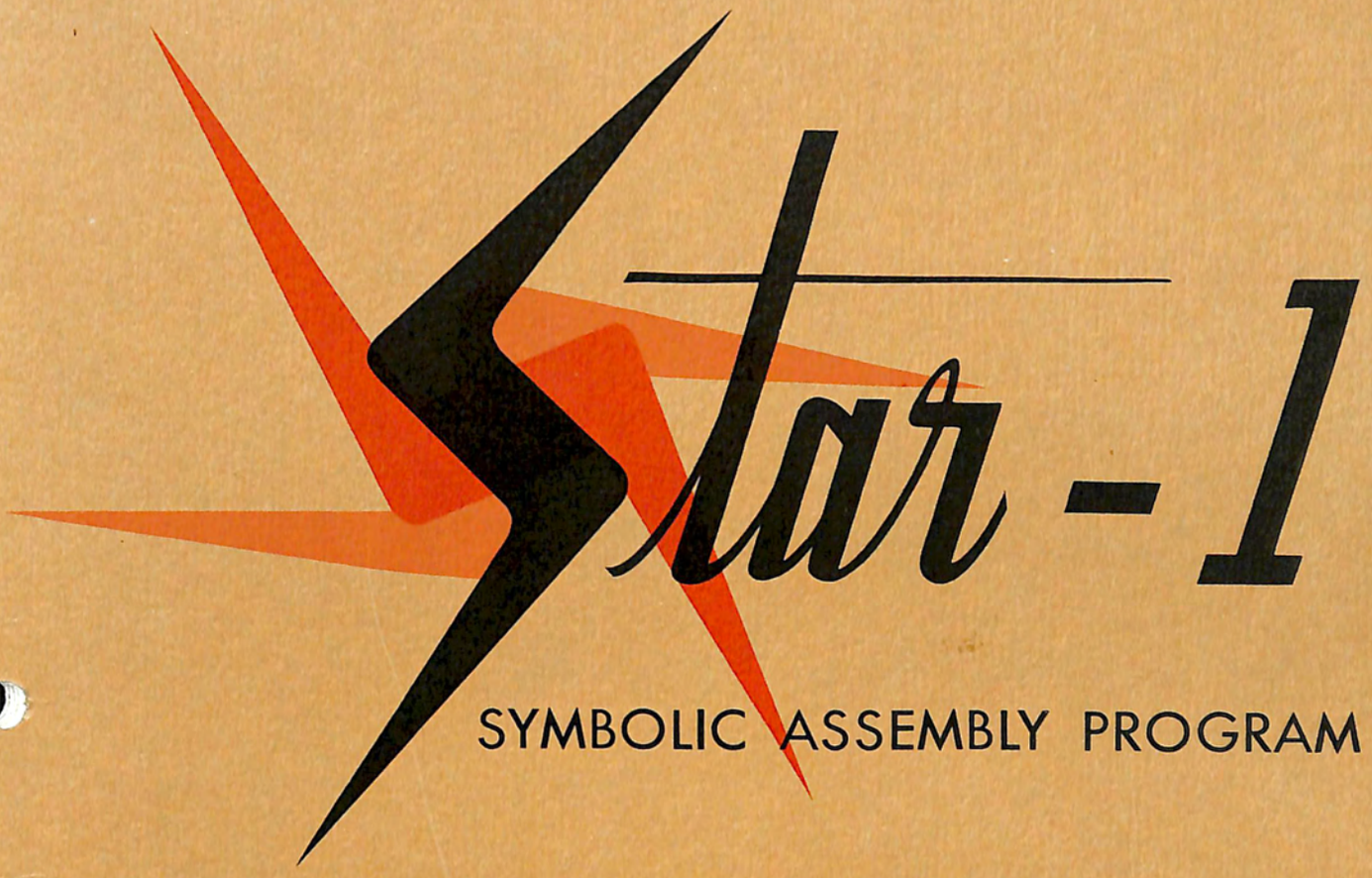
E. T. Pope



SYMBOLIC ASSEMBLY PROGRAM

BURROUGHS 220

ELECTRONIC DATA PROCESSING SYSTEMS

STAR 1

A SYMBOLIC ASSEMBLY

PROGRAM

FOR

THE BURROUGHS 220

ELECTRONIC DATA PROCESSING SYSTEM

Applied Programming Section
BURROUGHS CORPORATION
ElectroData Division

STAR 1

...an automatic coding system for the Burroughs 220

This handbook describes STAR 1, an assembly program for the Burroughs 220. It explains the symbolic notation used, the preparation and input of the program, and the two phases of the assembly process. Samples of input and output are included to show how a program is prepared and what the results are.

Although automatic coding devices—such as STAR 1—are considered essential by experienced programmers, their usefulness may not yet be apparent to all of the people connected with a computer installation and the organization it serves. This discussion of STAR 1, therefore, begins with a description of several types of automatic programming aids now in use. It then explains, in more detail, the nature and purpose of assembly programs and the characteristics of this particular example—STAR 1.

For those who will make direct use of STAR 1, a technical supplement with more detailed operating instructions is available.

TABLE OF CONTENTS

SECTION I

INTRODUCTION

AUTOMATIC CODING

Automatic coding is a convenient phrase used to describe a number of special programs that are used as aids to arrive at final programs. They all have one thing in common: they make use of the computer to do part of the programmer's work for him.

Without such machine assistance, a programmer must prepare a complete and detailed list of instructions, ready for direct acceptance by the computer.

With these programming aids, the computer itself takes care of many details, such as translating easily remembered letter codes to digits, choosing the most suitable areas of storage, and supplying required subroutines.

Not all of these automatic coding aids do all of these things. But some of them do all these and more. According to their major characteristics, the several types have been assigned names and, although there is much overlapping, each type can be fairly well described.

Assemblers

Assembly routines produce a machine-language program from "pseudo" instructions. Pseudo instructions can be defined as instructions that cannot be used directly by the computer but, instead, must be translated into "real" instructions by a special routine. Each pseudo instruction will result in one real instruction.

One way the assembler simplifies the programmer's job is by providing a means for convenient insertion or removal of program steps. Assemblers make use of relative addressing and symbolic coding—short cuts that will be described more fully in the typical assembly routine discussed later.

Interpreters

An interpretive routine differs from other automatic coding schemes in that it translates a pseudo instruction, executes the resulting actual instructions, and then processes the next pseudo instruction. Noninterpretive methods produce a record of the final program to be run at another time. An interpreter thus executes many real instructions for each pseudo instruction.

Compilers

The compiler is one of the most ambitious of these automatic coding methods. It comes the closest to relieving the programmer of the routine details of his job, allowing him to write in a notation that more closely resembles the language used to state the problem. It produces many real instructions from a single pseudo instruction.

A compiler assigns storage areas and specific locations and supplies commonly used subroutines, complete with means of entry and exit.

ADVANTAGES OF AUTOMATIC CODING

So far, we have considered the value of these specialized routines to the programmers who make direct use of the computer system. But there are further advantages to be gained by organizations using automatic coding: savings in time and money.

These savings result from:

Short cuts in preparing programs for an application
Use of less-experienced personnel
Ease of program revision
Less training of personnel required
Less machine time spent in finding program errors

It is these advantages that justify the time spent in planning and preparing automatic coding systems.

A TYPICAL ASSEMBLY ROUTINE

Most assembly routines—including STAR 1—have certain characteristics in common. Let's consider some of them in more detail.

Translating Alphabetic Codes

Nearly all coding manuals use alphabetic codes to refer to computer operations because they are easy to remember, while the numeric codes recognized by the computers are not. Usually a novice programmer uses the alphabetic codes when writing a program, then goes back and writes in the corresponding numeric operation codes by looking them up in a table. The assembly routine does this chore for him, saving him considerable trouble and removing one source of frequent error.

Regions

Assembly routines permit a problem to be broken into logical parts, so that each part may be coded as a unit. Thus several programmers can work on the same problem, each working on a region independently; the computer takes care of combining the regions into a final program.

Regions serve still another purpose. They provide a means for the use of relative addressing, a method of avoiding decisions about the choice of actual storage

locations. The locations written by the programmer for each region are relative to the first location of that region.

Symbolic Notation

One of the most annoying problems the programmer has to deal with arises when he discovers he needs to insert or delete an instruction. This forces him to renumber other instructions to retain the proper sequence. And, since many instructions have address portions referring to other instructions, these addresses must also be altered.

