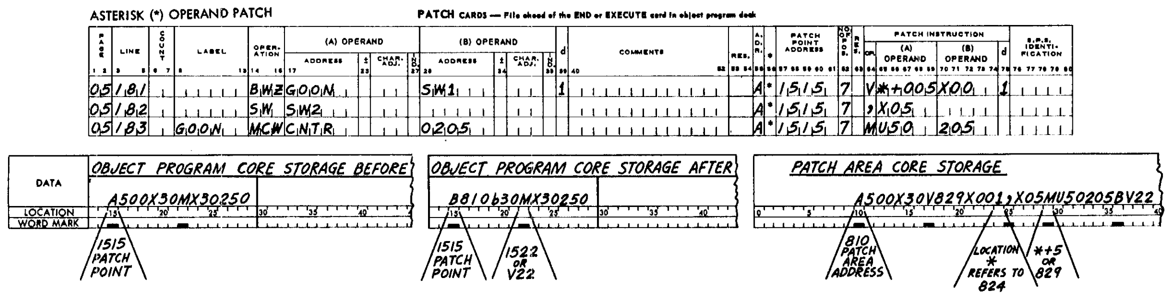
2. Adds or modify-adds the units position or low-order address of the instruction in which the asterisk (11-8-4) operand appears to the character adjustment specified in the patch operand to set up the desired A or B address (Figures 22 and 23).


ASTERISK (*) OPERAND PATCH

PATCH CARDS — File ahead of the END or EXECUTE card in object program deck

PAGE 1 2	LINE 3 4 5	LABEL 6 15	OPERATION 16 20	OPERAND 21 25 30 35 40 45 50 52	RES. 53 54	A.D.R. 55	* 56	PATCH POINT ADDRESS 57 58 59 60 61	NO. OF POS. 62	RES. 63	OP. 64	PATCH INSTRUCTION (A) OPERAND 65 66 67 68 69	PATCH INSTRUCTION (B) OPERAND 70 71 72 73 74	d 75	AUTOCODER IDENTI-FICATION 76 77 78 79 80
10	051		BE	NXTCD		A	*	805	7		B	*+005		S	
10	052		B	MINTOT		A	*	805	7		B	1405			
10	053	NXTCD	R			A	*	805	7		1				

Figure 22. Asterisk (*) Operand Patch



ASTERISK (*) OPERAND PATCH

PATCH CARDS — File ahead of the END or EXECUTE card in object program deck

PAGE	LINE	COUNT	LABEL	OPERATION	(A) OPERAND ADDRESS	±	CHAR. ADJ.	IND.	(B) OPERAND ADDRESS	±	CHAR. ADJ.	IND.	d	COMMENTS	RES.	A.D.R.	*	PATCH POINT ADDRESS	NO. OF POS.	RES.	PATCH INSTRUCTION OP	PATCH INSTRUCTION (A) OPERAND	PATCH INSTRUCTION (B) OPERAND	d	S.P.S. IDENTIFICATION
05	181			BWZ	GOON				SW1				1			A	*	1515	7		V	*+005	X00	1	
05	182			SW	SW2											A	*	1515	7		,	X05			
05	183		GOON	MCW	CNTR				0205							A	*	1515	7		M	U50	205		

Figure 23. Asterisk (*) Operand Patch

SUBROUTINE INSERTION

General Description. The patch feature of Auto-Test makes it possible to insert one or more store B register subroutines in the patch area of core storage and branch to the subroutines from several points in the object program. However, all branches to the inserted subroutines must be added to the object program.

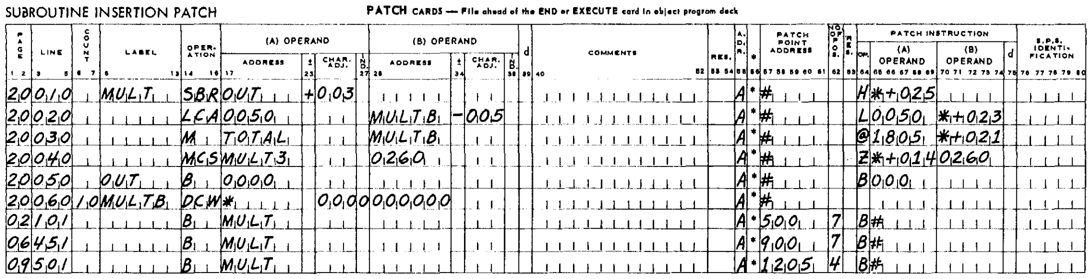
For example, if the programmer desires to add a subroutine of several instructions to the original object program without reassembly (Figure 24):

1. Add patches as described above are specified by placing the pound sign, #, (8-3 punch) in column 57 as the patch-point address of all instructions in the subroutine.

2. Add patches are coded specifying as patch-point address the instruction (OP code) address of the object program instruction after which a branch to the subroutine is to be added.

3. A B in column 64, and a pound sign (8-3) in column 65 are specified.

4. All branch to # add patches are placed directly after the subroutine patch cards.


SUBROUTINE INSERTION PATCH

PATCH CARDS — File ahead of the END or EXECUTE card in object program deck

PAGE	LINE	COUNT	LABEL	OPERATION	(A) OPERAND ADDRESS	±	CHAR. ADJ.	IND.	(B) OPERAND ADDRESS	±	CHAR. ADJ.	IND.	d	COMMENTS	RES.	A. D. R.	*	PATCH POINT ADDRESS	NO. OF POS.	RES.	OP.	PATCH INSTRUCTION (A) OPERAND	PATCH INSTRUCTION (B) OPERAND	d	S.P.S. IDENTIFICATION
1 2	3 5	6 7	8 13	14 16	17	23		27	28	34		38	39	40 52	53 54	55	56	57 58 59 60 61	62	63	64	65 66 67 68 69	70 71 72 73 74	75	76 77 78 79 80
20	010		MULT	SBR	OUT	+	003									A	*	#			H	*+025			
20	020			LCA	0050				MULTB	-	005					A	*	#			L	0050	*+023		
20	030			M	TOTAL				MULTB							A	*	#			@	1805	*+021		
20	040			MCS	MULT3				0260							A	*	#			Z	*+014	0260		
20	050		OUT	B	0000											A	*	#			B	000			
20	060	10	MULTB	DCW	*		0000000000									A	*	#							
02	101			B	MULT											A	*	500	7		B	#			
06	451			B	MULT											A	*	900	7		B	#			
09	501			B	MULT											A	*	1205	4		B	#			

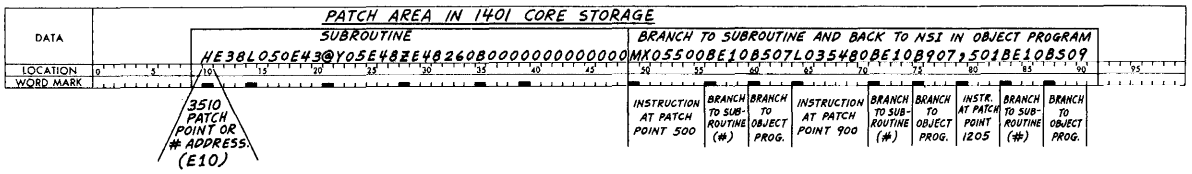

PATCH AREA IN 1401 CORE STORAGE
DATA
LOCATION
WORD MARK
SUBROUTINE
HE38L050E43@Y05E48ZE48260B0000000000000
BRANCH TO SUBROUTINE AND BACK TO NSI IN OBJECT PROGRAM
MX05500BE10B507L03548OBE10B907,501BE10B509
0
5
10
15
20
25
30
35
40
45
50
55
60
65
70
75
80
85
90
95
3510 PATCH POINT OR # ADDRESS (E10)
INSTRUCTION AT PATCH POINT 500
BRANCH TO SUBROUTINE (#)
BRANCH TO OBJECT PROG.
INSTRUCTION AT PATCH POINT 900
BRANCH TO SUBROUTINE (#)
BRANCH TO OBJECT PROG.
INSTR. AT PATCH POINT 1205
BRANCH TO SUBROUTINE (#)
BRANCH TO OBJECT PROG.

Figure 24. Subroutine Insertion Patch

The Auto-Test system:

1. Checks for pound sign (8-3) patch point in column 57 and loads subroutine patch instructions into the first free position of the patch area of core storage. The instruction (OP code) address of the first instruction of the subroutine is stored for later reference.