Symbolic notation solves these problems. Instead of writing actual locations, the programmer uses a symbolic notation, understood by the assembly routine, to represent locations. The symbols are simply numbers arranged to represent regions and relative addresses. These different parts of a symbolic address, when printed, are often separated by decimal points for clarity. To insert an additional instruction, the programmer can add another digit. Between the numbers 10 and 11, for example, he has the possibility of inserting 10.1, 10.2, and so forth. Again, the computer—as directed by the assembly program—will assign the proper locations.

STAR 1

STAR 1 includes the characteristics described above and, in addition, offers other conveniences to the programmer.

There are two phases of assembly operation. In the first phase, the contents of the cards that have been punched from the entries on STAR 1 coding forms are read into storage. There, a table is built up that records symbolic locations with their corresponding actual locations.

During the second phase, alphabetic operation codes are translated. The actual address associated with the symbolic address part of each instruction is found in the table built up in Phase I. Any coding errors which STAR 1 discovers in the original symbolic input are identified.

Both printed and punched-card output is produced at this time, which includes a complete record of the original symbolic input as well as the final assembled program.

STAR 1 may be used with equal facility in both scientific and business programming.

SECTION II

STAR 1 ASSEMBLY PROGRAM

BRIEF DESCRIPTION

The coding form on page 18 is designed for use with STAR 1. The location of instructions, data, or constants, as well as the address part of instructions, may be coded symbolically or in actual machine code. Operation codes may be designated alphabetically, or in actual machine code. Special entries may be flagged separately, and insertions can be made without changing other instructions. The STAR 1 program is stored in 1600 locations.

All information on one line of the coding form, including remarks, is first punched on one card or on paper tape. This information will be assembled to produce one instruction, data word, or constant.

The information on the cards or paper tape is then entered into the core storage of the Burroughs 220 at 240 cards per minute or 1000 paper tape characters per second. [Hereafter references will be made to punched card input-output only. Paper tape may be used with equal facility and is treated briefly in the section covering Program Control Switches.] The translation of this symbolic program is accomplished in two phases:

PHASE I As the cards are read, a cross reference table is generated internally, recording all symbolic locations with their corresponding actual locations. An image of the symbolic cards is recorded on magnetic tape.

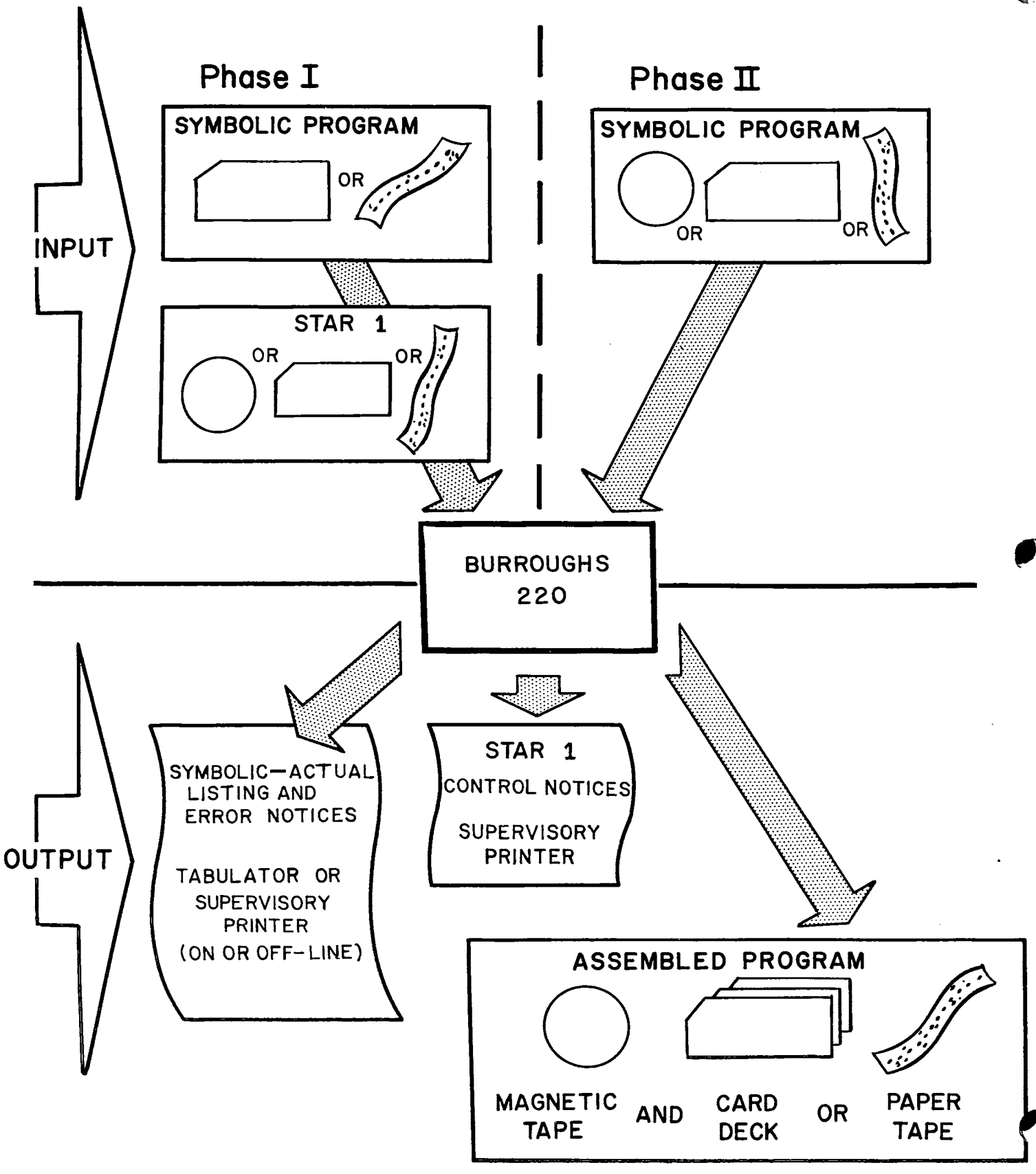
PHASE II The symbolic information is then read from magnetic tape (or the original cards, if tape is not used) into storage, one entry at a time. The alphabetic operation codes are translated by table look-up into the proper machine code. Actual instruction addresses are then assigned, using the table generated in Phase I. An "assembled" card deck, called the program deck, is then punched. A listing is made simultaneously showing all the original card information, actual machine instructions, and the actual locations assigned to instructions.

Certain errors or possible errors are noted on the Supervisory Printer or the Type 407 during Phases I and II. Among many are included:

- —Input out of sequence
- —Improper operation code
- —A symbolic instruction address without a corresponding symbolic location
- —Storage overflow
- —Improper field selection designation

STAR 1 ASSEMBLY

GENERAL FLOW CHART



The following sample illustrates the utility of symbolic coding in permitting insertions to be made in the middle of a group of instructions. This group of seven instructions appears near the end of region 123. Their purpose is to sum a block of 100 input data words in actual storage locations 8000-8099 and to compare that sum with a check sum word from location 8100. If the check sum is verified, the program proceeds to another section, called region 050; if not, there is a program error stop.

UNPUNCHED REMARKS	CLASS	LOCATION REGION	LOCATION SEQUENCE	LOCATION IN	INSTRUCTION S	CONTROL	OP	ADDRESS REGION	ADDRESS SEQUENCE	ADDRESS IN	increment	CONSTANTS/REGISTERS/REMARKS
		123	30			4400	DLB	123	36			0098
			31				CAD		8000		FIRST ITEM	
			32		1		ADD		8001		LAST ITEM...SECOND ITEM	
			33			0001	DBB	123	32			
			34				SUB		8100		SUBTRACT CHECK SUM	
			35				BFA	50	0			
			36			0098	HLT		9999		PROGRAM ERROR STOP	

Long after this bit of coding—and much more—has been written, it is all reviewed. The programmer discovers that, in summing the contents of locations 8000-8099, an unwanted overflow stop will probably occur unless the instruction in 123.0032.0 is followed immediately by a BOF command. He simply appends—onto any one of the coding sheets, if he chooses—the entry:

	123	321					BOF	123	33			

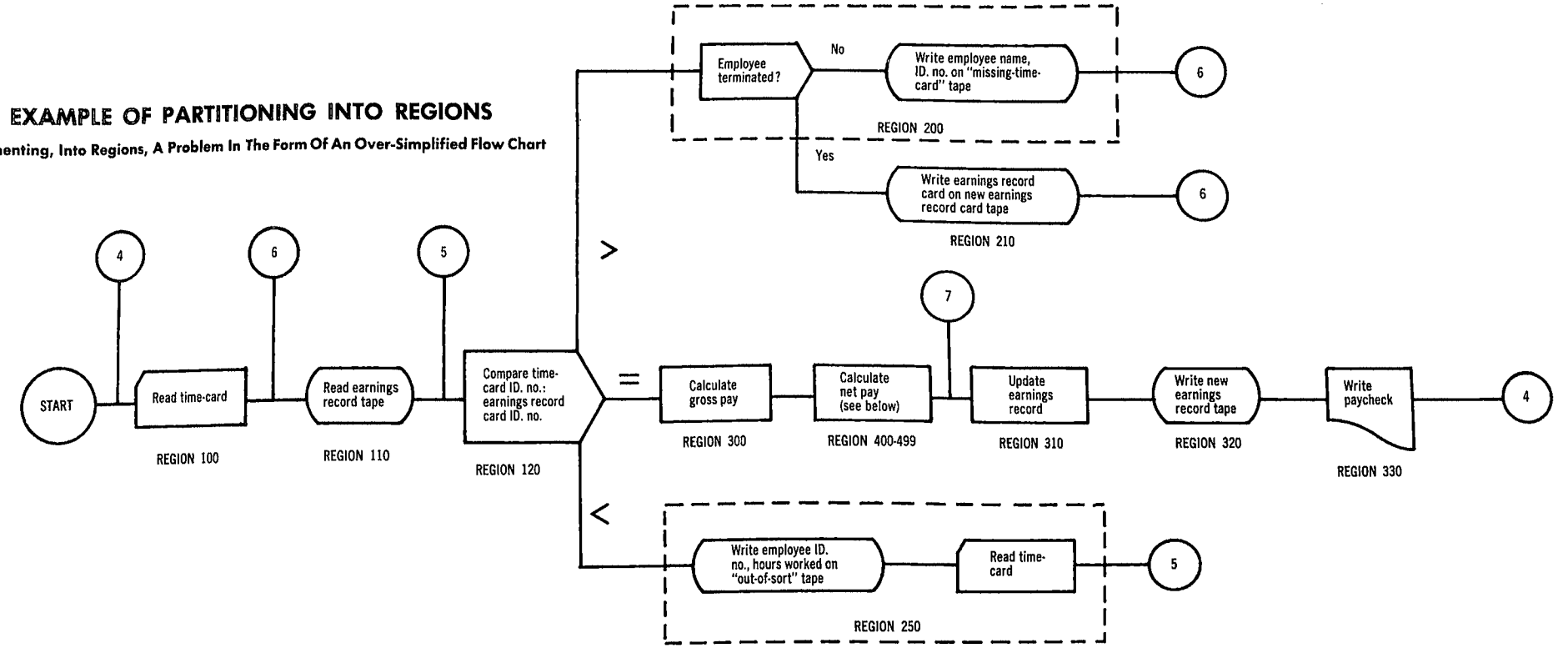
After further inspection, he remembers that, if the check sum is not verified, the program does not necessarily have to come to an error halt. A rare exception exists when digit K of the code word in 8101 is a 3, in which case the program should branch to region 051. He adds the inserts:

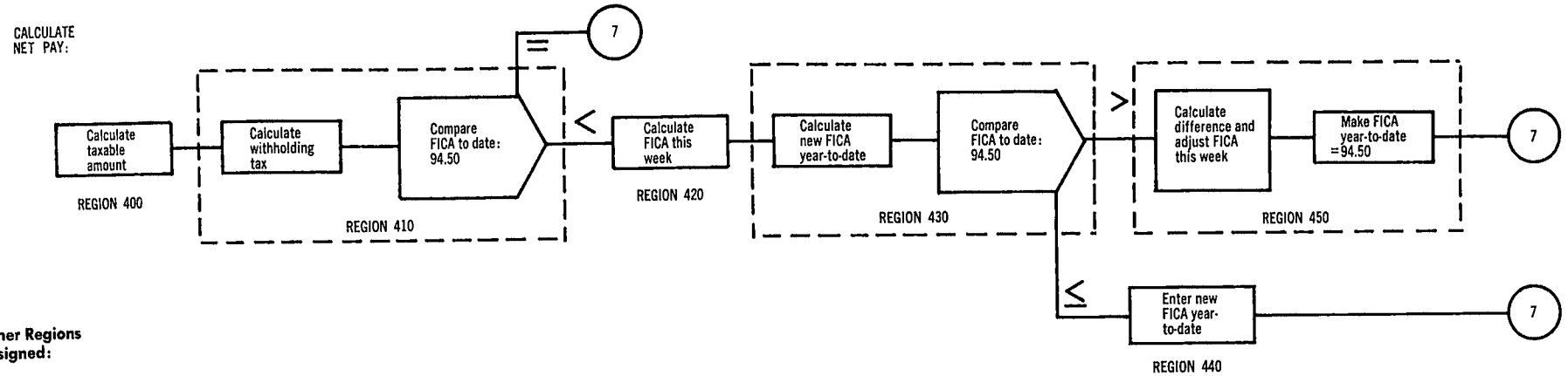
	123	355					LDR		8101			+OKNNNDDDDD
	123	356		2103			BFR	51	0		BYPASS STOP IF K EQ 3	

EXAMPLE OF SYMBOLIC CODING

EXAMPLE OF PARTITIONING INTO REGIONS

Segmenting, Into Regions, A Problem In The Form Of An Over-Simplified Flow Chart





Other Regions Assigned:

010 Control
050 Preliminaries
060 Cleanup

001 Temporary Working Storage
002 Permanent Working Storage

800 Input Storage Area: Earning Record from Tape Unit 1.
810 Input Storage Area: Time Card from Card Reader 1.

850 Format Band for Cardatron Input Unit 1 (Reading Time Cards)
860 Format Band for Cardatron Output Unit 1 (Printing Paychecks)

900 Output Storage Area: New Earnings Record for Tape Unit 2.
910 Output Storage Area: "Missing Time Card" Entry for Tape Unit 3.
920 Output Storage Area: "Out of Sort" Entry for Tape Unit 4.
930 Output Storage Area: Paycheck for Output Unit 1.

SECTION III

STAR 1 PROGRAMMING

REGIONAL-SYMBOLIC NOTATION

STAR 1 accepts symbolic storage references in the form of an eight-digit number, RRR SSSS I, as shown in columns 12-19 and 28-35 on the coding form:

RRR: these three digits represent a *region,* or logical section, of a program. Region numbers may vary between 000-999.

SSSS: the digits SSSS represent the relative *(sequence)* order of instructions, data, or constants within a region. Sequence numbers may vary between 0000-9999 within any one region.

I: this digit represents the relative order of any instructions, data, or constants inserted between two adjacent entries. *Insert* digits may vary between 0-9.

REGIONS, TYPE A AND TYPE B

Type A regions are generally used for data, temporary storage, permanent storage, subroutines, and input-output areas. These are numbered 000-009 and 800-999. Sequence numbers within any of these regions must ascend and be consecutive. Insert digits are ignored. Note that Type A regions are not used with symbolic instructions. The actual location of an entry in any Type A region is determined by adding the sequence number to the origin (actual location of the first word) of the region. Any number of Type A entries (limited by total storage size) may be assembled.

Type B regions are generally reserved for symbolic instructions and instruction constants. The *location* sequence numbers and insert digits of Type B regions must ascend, but need not be consecutive. Type B region numbers range between 010 and 799 inclusive. The actual location of an entry is determined by adding the relative location within the region to the origin of that region. The maximum number of Type B entries allowable in a problem is limited by the total storage size less 1600 locations for STAR 1 program and working storage.