2. Sets up add patches with B to pound sign (8-3) as linkage to the subroutine; if desired, the pound sign operand may be character-adjusted by specifying (#+xxx). The # is equal to the instruction (OP code) address of the first instruction of the subroutine and is available for reference until a second subroutine is inserted.

LOADABLE PATCH CARDS

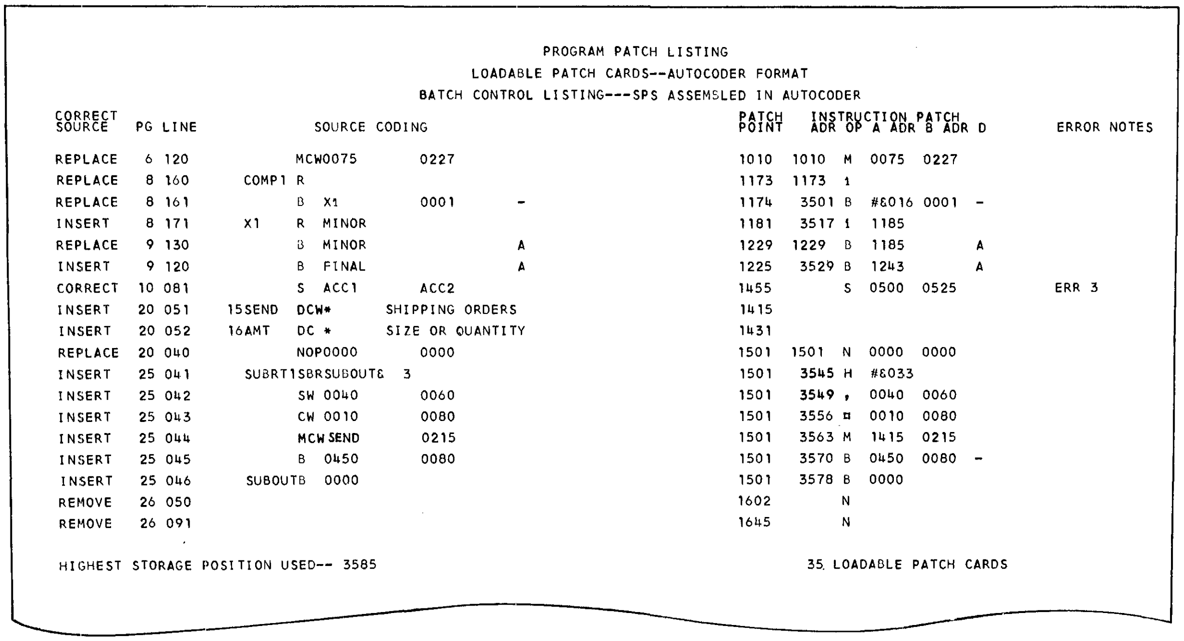
This feature of automatic patching provides for punching loadable patch cards from the original "source" patch cards so that a tested program can be run independently of Auto-Test without a reassembly.

After the 80/80 listing of the punched output and selected input cards at the end of the Auto-Test run, the following message will be printed and displayed:

```
INSTRUCTIONS
TO PUNCH LOADABLE PATCH CARDS,
SET SENSE SWITCH F ON
PLACE PATCH DECKS IN READ HOPPER--DEPRESS START
```

At this time loadable patch cards may be punched out for programs which have been successfully run with source patch cards under Auto-Test control.

The group of source patch cards should be removed from the object program deck and the Auto-Test program control card placed in front of the group. Multiple groups of source patch cards, each preceded by its Auto-Test program control card, may be placed in the read hopper and processed continuously to produce multiple groups of loadable patch cards and a final patch listing for each group of patch cards (Figure 25).



```
                                                      PROGRAM PATCH LISTING
                                                LOADABLE PATCH CARDS--AUTOCODER FORMAT
                                          BATCH CONTROL LISTING---SPS ASSEMSLED IN AUTOCODER
CORRECT                                                                    PATCH   INSTRUCTION PATCH
SOURCE    PG LINE           SOURCE CODING                                  POINT   ADR OP A ADR B ADR D           ERROR NOTES

REPLACE    6 120            MCW0075      0227                              1010  1010 M  0075  0227
REPLACE    8 160       COMP1 R                                             1173  1173 1
REPLACE    8 161            B  X1        0001       -                      1174   3501 B  #&016 0001  -
INSERT     8 171       X1   R  MINOR                                       1181   3517 1  1185
REPLACE    9 130            B  MINOR                A                      1229  1229 B  1185        A
INSERT     9 120            B  FINAL                A                      1225   3529 B  1243        A
CORRECT   10 081            S  ACC1      ACC2                              1455        S  0500  0525              ERR 3
INSERT    20 051      15SEND DCW*      SHIPPING ORDERS                     1415
INSERT    20 052      16AMT  DC *      SIZE OR QUANTITY                    1431
REPLACE   20 040            NOP0000      0000                              1501  1501 N  0000  0000
INSERT    25 041       SUBRT1SBRSUBOUT&  3                                 1501   3545 H  #&033
INSERT    25 042            SW 0040      0060                              1501   3549 ,  0040  0060
INSERT    25 043            CW 0010      0080                              1501   3556 ¤  0010  0080
INSERT    25 044            MCW SEND     0215                              1501   3563 M  1415  0215
INSERT    25 045            B  0450      0080                              1501   3570 B  0450  0080  -
INSERT    25 046       SUBOUTB 0000                                        1501   3578 B  0000
REMOVE    26 050                                                           1602        N
REMOVE    26 091                                                           1645        N

HIGHEST STORAGE POSITION USED-- 3585                                           35. LOADABLE PATCH CARDS
```

Figure 25. Program Patch Listing — Loadable Patch Cards

Each group of loadable patch cards punched is headed up by a patch packet identification card which may be left with the packet when it is placed just ahead of the end or execute card in the object program replacing the source patch card group.

When multiple patch decks are processed, the first loadable patch deck will fall in the punch normal stacker, the second in stacker 4, and the third in the 8/2 stacker. Additional groups processed will fall into punch normal, 4, 8/2, and continue to rotate stackers until all groups have been processed. It is suggested that individual packets of loadable patch card groups be removed from the stackers as they are punched. Each group of cards should be interpreted for proper program identification.

Note: Object program decks with loadable patch cards may be run under Auto-Test control. However, columns 3-7 of the Auto-Test program control card must be changed to the new highest core storage position used. This new high core position is indicated on the loadable patch listing.

When testing is being done on a 4K machine, source patch cards with addresses referring to core storage locations above 4,000 cannot be used as input cards for the Punch Loadable Patch Cards Option of Auto-Test. This is because modify-address instructions are changed to add instructions when the option is executed on a 4K machine.

Autocoder format cards have:

1. Patch sequence number in columns 73-75, and an 11 zone (X) punch in column 72 for sort separation of patch cards from object program cards.

2. Symbolic identification in columns 76-80 from the Auto-Test program control card.

3. Source patch page and line number in columns 35-39.

The patch packet identification card numbered 000 (columns 73-75) has the program identification punched in columns 1-39.

SPS format cards have:

1. A patch-deck sequence number in columns 7-9.

2. An 11 zone (X) punch in column 1 for sort separation.

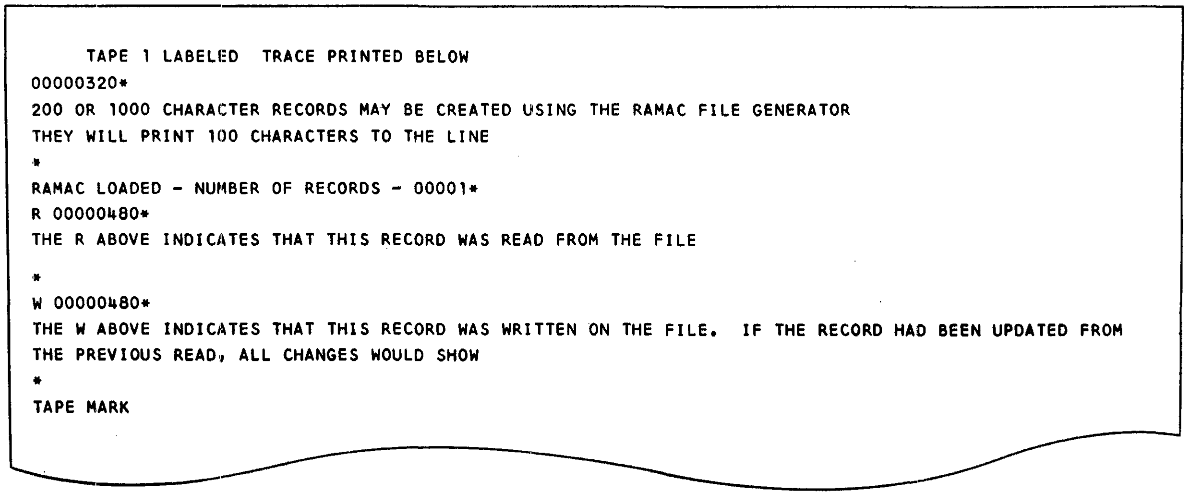
3. Symbolic identification in columns 76-80 from the Auto-Test program control card.