Example:

TYPE A REGION

Programmer—Written Symbolic Location	Machine—Assigned Actual Address
802.0000.0	2526 (origin)
1.0	2527
2.0	2528
. . . . .	. . . .
47.0	2573 = 2526 + 0047
. . . . .	. . . .
56.0	2582 = 2526 + 0056
. . . . .	. . . .
802.0060.0	2586 = 2526 + 0060

TYPE B REGION

Programmer—Written Symbolic Location	Machine—Assigned Actual Address
123.0000.0	1198 (origin)
1.0	1199
2.0	1200
2.1	1201
2.2	1202
2.5	1203
3.0	1204
20.0	1205
21.0	1206
. . . . .	. . . .
117.0	1237
118.0	1238
119.0	1239
120.0	1240
120.1	1241
120.2	1242
120.3	1243
120.4	1244
121.0	1245
. . . . .	. . . .
123.0500.0	1401

Flexibility of Region Assignments

All symbolic instructions and program constants should be considered Type B regions (010-799). Data, constants, temporary storage, etc., *may* be assigned a Type B region, if desired, but for simplicity of program check-out and installation control, the practice should be avoided.

Conversely, symbolic instructions *may* be assigned a Type A region. This use of Type A regions would amount to machine language coding, since insert digits are ignored and sequence numbers must be consecutive. The use of Type A regions in this way should be avoided.

LOCATION COUNTER

Actual locations for words assembled using STAR 1 are assigned by a location counter (a tally kept within the program itself). The origin of any region is normally 1 greater than the last location assigned to the previous region.

For Type A regions, the location counter is controlled by adding the sequence number to the origin of the region.

For Type B regions, once the origin has been determined, the location counter is advanced by 1 for each instruction or program constant encountered.

SPECIAL REGIONS 000, 001, 002, 003

Region 000 is used to indicate instruction addresses which are of the non-operand type and normally to provide ten unassigned locations (0000-0009). When the regional part of the *address* in a symbolic instruction is 000, non-operand unmbers (SHIFT RIGHT A and R *14* —SRT 000.0014.0) are indicated.

Since the origin of Region 000 is always 0000, the actual machine location assigned to a region 000 entry is identical to the location sequence number. Effectively, this would result in machine coding. Region 000 location numbers should be used with caution. Some uses are magnetic tape end-of-file control, modification of starting and rerun procedures, etc. Unless otherwise specified, region 000 locations are 0000-0009.

Region 001 is used for temporary storage. Only the *address* region of a symbolic instruction should refer to Region 001. Unless otherwise specified, Region 001 locations are 0010-0059.

Region 002 is used for permanent storage (intermediate results, processed data, etc.). Only the *address* region of a symbolic instruction should refer to Region 002. Unless otherwise specified, Region 002 locations are 0060-0099.

Region 003 is a special notation for symbolic instruction addresses which refer to other instructions or program constants within the same region.

	LOCATION	INSTRUCTION
Instead of:	327.0421.3	BUN 327.0391.0
write:	327.0421.3	BUN 3.0391.0

This technique reduces writing errors, simplifies the symbolic listing, and eliminates subroutine address region alteration.

TYPES OF ENTRIES: CLASSES 0-9

There are three general categories of entries, designated by class codes.

—Symbolic instructions—Class 0 and 3.
—Numeric and alphanumeric constants—Class 1 and 2.
—Control cards—Class 5, 6, 7, and 9.

Class Code — **Type of Entry**

0 or blank A symbolic instruction, punched in columns 20-35.

1 A numeric constant, punched in columns 36-46. Columns 20-35 are blank.

2 An alphanumeric constant, punched in columns 37-41. Columns 20-36 are blank. A 2 is automatically inserted in the sign position of the corresponding assembled word by STAR 1.

3 A symbolic instruction, punched in columns 20-35. The number in 36-40 (± in 36) is added to the actual address after it has been looked up. This is used to refer to words whose symbolic location is unknown. The reference is made in terms of the symbolic address and the constant increment.

4 Not assigned.

5 Region Origin Control Card. The number in columns 31-34 replaces the previous location counter setting. Columns 20-30, 35 are blank. This card is used to arbitrarily set region origins. This card, if used, is the first symbolic entry of any region.

6 Region Increment Control Card. The number in columns 31-34 is added to the location counter setting. Columns 20-30, 35 are blank. The card is used to create open areas within or at the end of any region.

7 Location Lookup Control Card. The actual address already assigned to the symbolic location punched in columns 28-35 replaces the previous location counter setting. Columns 20-27 are blank.

8 Not assigned.

9 Remarks Control Card. This card is used to print out extra remarks, headings, etc., from columns 36-65. A Class 9 card must conclude each deck with 999.9999.9 as the symbolic location and END as the operation code.

CODING FORM USE: COLUMNS 01-65

Column 01

Format band selection is a one-digit numeric field. Column 1 must contain a 1 to select format band 1 during card reading.

Columns 02-10

General identification is an optional alphanumeric field used for card identification.

Column 11

Class is a one-digit numeric field specifying the type of entry being made.

Columns 12-19

Symbolic location is an eight-digit numeric field.

For convenience, write:			instead of:		
REGION 12-14	SEQUENCE 15-18	INSERT 19	REGION 12-14	SEQUENCE 15-18	INSERT 19
27	0		027	0000	0
	1		027	0001	0
	2		027	0002	0
	2	5	027	0002	5
	30		027	0030	0
	35		027	0035	0
	35	1	027	0035	1
	500		027	0500	0
	1200		027	1200	0

Column 20

Sign is a one-digit numeric field. The sign of an instruction is any digit 0-9. An 11 or 12 punch is not used. 0 is plus and 1 is minus. Sign control is not exercised during Phase I or II.

Columns 21-24

The *control digits* are a four-digit numeric field. Column 24 may be blank if alphabetic abbreviations are used in place of numeric operation codes. Various tests are made on the control digits.

Columns 25-27

The *operation code* is a three-alphanumeric-character field. Three letters are used in columns 25-27 or the actual two-digit operation code in columns 25-26, with column 27 blank. When the two-digit operation code is used, any variation designator must be noted in column 24; when the alphabetic abbreviation is used, the assembly program will supply the correct variation designator in column 24.

Columns 28-35

Symbolic address is an eight-digit numeric field used with class 0 or class 3 entries only.

Columns 36-65

Constants/Registers/Remarks is a thirty-alphanumeric-character field. The programmer may use these columns to write up to thirty alphanumeric remark characters, referring to the entry on that card. With three exceptions, nothing appearing in columns 36-65 will influence the assembly process. These are:

1. Class 1 entry: numeric constant entered in columns 36-46.
2. Class 2 entry: alphanumeric constant entered in columns 37-41.
3. Class 3 entry: instruction address increment entered in columns 36-40.

The remainder of the field may be filled with any remarks. Headings are provided at the top of the coding form to keep track of the contents of the A, R, and B registers.

Because there are no alphanumeric codes for certain characters, the programmer must use substitutions for some commonly used characters, e.g.,

—— for ( )
EQ for =
PLUS for +

Columns 62-64

Columns 62-64 are used to keep track of the decimal points of numbers shifted, multiplied, etc. Recommended orientation for decimal location is with respect to the left end of the A register.

Column 65

A "reference" digit, correctly employed and kept up to date, may be used to provide an indication of the extent to which a change in this instruction might affect other entries—for the programmer contemplating the ramifications of an alteration—and as an aid in following the program. It is *never* meant to supplant the thorough use of a pre-assembly listing of all instruction cards, sequenced on symbolic *address*. (The assembly program listing is in sequence by symbolic *location*.) Following is one *abbreviated possibility* for a code:

CHARACTER	SIGNIFICANCE
Blank	No reference whatsoever to this entry.
1	The symbolic location of this instruction—and *only* that—is referred to by another instruction, that is, appears as the address of a branch instruction somewhere.

By inference, the symbolic location is *also* referred to simultaneously in the following:

4	Some field of this instruction is referred to somewhere, e.g., by a DLB instruction, a CFA instruction with sLf = 221, etc.
8	Some *partial* field of this instruction has been modified *in storage*, e.g., a prestored address.
9	The *whole* instruction has been modified *in storage*, e.g., prestored.

SECTION IV
STAR 1 OPERATION

PROGRAM CONTROL SWITCH SETTINGS FOR VARIATIONS

1 OFF Normal; Symbolic input is on cards using Cardatron Input Unit #1 (Type 089 Collator). Output consists of a printed listing using Cardatron Output Unit #1 (Type 407 Printer) and a punched program deck using Cardatron Output Unit #2 (Type 523 Punch).