4. Source patch page and line number in columns 2-6.

The patch packet information card numbered 000 (columns 7-9) has program identification punched in columns 10-55.

RAMAC File Trace

Auto-Test provides the option for a trace of the data which is read from or written on the disk file during the execution of the object program. The trace utilizes a tape drive that is not being used by the object program. On this tape will be written in the MOVE mode the disk address used in the execution of the RAMAC instruction, an indication of whether it was a read (R) or a write (W) instruction, and in the LOAD mode the actual data read or written. The tape will be printed on the 1403 automatically at the end of the test (Figure 26). The tape will also contain the data written on the disk by the RAMAC File Generator if that feature was used. The RAMAC File Trace subroutine requires 122 positions of core storage during the execution of the user's object program. These positions will be loaded automatically into the Auto-Test storage area just before the execution of the object program. If there is not sufficient room in the Auto-Test storage area for the 122-position subroutine, the object program will still be tested but no trace of the RAMAC data will occur.



```
     TAPE 1 LABELED  TRACE PRINTED BELOW
00000320*
200 OR 1000 CHARACTER RECORDS MAY BE CREATED USING THE RAMAC FILE GENERATOR
THEY WILL PRINT 100 CHARACTERS TO THE LINE
*
RAMAC LOADED - NUMBER OF RECORDS - 00001*
R 00000480*
THE R ABOVE INDICATES THAT THIS RECORD WAS READ FROM THE FILE

*
W 00000480*
THE W ABOVE INDICATES THAT THIS RECORD WAS WRITTEN ON THE FILE.  IF THE RECORD HAD BEEN UPDATED FROM
THE PREVIOUS READ, ALL CHANGES WOULD SHOW
*
TAPE MARK
```

Figure 26. RAMAC File Trace Printout

To utilize this feature the programmer must:

1. Select a tape not used by the object program for the trace output by punching a T in card column 2, 16, 30, 44, or 58 of the tape control card. The columns labeled Print, Marks and Symbolic Tape Label will be filled in automatically by Auto-Test and can be left blank in the card.

2. Assemble in the object program a four-position NOP (N000) instruction immediately after every RAMAC read or write instruction that is to be traced. RAMAC instruction and the NOP must be in core sequence. It must not be patched into the program by some other method.

The two conditions above will cause the 122-position trace subroutine to be loaded into core storage. The four-position NOP instruction will be changed to an unconditional branch to the 122-position trace subroutine, which will trace the instruction, write the information on tape and give control back to the object program. If the NOPs have been assembled in the program but no tape has been specified, the object program will be tested but no trace of the RAMAC instructions will occur. If it is desired to print the data written on the disk by the RAMAC File Generator, but not to trace the actual RAMAC instruction, only the T needs to be punched in the tape control card, and the four-position NOPs are not necessary in the object program.

Note: There can be only one trace tape per program — only one T can be punched in the tape control card. If it is desired to use both the RAMAC File Trace feature and the Snapshot feature during the same program test, the output from both of these features will be written on the one tape.

Snapshot Feature

The Snapshot feature of Auto-Test provides a means for taking a printout of core storage between specified limits at one or several points in the object program during its execution.

The Snapshot is a 278-position subroutine which is automatically inserted in the Auto-Test storage area when called for by the user. When branched to during the execution of the object program, the selected area of core storage is written on the tape unit which has been specified as the trace tape in the tape control card. The maximum size of the core snapshot is limited by the maximum size tape record which may be printed by the Auto-Test tape print program (see "Tape Printout," page 40). Object program core storage is restored to its original condition after the execution of the snapshot routine.

After the execution of the object program being tested and the regular Auto-Test core storage print, the trace tape will be printed. The printout of the trace will be in blocks of 100. Therefore, it is recommended that the lower limits of the trace be the first position after an even hundred (501, 2301, etc.). The first line printed by the snapshot will be the three-position address of the instruction in the object program at which the snapshot was called for and the lower limit of the snapshot — for example, X05 — 01001*. The line following the snapshot print will indicate the upper limit of the snapshot — for example 03000*.

For example, if the programmer desires to take a snapshot of core storage from 001 through 500 at several points during the execution of the object program:

1. A snapshot is called for by coding SNAP in columns 57-60, * in column 56, lower core limit in three-, four- or five-position address form in columns 65-69, and upper core limit in three-, four- or five-position address form in columns 70-74 (Figure 27).

2. The points are specified in the object program at the desired branch to the snapshot subroutine by coding an A in column 55, * in column 56, the four- or five-position patch-point address of the OP code of the instruction after which the snapshot is to be taken in columns 57-61, the number of positions in the instruction (minimum of four), a B in column 64 and a # in column 65 (Figure 27).

   A patch-point instruction cannot be specified between a compare instruction and the corresponding test for the status of the compare indicators. The reason is that the snapshot subroutine uses the compare instruction and thus resets the compare indicators.

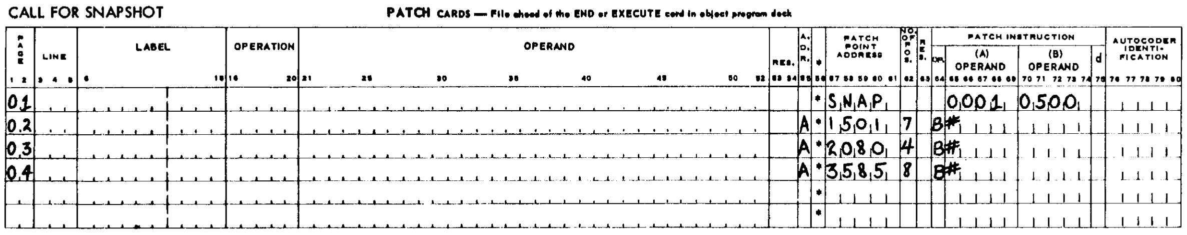
   Note: The snapshot must not include the Auto-Test storage area.

3. The tape unit to be used by the snapshot subroutine is specified by punching a T in the proper tape unit column in the tape control card. The columns labeled Print, Marks, and Symbolic Tape Label will be filled in automatically by Auto-Test and can be left blank.

4. Call for snapshot cards and B# (Branch to #) cards <u>must</u> precede patch cards if the program is being patched.

Note: Normally only one snapshot subroutine can be inserted in each program. It is possible, however, to insert a separate snapshot for each EXecute in addition to the one main-line snapshot.

See Figure 27 for an example of a call for snapshot.



CALL FOR SNAPSHOT

PATCH CARDS — File ahead of the END or EXECUTE card in object program deck

PAGE 1 2	LINE 3 4 5	LABEL 6 15	OPERATION 16 20	OPERAND 21 … 52	RES. 53 54	A.D.R. 55	* 56	PATCH POINT ADDRESS 57 58 59 60 61	NO. OF POS. 62	RES. 63	OP. 64	PATCH INSTRUCTION (A) OPERAND 65 66 67 68 69	PATCH INSTRUCTION (B) OPERAND 70 71 72 73 74	d 75	AUTOCODER IDENTIFICATION 76 77 78 79 80
01							*	SNAP				0001	0500		
02						A	*	1501	7		B	#			
03						A	*	2080	4		B	#			
04						A	*	3585	8		B	#			
							*								
							*								

Figure 27. Snapshot Patch Cards

The Auto-Test system:

1. Checks the Auto-Test storage area and inserts the modified snapshot routine in core if there are enough available positions. If there are not enough positions to insert the subroutine and branch to it at least one time, a message is printed as shown below, and the program is tested without the snapshot feature.

```
PATCH AREA LIMITS EXCEEDED, NO TRACE
```

2. Processes the branch to pound sign (8-3) cards to insert linkages from the object program to the snapshot subroutine.

Note: The snapshot may include the Auto-Test storage area but the high-limit position specified must be a position in the object program storage area. Also, there can be only one trace tape per program — that is, only one T can be punched in the tape control card. If it is desired to use both the RAMAC File Trace and Snapshot features during the same program test, the output from both of these features will be written on the same tape.

Core Storage Printout

1. Auto-Test determines the size of the core storage printout:

   a. If the programmer has specified that the Auto-Test storage area is relocated, then machine size core is printed out.

   b. If a Disaster Restart situation has occurred, then machine size core is printed out.

   c. If the object program under test is being run without relocation of the Auto-Test storage area, then all of core used by the object program plus program patches is printed out.

2. In all cases the core storage printout will be either 1.4, 2, 4, 8, 12, or 16K size.

3. Core storage location 200 never shows on the printout.

4. The three highest positions of core in the machine will be changed by the core storage printout.

5. Core storage locations 201 through 332 are printed as the first line of the printout.

6. All group marks in the object program under test are changed to a lozenge (12-8-4) so that they will show on the printout.

7. Each 100-position area of core storage is identified by both an alphabetic and numerical address notation — for example,

```
01301 THRU 01400  T01 THRU U00.
```

8. The last line of the printout will show the status of the sense switches as they were while the test was being run — for example,

```
THE FOLLOWING SENSE SWITCHES WERE ON WHILE
THE PROGRAM WAS BEING TESTED, A,B,D.
```

Tape Printout

The Auto-Test printout for tape will print an exact picture of any tape file, including blocked or unblocked, fixed or variable length records, header and trailer labels. If the tape has word marks, they will print.

1. The group mark at the end of the record prints as a lozenge (12-8-4). Any other group marks within the tape record will print as blanks.

2. When a tape mark is sensed, the words TAPE MARK will print. Each tape may have up to nine tape marks. Each tape should end with a tape mark.

3. No physical description of the tape or separate control card is needed for printing. To print a tape, the programmer indicates on the Auto-Test tape control card that he wants a tape printed by punching a P in columns 3, 17, etc.

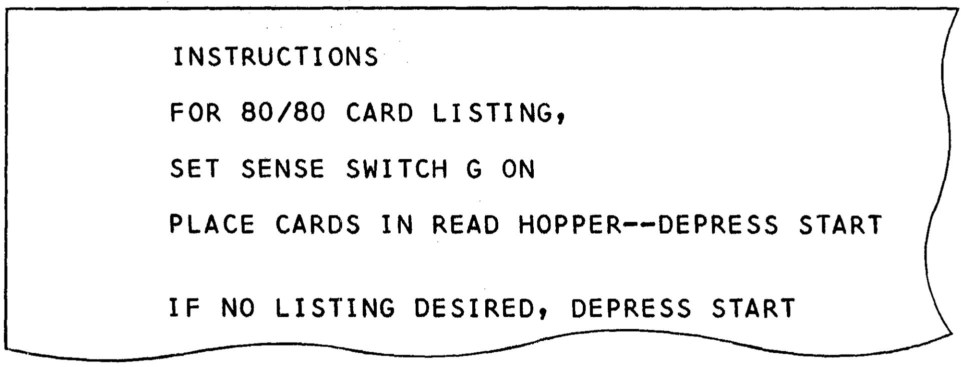
4. Each tape printout is identified by both the label and drive number.

5. The printing normally stops at the number of tape marks specified on the tape control card. If the program under test does not write the final tape mark (possibly because of a process error), Auto-Test will terminate the printing after the tape has been printed, indicating how many tape marks were specified and how many were actually found.

6. The maximum record length that can be printed on a 4K 1401 is 2,475 positions. If a record exceeds this size, the remaining portion of the record will not print. A message identifies these records. For an 8, 12, or 16K machine, the maximum record size is 6,475, 10,475, and 14,475 respectively.

80/80 Listing of Output Cards

At the beginning of each object program test, each card stacker (other than normal read and stacker 1) is identified with a stacker identification card. This card contains the Auto-Test test number, stacker number, and the name of the program as punched in the program control card. The Auto-Test program control card is selected to pocket 1 and serves as the stacker identification card for that pocket. At the end of the complete test session, the instructions will appear on the printer as shown in Figure 28.



```
INSTRUCTIONS

FOR 80/80 CARD LISTING,

SET SENSE SWITCH G ON

PLACE CARDS IN READ HOPPER--DEPRESS START

IF NO LISTING DESIRED, DEPRESS START
```

Figure 28. Instructions — 80/80 Listing of Output Cards

Executing these instructions will give an 80/80 listing of the output cards as well as those data cards which have been selected to pockets 1 and 2. Each test will print on a separate page if there are cards to be listed for that test. The listing will be completely identified as to program and pocket. During the 80/80 list, the program control card which identifies pocket 1 will be selected to pocket 1 so that it may easily be put back in the test deck for further testing use.

OPERATOR INSTRUCTIONS

Testing with Auto-Test

1. To ready the 1401 system for Auto-Test:

   a. Mount the Auto-Test systems tape on drive 1.

   b. Turn off all sense switches except A and the I/O Check Stop Switch.

   c. Clear the card reader.

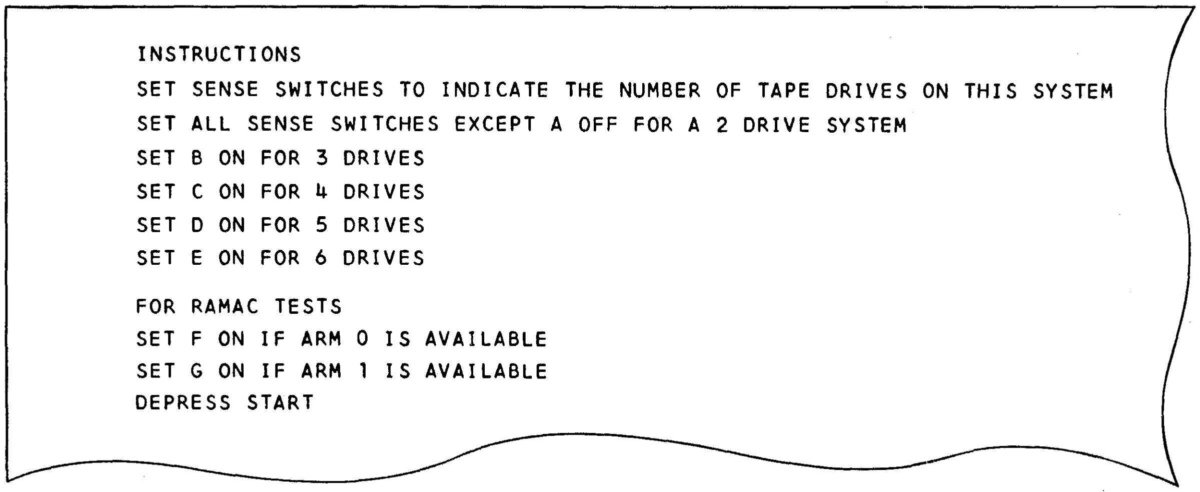
   d. Clear the card punch. Place blank cards in the punch feed.

   e. Depress Check Reset on the 1402 Card Read Punch.

   f. Place the program decks to be tested in the Read File Feed.

   g. Depress Check Reset, Start Reset and Tape Load on the 1401.

   Figure 29 shows what will be displayed on the printer.