ON Paper tape input/output. Symbolic input is presented on punched paper tape using Photoreader #1. Output consists of two punched tapes prepared on Paper Tape Punch Units #1 and #2. The Unit #1 tape is used to prepare the off-line listing and the Unit #2 tape is the program tape.

2 OFF Normal. The listing is prepared using Cardatron or Paper Tape System as specified by Program Control Switch #1.

ON Bypass the listing of the assembled symbolic program.

3 OFF Normal. The assembled program is prepared using Cardatron or Paper Tape System as specified by Program Control Switch #1.

ON Bypass the punching of the assembled program deck.

4 OFF Normal. Check the sequence of the symbolic input. Identical symbolic locations are treated as a sequence error unless the first entry is a control card.

ON Same as above, except regions need not be in sequence.

5 OFF Normal. No significance.

ON Punch Table I (of the symbolic-absolute table) into paper tape using Paper Tape Punch Unit #1 at the end of Phase I. After Program Halt 5551, leave on to punch Table II also; otherwise, switch off. After Program Halts 5551 and 5552, space Paper Tape Punch Unit #1 before continuing.

6 OFF Normal. No significance.

ON Certain letters entered in column 65 (the "Reference" column) of the coding sheet serve as special control flags for the assembly program, as follows:

S: Space twice (double space, or one extra line) on the 407 before printing the line for this entry on the listing.

R: Restore (skip to 1) the page on the 407 before printing the line for this entry on the listing.

P: Punch the present (incomplete) program deck card and make this the first entry in a new card.

Q: Combines functions P and R above.

Z: Stop assembly program at Program Halt 6666 after reading this entry and transferring input to working storage. The phrase "Z STOP" will have been printed by the Supervisory Printer prior to the halt. To continue, press START switch on Control Console.

A thru I: Same function as S. A-I may be read as 1-9, plus the spacing function, if Program Control Switch #6 is on.

The use of Program Control Switch #6 in effect "limits" one to 29 characters of actual remarks, instead of 30.

7 OFF Normal. No significance.

ON Adjust signs of assembled instruction words to be consistent with B (the digit in column 20 of the END card), which is punched in column 4 of all program deck cards to control relocation.

B = 8: Any Class 0 or Class 3 entry with a sign digit of 8 or 9 is modified so that the assembled word has a sign of 0 or 1, respectively. This is to allow incorporation of subroutines or regions coded initially in "floatable" form into a "non-floatable" program.

B = 0: Any Class 0 or Class 3 entry with a sign digit of 0 or 1—whose symbolic address does not refer to Region 000, 001 or 002—is modified so that the assembled word has a sign of 8 or 9, respectively. This is to allow incorporation of subroutines or regions coded initially in "non-floatable" form into a "floatable" program. More important, this permits one to code any problem in regular fashion; conversion to floatable form may be accomplished during the assembly process.

8 OFF Normal. No significance.

ON The final assembled program is written on lane 0 of Magnetic Tape Unit #3, and this tape is rewound. If, following Program Halt 2222 at the termination of the assembly, the START switch on the Control Console is pressed, then

1. The program will be read into Data Processor storage from magnetic tape.
2. The phrase "READY TO RUN" will be printed by the Supervisory Printer.
3. Program Halt 8888 will occur.

Press START switch to begin program.

9 OFF Normal. No significance.

ON For re-assembly, symbolic entries (from a prior assembly) read from lane 1 of Magnetic Tape Unit #2 and symbolic entries (changes) read from cards or paper tape (as specified by Program Control Switch #1) are merged as input to Phase I. Appropriately coded "changes" may be insertions, alterations, or deletions. The time for Phase I is reduced considerably during re-assembly.

10 OFF Normal. Symbolic entries must be reloaded for input to Phase II (as specified by Program Control Switch #1); input to Phases I and II, and output from Phase II, must be either all via the Cardatron System or all via the Paper Tape System.

ON Symbolic entries read during Phase I are stored on lane 1 of Magnetic Tape Unit #1 and are re-read from that lane as input to Phase II. Each entry occupies one 10-word block, which includes a check sum and (except for control

cards) the corresponding assigned absolute location. The MAGNETIC TAPE OVERWRITE operation is used, presupposing that 10-word blocks have previously been laid down on lane 1. All rewinds are under program control. If the phrase "PREFACE ERROR" or "CHECK SUM ERR" is printed by the Supervisory Printer followed, respectively, by Program Halt 8881 or 8882, press the START switch on the Control Console to back up one block and attempt again to read.

Since input to Phase II is via the Magnetic Tape System, input to Phase I and output from Phase II may be in different media, selected by setting Program Control Switch #1 prior to Phase I and altering the setting during the Program Halt between Phases I and II.

PROGRAM HALTS

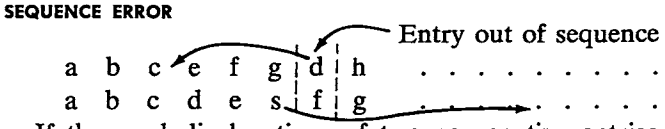
CONTROL REGISTER (rC: 44)	SIGNIFICANCE
1212	*End of Phase I.* Phrase "END PASS ONE" will have been printed by the Supervisory Printer. Reload the symbolic input for Phase II and press the START switch on the Control Console.
2222	*End of Phase II.* Phrase "END PASS TWO" will have been printed by the Supervisory Printer.
7711	*Class 4 or Class 8 entry.* Phrase "CL 4 OR 8" will have been printed by the Supervisory Printer. Class 4 or Class 8 entries have no significance for STAR 1. To skip a Class 4 or 8 entry and process the next entry, press START switch. If only the class digit is incorrect, assign the correct value in rR:21 and manually transfer to the location specified in Address register.
7722	*Storage exhausted.* Phrase "END OF TABLE SPACE" will have been printed by the Supervisory Printer. To transfer to the end of Phase I, press the START switch.
7733	*Input error.* Phrase "INPUT ERR" will have been printed by the Supervisory Printer. Improper use of column 1 is made. If (rA:11) $\neq$ 1, column 1 of the card read is improperly punched with a digit other than 1. This card may be ignored and the next card may be fed by pressing the START switch on the Control Console.
9911	Assembly or system error.
9922	Assembly or system error.
9933	Assembly or system error.
9944	Assembly or system error.
9955	Assembly or system error.
9966	Assembly or system error.

SUPERVISORY PRINTER NOTICES

With the exception of END PASS ONE and END PASS TWO, all of the following are error notices and are preceded by the symbolic location of the entry to which the error refers. The assembly process continues for all such notices except those associated with the previously noted Program Halts, determined during Phase I. Following a review of the coding sheets, a decision should be made to:

1. Stop the assembly process, correcting the error in the symbolic card deck away from the system, or
2. Proceed with the assembly process, because:
   a. Useful information regarding other probable errors may be obtained, especially from the listing.
   b. The error can be rectified later by appending a few absolute correction instructions at the end of the assembled program deck; the faulty symbolic entry must be corrected prior to future reassemblies.

SEQUENCE ERROR



If the symbolic locations of two consecutive entries are identical, the condition is treated as a sequence error unless the first entry is a control card.

INSERT DIGIT NOT EQ ZERO
Refers to a Type A region entry; the insert digit is ignored during the assembly process.

CL 567—A REG.
An uncommon error which occurs when a Class 5, 6, or 7 control card is used improperly in the middle of a Type A region.

SEQ NOS. JUMP
The difference in sequence numbers of adjacent entries in a Type A region is greater than ~~0100.~~ 0001

LOC PAST UPPER LIM.
The actual location assigned to this entry or about to be assigned to the next entry is beyond the upper storage limit. The "upper limit" is usually set to the highest core storage location available, such as 4999, 9999, etc.

CLASS 5 ORIGIN LOW
The location counter has been reset to a value less than the previous setting.

PAST 999 INSTR. IN B REG.
More than 999 entries have been made in a Type B region.

CLASS 7 LOOKUP ERR
There is no symbolic location in Table 1 or 2 corresponding to the symbolic address. The location counter is unchanged.

CL 4 OR 8
See program halt 7711.

END OF TABLE SPACE
See program halt 7722.

INPUT ERR
See program halt 7733.

END PASS ONE
See program halt 1212.