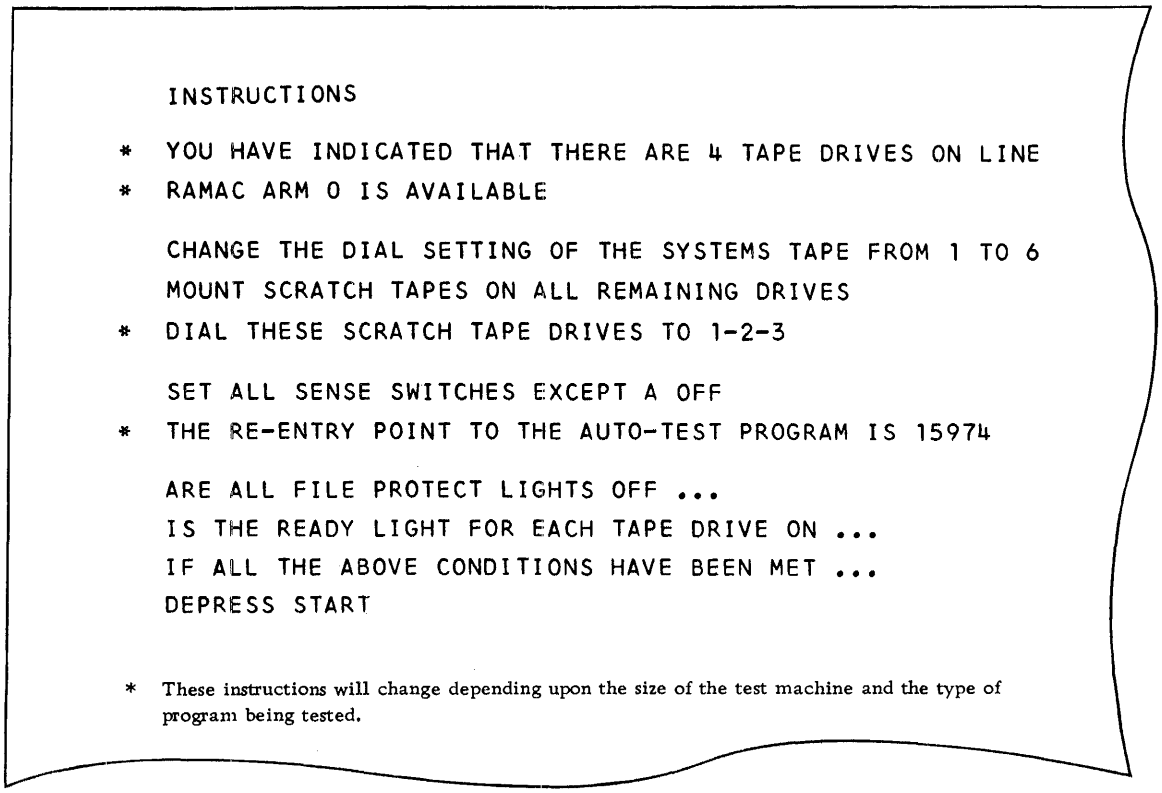
```
INSTRUCTIONS
SET SENSE SWITCHES TO INDICATE THE NUMBER OF TAPE DRIVES ON THIS SYSTEM
SET ALL SENSE SWITCHES EXCEPT A OFF FOR A 2 DRIVE SYSTEM
SET B ON FOR 3 DRIVES
SET C ON FOR 4 DRIVES
SET D ON FOR 5 DRIVES
SET E ON FOR 6 DRIVES

FOR RAMAC TESTS
SET F ON IF ARM 0 IS AVAILABLE
SET G ON IF ARM 1 IS AVAILABLE
DEPRESS START
```

Figure 29. Instructions — Initializing

   Note: If the test system has only one tape drive on line, indicate this as if there were two. However, under this situation no tape programs may be tested, nor may the RAMAC File or Snapshot Traces be utilized.

2. After these operator instructions have been executed, Figure 30 shows the instructions that will be displayed on the printer.

3. Auto-Test begins with the assumption that these instructions have been followed. The test must begin in this way in order to properly generate operating instructions for the tape drives and sense switches. When these instructions have been carried out, testing of object programs will begin.



```
     INSTRUCTIONS

*    YOU HAVE INDICATED THAT THERE ARE 4 TAPE DRIVES ON LINE
*    RAMAC ARM 0 IS AVAILABLE

     CHANGE THE DIAL SETTING OF THE SYSTEMS TAPE FROM 1 TO 6
     MOUNT SCRATCH TAPES ON ALL REMAINING DRIVES
*    DIAL THESE SCRATCH TAPE DRIVES TO 1-2-3

     SET ALL SENSE SWITCHES EXCEPT A OFF
*    THE RE-ENTRY POINT TO THE AUTO-TEST PROGRAM IS 15974

     ARE ALL FILE PROTECT LIGHTS OFF ...
     IS THE READY LIGHT FOR EACH TAPE DRIVE ON ...
     IF ALL THE ABOVE CONDITIONS HAVE BEEN MET ...
     DEPRESS START
```

* These instructions will change depending upon the size of the test machine and the type of program being tested.

Figure 30. Instructions — Initializing

4. Several blank cards should be placed after the last data card of the last program to be tested.

5. All instructions necessary to properly ready the 1401 system for each test will be automatically generated and printed. Follow the instructions sequentially.

6. If it is desired to mount precreated master tapes, instructions will be generated to mount these tapes. However, because these precreated tapes may be file-protected, Auto-Test will always unload these tapes at the end of the test and generate instructions to dismount them.

7. The restart points to Auto-Test are:

   a. If the Auto-Test storage area is <u>not</u> relocated:

      (1) 4K test system - 3974
      (2) 8K test system - 7974
      (3) 12K test system - 11974
      (4) 16K test system - 15974

   b. If the Auto-Test storage area <u>is</u> relocated, the restart point will be generated, printed and displayed on the printer for each program that utilizes this mode.

]]>

c. FARGO programs for all systems — 333. The instructions which indicate the restart point when the Auto-Test storage area is relocated or when FARGO programs are being tested will appear on the 1403 printer as:

```
SET ADDRESS DIALS TO 00333---RESTART POINT
```

d. **Tape Print Runaway occurs when the tape keeps printing past the final tape mark because there is no tape mark to stop the printing, or because the tape control card has been improperly specified.** Depress the STOP key and manually restart at:

(1)	4K system	- 3974
(2)	8K system	- 7974
(3)	12K system	- 11974
(4)	16K system	- 15974

e. Disaster Restart (see "Disaster Restarts," below).

8. A single carriage tape which will function properly for all programs being tested should be used. All channels should be punched to eliminate forms runaway. The IBM test centers have standard carriage tapes which are adequate for the majority of 1401 program tests.

Disaster Restarts

Normally, when a process error occurs during the execution of an object program, linkage to the Auto-Test program may be effected by manually dialing in to either the normal restart point for that size of test machine or to the relocated point if the programmer has elected relocation. (In this case, instructions would have been generated which would instruct the operator to set the address dials to a point other than the normal restart point.)

If, however, the program under test destroys this restart monitor, dialing in will not effect linkage with the Auto-Test program and a special restart method called Disaster Restart must be executed.

Instructions for a Disaster Restart:

1. Manually print the first 500 positions of core storage from the console.
2. Rewind and change the dial setting of the systems tape from 6 to 1.
3. Dial out any other drive which was dialed to 1 for the test.
4. Turn on sense switch B.
5. Depress Start Reset, Check Reset, and Tape Load.

At this point a message will appear on the printer which will say:

```
DISASTER RESTART--TURN OFF SENSE SWITCH B--
CHANGE THE DIAL SETTING OF THE SYSTEMS TAPE
FROM 1 TO 6--DEPRESS START
```

If a tape drive was previously set to 1, it must be dialed back to 1 at this point. The Auto-Test program has retained all the information to give a full machine size core storage dump from 500 on, print out all tapes as it would have done normally, bypass the program that caused the error, and begin automatically testing the next program.

Note: On a disaster restart, the core storage prints will not change group marks to lozenges.

Creating the Auto-Test Systems Tape

1. Mount a scratch tape on unit 1.

2. Load the Auto-Test program in the 1402 read hopper.

3. Depress the Check Reset on the 1402, Check Reset on the 1401, Start Reset on the 1401 and Load on the 1402.

4. When the program has been successfully created on tape 1, the following message will appear on the printer:

```
1401 AUTO-TEST SYSTEMS TAPE CREATION COMPLETE
DO NOT FILE PROTECT THE SYSTEMS TAPE
```

5. There is only one Auto-Test deck and it is used to create the tapes for all sizes of 1401 systems from 4K through 16K.

6. In that the systems tape is read from and written upon during the testing of an object program, IT MUST NOT BE FILE-PROTECTED.

Note: Instead of creating the systems tape from cards, it is possible to copy an already existing systems tape by using a tape copy program. It should be noted, however, that the Auto-Test systems tape contains 2 (two) tape marks.

Programmed Halts

The programmed halts in Auto-Test are divided into two groups:

1. Those which print instructions on the printer which are self-explanatory.

2. Tape read and write error halts.

   a. Depress start to try again.

   b. After about ten tries, re-create the systems tape and start the test over.

APPENDIX

Organization of Auto-Test

The Auto-Test program consists of a series of subroutines on a systems tape which are executed as required for each program. The functions of Auto-Test can be broken down into four major sections as follows:

1. Test Initialization

   a. Operator instructions for the program to be tested are generated and printed.

   b. The Tape and RAMAC File Generations are executed if needed.

2. Controlled Load

   a. The object program to be tested is written on the end of the systems tape in a condensed card image form.

   b. The changes described under "Automatic Modification to the Object Program Being Tested," page 16, occur at this time.

   c. The linkages necessary for patching are created.

   d. The trace subroutines are inserted, if needed.

   e. The complete object program is then read from the systems tape into core storage.