***END PASS TWO**
See program halt 2222. } **Phase II Notices*

***PAST THREE ERROR FLAGS**
Three possible errors have already been indicated on the listing.

After the phrase "END PASS TWO" has been printed on the Supervisory Printer, the following information regarding the program just assembled is also printed on the Supervisory Printer, in the following sequence:

1. Number of symbolic entries read in during Phase One.
2. Number of symbolic entries read in during Phase Two.
3. Number of lines printed out on listing.
4. Number of entries punched out (program deck or program tape).
5. Number of program deck cards punched out.
6. Number of control words punched out for paper tape program, if any.
7. Number of entries made in Table 1.
8. Number of entries made in Table 2.
9. Last Table 1 location filled.
10. Last Table 2 location filled.
11. Last entry made in Table 1.
12. Table 1 entry corresponding to last symbolic lookup.
13. Table 1 entry corresponding to last "new region" symbolic lookup.
14. Last new region origin.
15. Actual location of last processed output.

ERROR FLAGS ON LISTING

"Possible Error" Flags

Any of the following two-digit flags may appear on the listing, up to three per line. In the unusual event that a fourth or fifth "possible error" condition should be observed, the symbolic location of the entry will be printed on the Supervisory Printer with the phrase "PAST 3 ERROR FLAGS."

The error flags are broken into groups. "Possible errors" are tested during Phase II in group order of A, B, C, D, and then E.

Group A

Symbolic Location Lookup when no magnetic tape is available.

40 Location is beyond the storage limit.
43 Insert digit of Type A region location is not zero.

Group B

Symbolic Address Lookup for Class 0, Class 3 entries

30 No corresponding region.
31 No corresponding sequence number.
32 Address is beyond the storage limit (Type A regions, excluding Region 000.)
33 Insert digit of Type A region address is not zero.
34 Address after adding an increment value is beyond the storage limit.

Group C

The following possible errors are noted when the operation code of a Class 0 or Class 3 entry is assigned. Only one flag from this group may appear per entry. An error in Group C will obviate the need for testing any of the conditions in Group D.

20 *Incorrect variation digit in column 24.
21 *Incorrect alphabetic abbreviation: no operation code corresponds to this abbreviation.
22 *Incorrect numeric operation code (mixed numeric and alphanumeric).
23 *Incorrect numeric operation code (includes variation digit, if any applies).
24 Third operation code character (column 27) is zero (instead of a letter or blank).
25 Blank columns in operation code field with one or more non-zero digits in the remainder of the instruction word.

*00 (HLT) is inserted as operation code.

Group D

These apply to Class 0 or Class 3 entries. Various checks are made for possible errors in control, address, or sign digits of an instruction.

ERROR FLAG	NUMBER OF CASES	SAMPLE CASE	SIGNIFICANCE
00	6	PRD	No test necessary or possible. 00 not printed out.
01	27	CAD	iiix $\neq$ 000x.
02	16	HLT	iiii $\neq$ 0000
03	7	CRD	xiix $\neq$ x00x
04	4	MRD	xxix $\neq$ xx0x
05	4	BCS	xiii $\neq$ x000
06	1	RTF	ixxi $\neq$ 0xx0
07	2	MIR	xxii $\neq$ xx00
08	1	CWR	xixx $\neq$ x0xx
09	1	PWR	xxxi $\neq$ xxx0
10	5	STA	error in sLf
11	5	DFL	error in sLnn
12	2	MIW	kk = 02 – 09
13	2	MFS	sign $\neq$ 4 or 5; will change 0 or 1 to 4 or 5, respectively.
14	2	MTS	sign $\neq$ 0 or 8
15	2	SLT	iinn > 0019. If no error, test for 01 error flag.

ERROR FLAG	NUMBER OF CASES	SAMPLE CASE	SIGNIFICANCE
16	2	SLS	iinn > 0010. If no error, test for 01 error flag.
17	2	SLA	iinn > 0009. If no error, test for 01 error flag.
18	2	IBB	nnnn = 0000
	93		Corresponding to 93 separate operation codes.

Group E

Miscellaneous

51 Class 1 entry; non-numeric characters keypunched.
52 Class 2 entry; column 36 not blank.
53 Class 3 entry; sign of increment is not 0 or 1.
60 Sign of a Class 0 or 3 entry is a 2, 3, 4, or 5.
67 Sign of assembled word is 6 or 7.

PRE-ASSEMBLY PROCEDURES

Pre-assembly procedures are designed to help detect errors in programming and in preparation of the symbolic deck. These "off-line" techniques are very useful in the reduction of program checkout costs, and should be made a part of standard operating procedures.

1. Sight check column 1 of the symbolic cards for a 1 punch. Sight check columns 2-10 for standard identification, if any. Check that the proper END card is at the end of the symbolic deck.
2. Interpret the symbolic cards for easier reading.
3. Sort the cards on *symbolic address* (columns 28-35). List the cards using the standard tabulator pre-assembly board (see diagram on page 20). With ALTERATION SWITCH #1 on, the tabulator will space an extra line on a break in address. Sorting is simplified if zeros are punched for zeros (instead of leaving fields blank) in this field or any numeric field to be sorted.
4. Sort the cards alphabetically on operation code (column 25-27). List the cards using the standard tabulator pre-assembly board. With ALTERATION SWITCH #2 on, the tabulator will space an extra line on a break in operation code.
5. Sort the cards on symbolic location (columns 12-19). List the cards using the standard tabulator pre-assembly board. With all ALTERATION SWITCHES off, the tabulator will space an extra line on a break in symbolic location region.
6. Review the listings 3, 4, 5 (above) thoroughly, making any necessary corrections.
7. Sequence check the symbolic deck on symbolic location (columns 12-19) using the Type 089 Collator. Wire both LOW and EQUAL PRIMARY SEQUENCE hubs to the ERROR STOP hub. This insures that all errors from duplicate symbolic locations will be detected, even though anticipated stops will occur when a control card has been given the same symbolic location as the following entry. It is recommended that this sequence checking board be permanently wired and available at all times.

PROCEDURES FOR ASSEMBLY USING CARDATRON

1. Prepare the Supervisory Printer.
2. Verify that the proper standard Input Unit plugboard is in place.
3. Run out any leftover cards from the card reader.
4. Press CLEAR buttons on Cardatron Input and Output Units.
5. Place cards to be loaded in the primary hopper of the card reader.
6. Press the card reader START button for initial read-in. (One card should drop into stacker.)
7. Press CLEAR switch on Control Console.
8. Set PROGRAM CONTROL switches as desired.
9. Press the KEYBOARD switch and enter 1000 60 (CRD) 0000; press the "C" switch on the Keyboard and press the START switch; card reading will commence.
10. If necessary, stop the card reader to remove stacker overflow and/or add cards to the hopper. Press the card reader START button.
11. Verify that the proper standard Output Unit boards are in place. Check the paper and card supply in the tabulator and card punch. Press the START switches on the tabulator and the card punch (idling).

Phrase "END PASS ONE" will be printed on the Supervisory Printer. The Data Processor halts with 1212 in rC: 44.

12. *Run out last two cards (reject) from card reader.
13. *Store in a box all cards except the symbolic deck and the last two reject cards.
14. *Insert one card selecting Reject Format in front of these symbolic cards.
15. *Clear Input Unit and insert these cards in hopper of card reader.
16. *Press the card reader START button for initial read-in (one card in stacker).
17. *Press START switch on Control Console: Card reading will commence.

Phrase "END PASS TWO" will be printed on Supervisory Printer. Data Processor halts with 2222 in rC: 44.

Remember to:
Remove listings from tabulator and Supervisory Printer.
Remove all cards from card reader and card punch.

*Not required when magnetic tape is used.

SECTION V
MISCELLANEOUS

SUBROUTINES ON CARDS

Any program being written in STAR 1 notation can incorporate any other program coded in either STAR 1 or machine language notation. Such sub-programs, or subroutines, are usually in symbolic form on cards or in machine language form on cards, paper tape, or magnetic tape. Burroughs programs will be published in both symbolic and machine language to assist users in program preparation. Users may also make available to other users selected programs written in STAR 1 notation, thereby further reducing program preparation costs.

1. *Programs in Symbolic Form* (one instruction or constant per card)
   a. Location region is already assigned and punched in columns 12-14; insert the subroutine in the proper place in the symbolic deck being assembled, as any other region.
   b. Location region is not punched in columns 12-14. Assign and gang punch a chosen region number; then proceed as in (a). If the Region 003 notation has been used consistently with instruction addresses, no other modification is necessary.

   Any such symbolic subroutine must be assumed to be a Type B region. The sequence numbers of the first two entries must be 0000.0 and 0001.0 if basic linkage (STP S, BUN S+1) is used.
2. *Programs in Absolute Form* (five instructions or constants per card). Such routines should be coded relative to location 0000 and be capable of being relocated during input. Subroutines are placed sequentially *after* the program deck and are read in by the *assembled* program. This approach requires that:
   a. During assembly, storage areas must be reserved for the different subroutines. Regional identity (Type A) is established and space is allocated by using a Region Increment (Class 6) card. The increment is equal to or greater than the storage requirement of the associated subroutine.
   b. A subroutine loading package is provided separately (referred to as Region 011 hereafter) and must be included with the symbolic cards during assembly. A "calling sequence" (see example which follows) supplies Region 011 with the symbolic origin and identification number of each subroutine being assembled. The subroutines are loaded under the control of Region 011 by the same loading routine which read in the STAR 1 program deck. Locations 0000-0099 are used temporarily by this loading routine. Region 011 supplies the loading routine with the origin and identification number of each subroutine. If the six-digit identification field (columns 21-26) is blank, no comparison for equality with card columns 5-10 is made.
   c. The calling sequence (shown in the example as the first five entries in Region 010) to Region 011 is assembled with the other symbolic entries.

When referring to a subroutine in a library stored on paper tape or magnetic tape, instead of on cards, sections (a) and (c) above are unchanged. Section (b) (Region 011) must incorporate searching instructions.

PROGRAM DECK MODIFICATION AND RE-ASSEMBLY

The assembly process may detect certain programming errors and subsequent check-out will expose others. Corrections to the program deck are punched in the same form as is the program deck. It is best to append all correction cards at the end of the program deck without changing any of the program deck cards.

A program is re-assembled when finally debugged, or when an excessive number of appended corrections during debugging becomes confusing or bothersome. For re-assembly, remove deleted or changed instruction cards from the original symbolic deck, and add the proper symbolic insertions and changes and re-assemble.

A BRIEF LOOK INSIDE STAR 1

The table generated during Phase I, showing the correspondence between symbolic and actual locations, is composed of *two* separate tables, as follows:

Table 1: Contains a single one-word entry for each region assembled, in the following format:

F nnnRRRzzzz

F is a one-digit flag, 0 for a Type A region and 1 for a Type B region.

nnn is a running total kept of the number of entries processed, including control cards, and represents the corresponding number of Table 2 entries in each Type B region. Each Type B region is limited to 999 entries.

RRR is the region designated.

zzzz is a location. For a Type A region, zzzz specifies the origin assigned to the first word of this region. For a Type B region, zzzz specifies the location in Table 2 where the entries corresponding to this region *begin*. Table 1 builds upward in storage following the storage area used by the STAR 1 program.

Table 2: Contains a single one-word entry for each Type B region symbolic entry assembled, in the following format:

0 0LLLLSSSSI

SSSSI is the sequencing number of this entry within the region RRR.

LLLL is the actual location assigned. Table 2 builds downward in storage starting from the upper storage limit.

Operation code lookup is performed by STAR 1 during Phase II using a table with 93 entries corresponding to the different operations. Each word is of the form:

$F_1\ F_2\ V\ N_1\ N_2\ A_1\ A_2\ B_1\ B_2\ C_1\ C_2$

where: $A_1\ A_2\ B_1\ B_2\ C_1\ C_2$ is the bi-decimal form for the alphabetic abbreviation for the operation code;

$N_1\ N_2$ is the corresponding numeric operation code;

V is the appropriate variation digit, if any; and

$F_1\ F_2$ specifies the type(s) of "possible error" to be tested.

MISCELLANEOUS INFORMATION

A "PROGRAM CONTROL SWITCH code word" is printed on the Supervisory Printer at the beginning of both Phase I and Phase II. The code word is initially the ten-digit number 1234567891; in each of the digit positions corresponding to a switch which is on is printed the number of the switch; if the switch is off, a zero is printed.

During Phase II, assembled "actual" entries are punched, five per card, into program deck cards, in the format required by the Standard 220 Card Loading Routine. Whatever digits are entered in columns 20-24 of the END (last) symbolic card are punched into columns 4-8 (referred to as Bwxyz) of all the program deck cards. Columns 21-24 contain an identification number; column 20 contains either a 0 or an 8 to control relocation. Bwxyz is printed on the Supervisory Printer for purposes of identification, immediately prior to the end of Phase I; if B is not a 0 or an 8, the error notice "B NOT 0/8" is also printed. STAR 1 punches a control card for the end of the program deck, entering the cards-punched tally into columns 11-14. If columns 28-35 (symbolic address) of the END card are zeros, a HLT instruction is punched into columns 15-25 of the control card. If a symbolic address has been written, however, the corresponding actual location is ascertained and stored in the address part of a BUN instruction punched into columns 15-25 of the control card.

Without using control cards, it is impossible to overlap storage locations.

The symbolic locations of all control cards except Class 9 entries *are* entered in the table generated during Phase I; except for Class 6, the *new* location counter setting is used. The location counter is *not* automatically stepped up by one after processing any control card.

During the assembly process, *any* region whose *first* entry is a Class 5, 6, or 7 control card will be assumed to be a Type A region, unless a non-zero digit appears in column 24 of that control card.

If necessary, a Class 6 region increment card with increment of 0000 and the appropriate zero or non-zero digit in column 24 may be used to switch region types, if the card is placed at the beginning of a region.

Normally, the first region assembled will start in location 0100, with 0000-0099 being reserved for Regions 000, 001, and 002, regardless of whether these regions are used in the program. To assemble beginning at 0000, use a region origin card with an origin field of 0000 as the first entry of the first region.

The "remarks" field should be fully utilized by all programmers to provide a clear, adequate explanation of the program. These notations should be copious, kept up-to-date with respect to program modifications, and should cover all "tricky" programming. They are intended not only to aid the programmer himself but also to enable others to follow the logic at a later date.

Keypunchers become quite proficient at keypunching script from the remarks field, provided it is neatly written and spaced approximately one character per column.

All output is completely edited internally; only the Standard 120-120 407 Board and the Standard 80-80 523 Board need be used.

Supplements to STAR 1 will be made available if desired. These may include such items as:

1) Provision for assembling both symbolic cards and standard program deck cards together.

2) Provision for punching the information which is normally printed, so that numerous copies of a program may be obtained off-line.