   Note: The Auto-Test program load instructions utilize the card read area (001-080) in loading the object program from tape to core storage; therefore instructions or constants must not be located in this area.

3. Execution

   a. The object program to be tested begins to execute.

   b. Control is transferred to the Auto-Test program either automatically or manually. It is entered automatically by reaching the final halt instruction. It is entered manually by dialing in the restart point on the console when a process error is encountered. It is restarted with the Disaster Restart procedure when the linkage has been destroyed.

4. Documentation

   The following printouts occur:

   a. The core storage print.

   b. The tape prints, if required, including the trace prints.

Control is transferred to the operator instruction generation program, which reads to the next program control card and begins the test of the next program.

Testing Hints

To avoid halting the 1401 during the test session and to speed testing as much as possible, the following are recommended:

1. Do not use precreated tapes. Normally, a tape file generated by the Auto-Test program for testing purposes can be created in less time than it takes to halt the system and mount a precreated tape.

2. To avoid halts for a change in a tape drive dial setting, do not specify in the object program a tape drive number larger than the number of drives on the test machine. For example, the Auto-Test program begins with the tape drives at drive number 6 (the systems tape) and at drive number 1, 2, 3 on a four-tape-drive system. If an object program refers to drive 4 or 5, the program halts to allow this dial change to be made.

3. Do not "save" tapes (by punching S in the tape control card) unless necessary. Saving tapes requires that the operator dismount the saved tape and mount a new reel. For file-protect purposes, precreated tapes will be saved regardless of whether the S is punched.

4. Do not relocate the Auto-Test linkage from its normal position in high core unless absolutely necessary. Relocation not only causes a machine halt for printing the new console restart point, but may, in some cases, reduce the amount of Auto-Test storage available for automatic patching.

5. If a program to be tested requires more tape drives than are on the test machine, it is possible to simulate additional drives with Auto-Test. This may be done by using automatic patching to temporarily change the unit number on the tape read, write, rewind commands, etc., so that two different data files are written on a single physical tape drive. For these output files, the object program would simply write two different types of records on the same reel. This tape print will show the information and the sequence in which it was written.

6. Wherever possible, make the final halt in the object program a four-position (or longer) command — for example, HALT & BRANCH TO HALT. This will avoid a machine halt and a manual console restart during testing.

7. A significant advantage is provided by the console status cards in that they provide the operator with immediate identification of the program being tested. Console status cards are automatically selected into the center pocket. It is recommended that the operator remove these cards as each program is processed, so that he has a complete list of the halts contained in the program as well as the name of the program being tested.

Object program instructions which will interfere with Auto-Test:

1. No reference to tape drive 6 may be made by the object program under test. These instructions will be changed to NOP by the controlled load phase of Auto-Test. If, however, the user is modifying tape unit numbers during the execution of the object program, drive 6 might be inadvertently accessed. In this case, a Disaster Restart is required. If the object program wrote on drive 6, it will be necessary to re-create the Auto-Test systems tape.

2. For RAMAC tracing, the NOP instruction which follows the RAMAC read or write must not only be the next sequential instruction in storage, but must also be the instruction which follows these reads and writes as the object program is loaded into core — that is, the cards must be in their proper sequence.

3. If a last-card test appears in an execute routine, it must be patched by a "replace patch" which contains Nxxxx. These positions shown as x must contain a valid machine address. Note: a delete patch must not be used.

4. When using Auto-Test, the normal last-card test in the object program will be changed to a test for lozenge in the card column specified by the user. This lozenge position of the read area therefore cannot be destroyed by the object program until after these simulated last-card tests have been executed.

5. Some type of test for the last card must be included in every program which reads data cards. If this is not done, the object program under test will read subsequent object programs which are stacked in the reader, rather than returning control to Auto-Test.

6. Start-Punch-Feed and Start-Read-Feed commands must be preceded by a last-card test or they may cause a machine halt while the core storage print is being executed. The last-card test (to branch around these commands) may be patched with an add patch of automatic patching.

   Note: Last-card tests that are patched using an add patch will automatically be modified to an eight-position test for the lozenge in the last data card.

Conditions under which a specified program will not be tested:

1. The program is calling for more tape drives than the test machine has on line. A notification,

   ```
   INSUFFICIENT DRIVES - NO TEST
   ```

   will be printed and Auto-Test will then search for the next program to be tested.

2. If a TFG control card was not preceded by a tape control card, the following message will print:

```
NO TAPE CONTROL CARD - NO TEST
```

That is, a tape program is being tested but there is no tape control card.

3. If an object program is not preceded by clear storage cards and/or the bootstrap card, the following message will print:

```
BOOTSTRAP CARD NOT 3D CARD IN PROGRAM--NO TEST
```

4. If the END card is missing, the following will print:

```
NO END/START CARD--NO TEST
```

5. If, while loading the object program, there is not enough room in the Auto-Test storage area for the twelve positions needed to simulate the last-card test, the following will print:

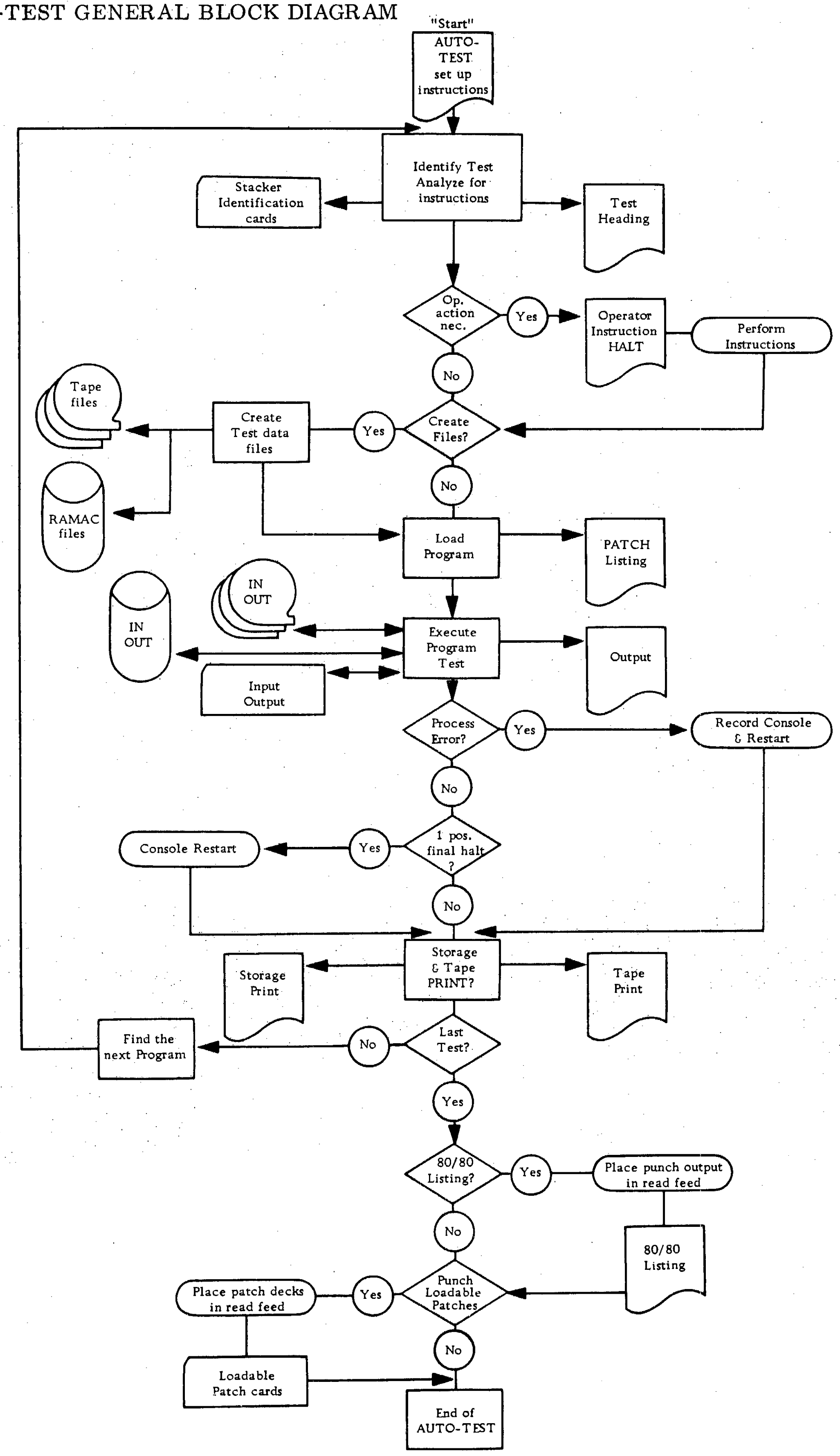
```
SIMULATION OF THE LAST CARD TEST WOULD
EXCEED THE HIGH LIMIT OF CORE--NO TEST
```

   This must be done because without the proper last-card-test simulation, the object program being tested will read past the last data card and attempt to process the next program.

6. If a program to be tested is not preceded by a program control card, Auto-Test will keep reading cards until it finds the next program control card. This means that the whole program, TFG cards, data cards, etc., will be bypassed until a program control card is found.

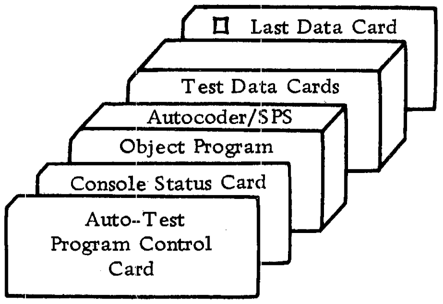
Note: Both object-program cards and data cards cannot have an asterisk in card column 1 because they would be recognized as program control cards in a bypass situation.

AUTO-TEST GENERAL BLOCK DIAGRAM

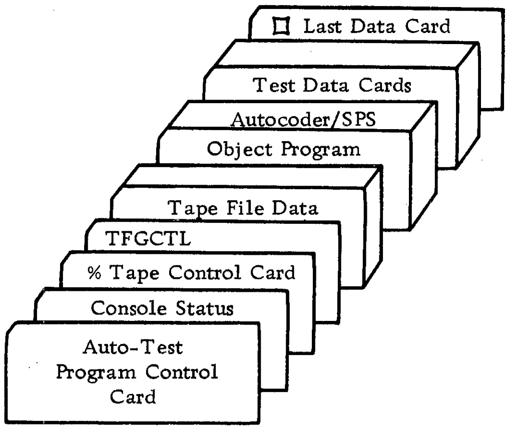


AUTO-TEST PROGRAM TEST INPUT SEQUENCE

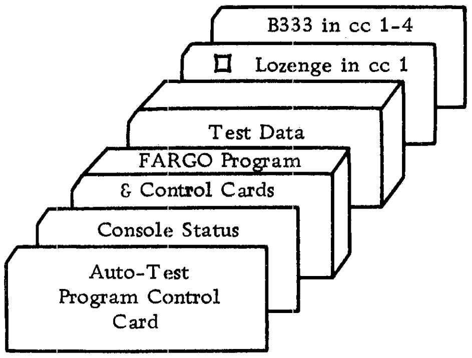
A. Card Program



B. Tape Program

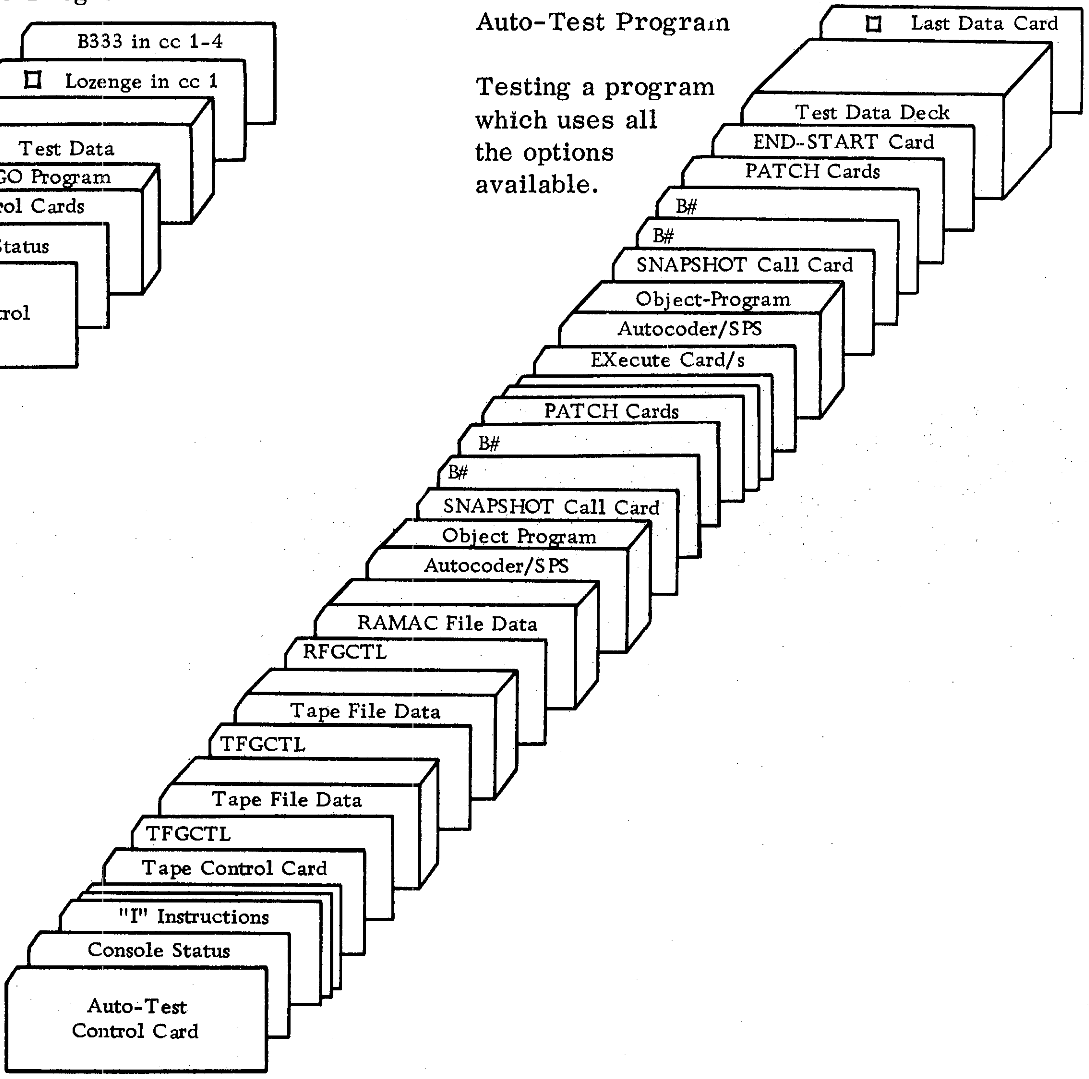


C. FARGO Program

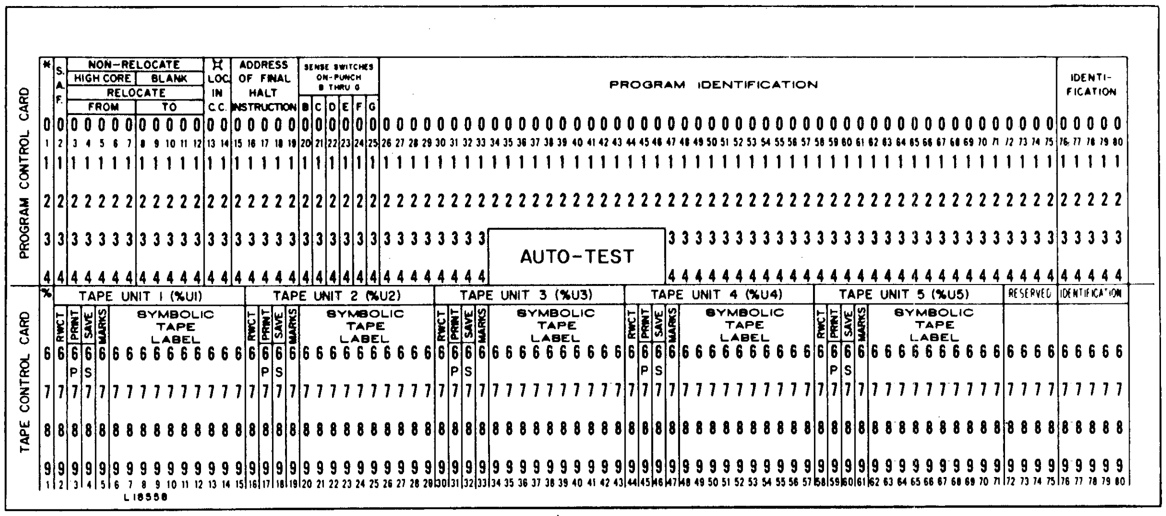


Auto-Test Program

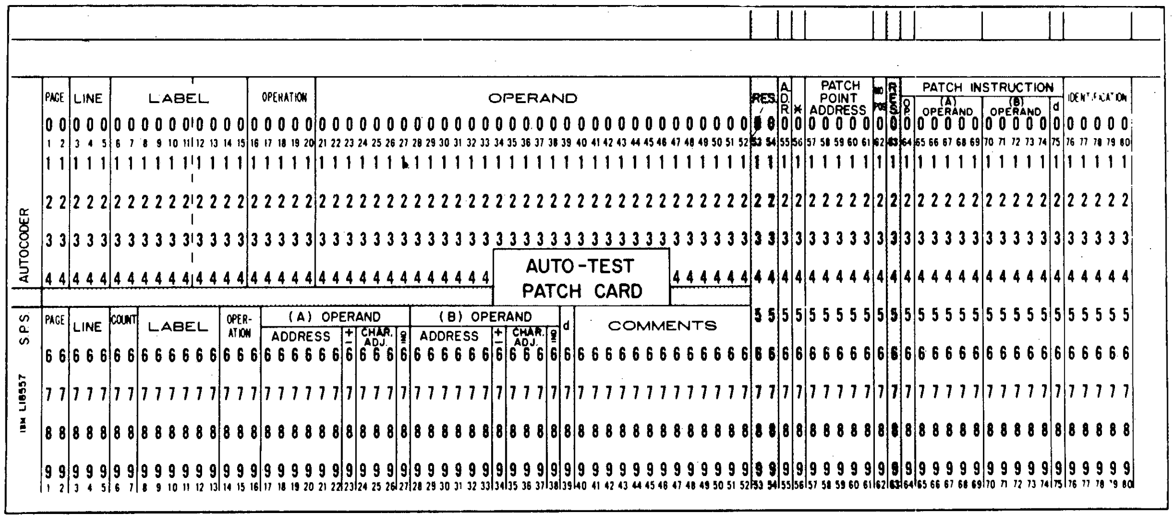
Testing a program which uses all the options available.



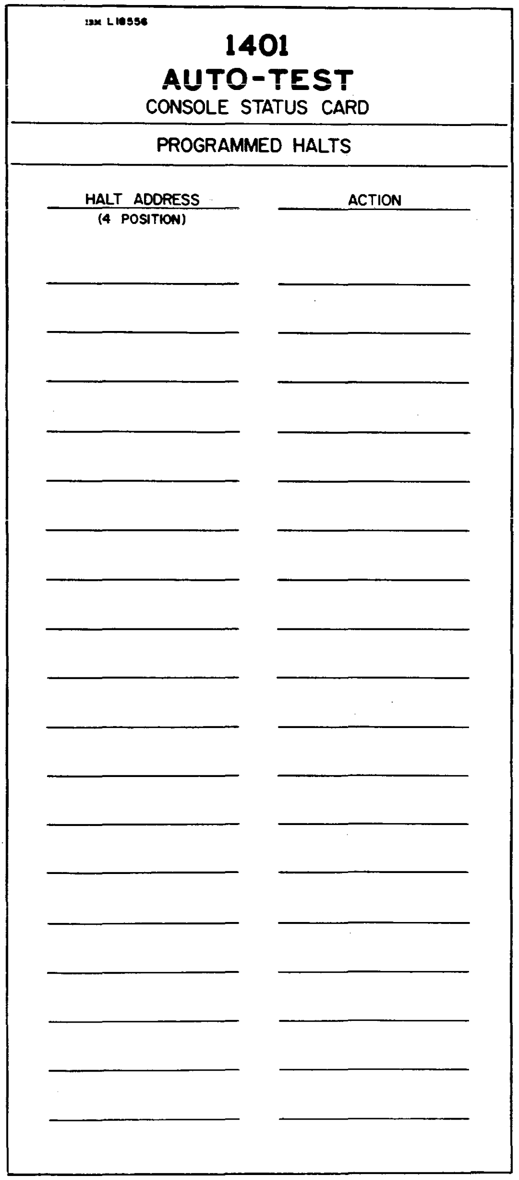
SAMPLE CARD FORMS AND ELECTRO NUMBERS


PROGRAM CONTROL CARD
S. A. F.
NON-RELOCATE
HIGH CORE
BLANK
RELOCATE
FROM
TO
LOC. IN C.C.
ADDRESS OF FINAL HALT INSTRUCTION
SENSE SWITCHES ON-PUNCH B THRU G
B C D E F G
PROGRAM IDENTIFICATION
IDENTIFICATION
AUTO-TEST
TAPE CONTROL CARD
TAPE UNIT 1 (%U1)
TAPE UNIT 2 (%U2)
TAPE UNIT 3 (%U3)
TAPE UNIT 4 (%U4)
TAPE UNIT 5 (%U5)
RESERVED
IDENTIFICATION
RWCT
PRINT
SAVE
MARKS
SYMBOLIC TAPE LABEL
L18558

Auto-Test Card


AUTOCODER
PAGE
LINE
LABEL
OPERATION
OPERAND
RES.
A D R
PATCH POINT ADDRESS
PATCH INSTRUCTION
(A) OPERAND
(B) OPERAND
d
IDENTIFICATION
AUTO-TEST PATCH CARD
SPS
COUNT
OPER-ATION
(A) OPERAND
(B) OPERAND
ADDRESS
CHAR. ADJ.
COMMENTS
IBM L18557

Patch Card



IBM L18556

1401

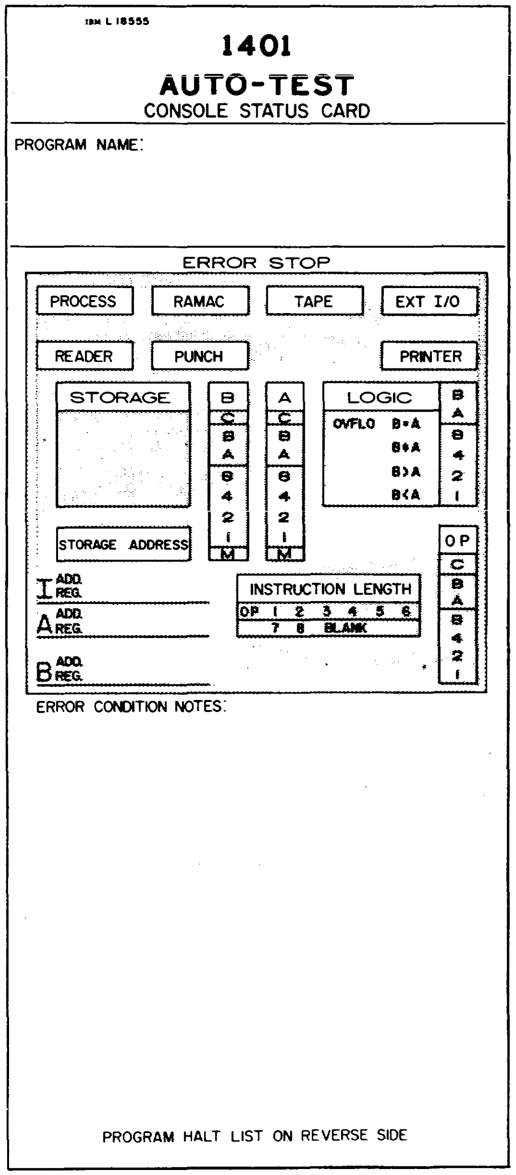
AUTO-TEST

CONSOLE STATUS CARD

PROGRAMMED HALTS

HALT ADDRESS (4 POSITION)	ACTION

Console Status Card (Back)



IBM L 18555

1401

AUTO-TEST

CONSOLE STATUS CARD

PROGRAM NAME:

ERROR STOP

PROCESS RAMAC TAPE EXT I/O

READER PUNCH PRINTER

STORAGE

B C B A 8 4 2 1 M

A C B A 8 4 2 1 M

LOGIC OVFLO B=A B≠A B>A B<A

B A 8 4 2 1

STORAGE ADDRESS

I ADD. REG.

A ADD. REG.

B ADD. REG.

INSTRUCTION LENGTH

OP 1 2 3 4 5 6

7 8 BLANK

OP C B A 8 4 2 1

ERROR CONDITION NOTES:

PROGRAM HALT LIST ON REVERSE SIDE

Console Status Card (Front)

F20-0234-2

IBM 1401 Printed in U.S.A. F20-0234-2

IBM
International Business Machines Corporation
Data Processing Division
112 East Post Road, White Plains, N. Y. 10601