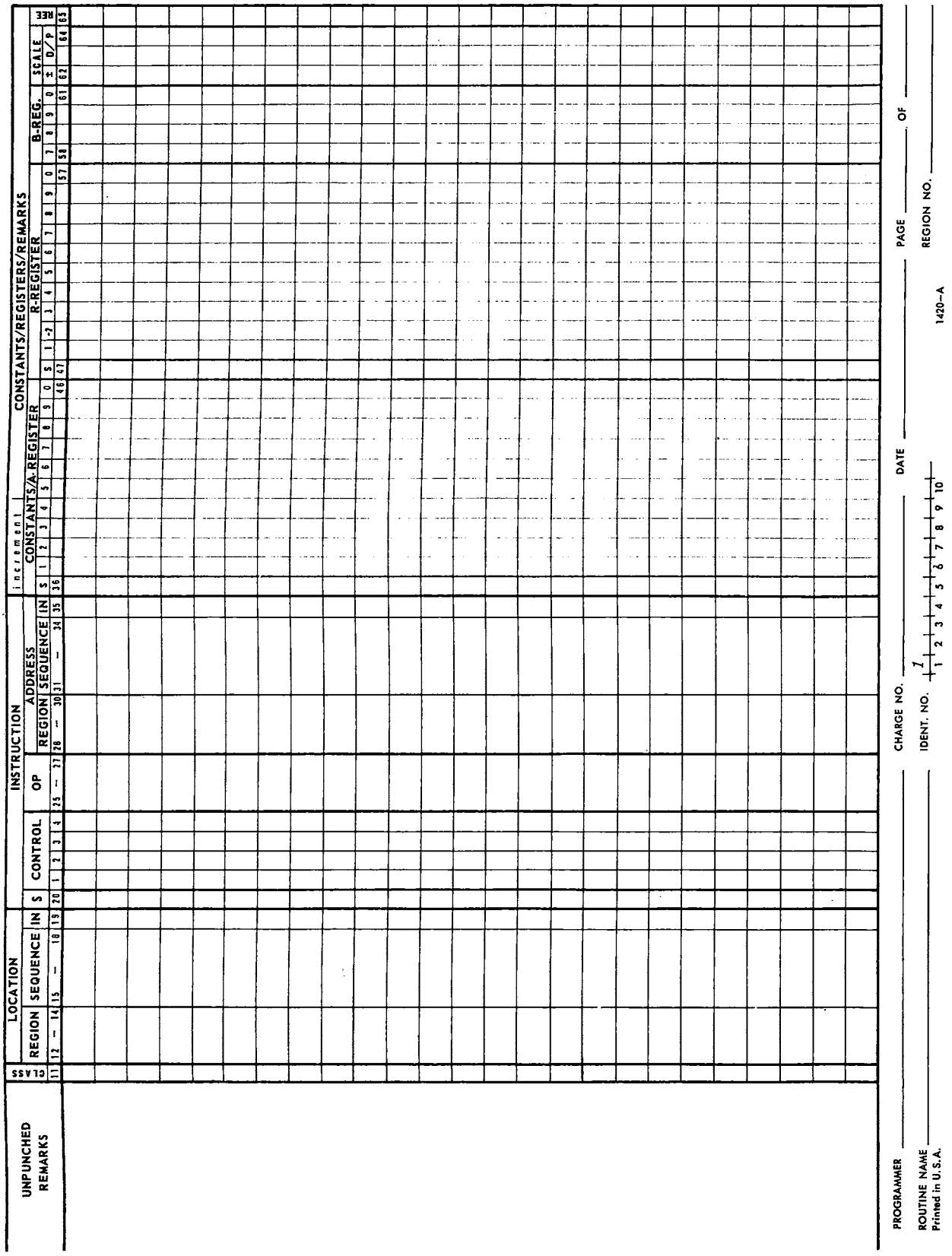
3) Provision for advancing the STAR 1 location counter to the next location ending in the specified digits x, xy, or xyz (generally zeros).

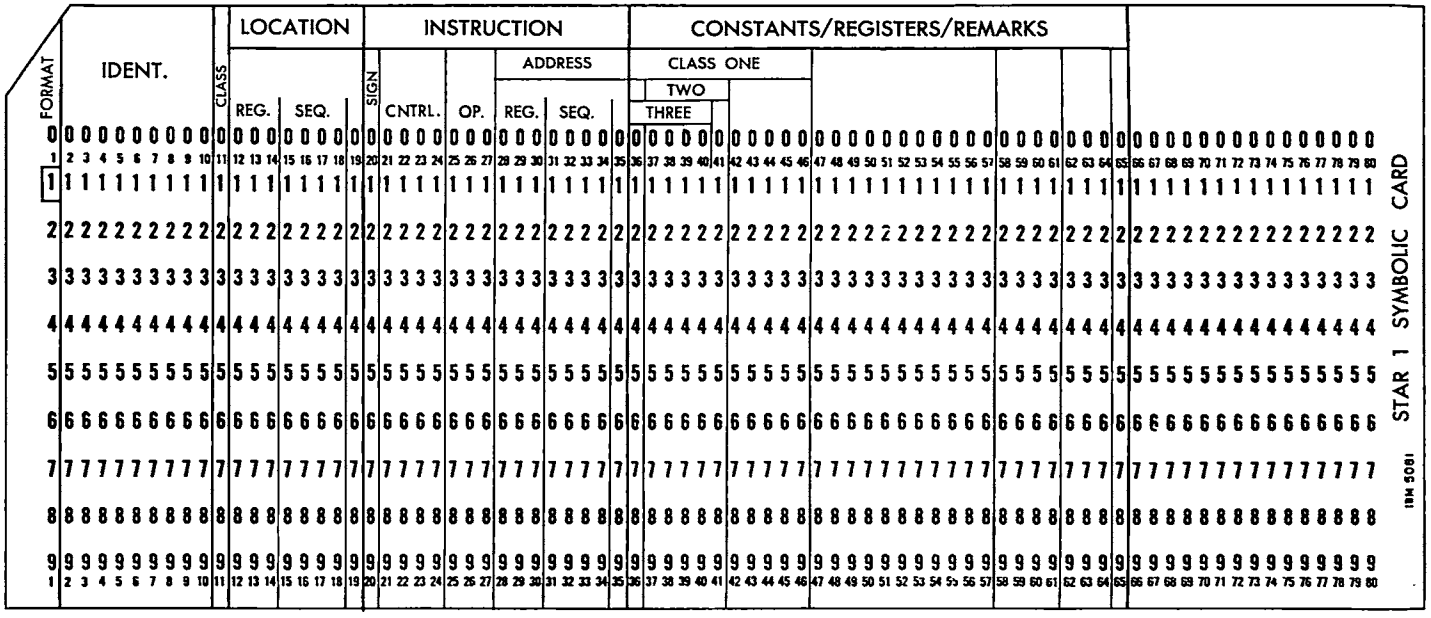
4) Provision for checking during assembly that unit designation digit "u", in the control field of input-output instructions, is compatible with the standard configuration of a particular installation.

There are a few parameters inside STAR 1 which particular installations may adjust easily. These include input-output unit designations and storage capacity limits.

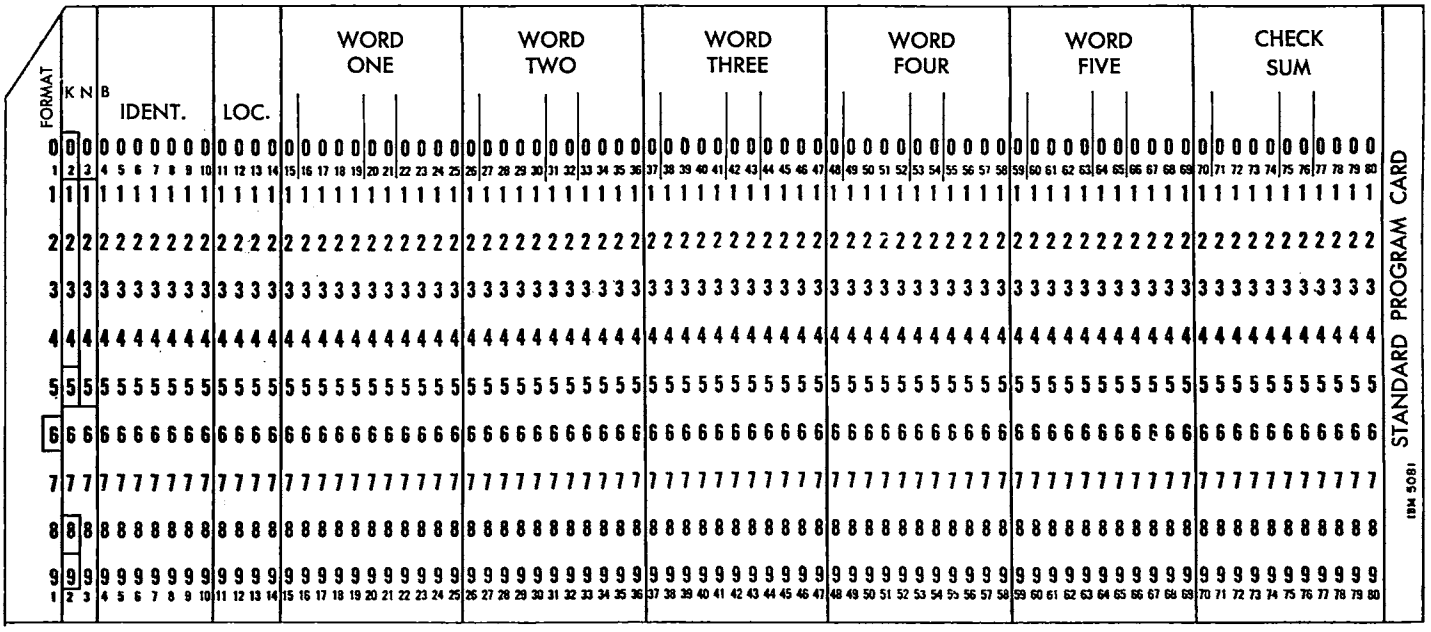
SECTION VI

APPENDIX



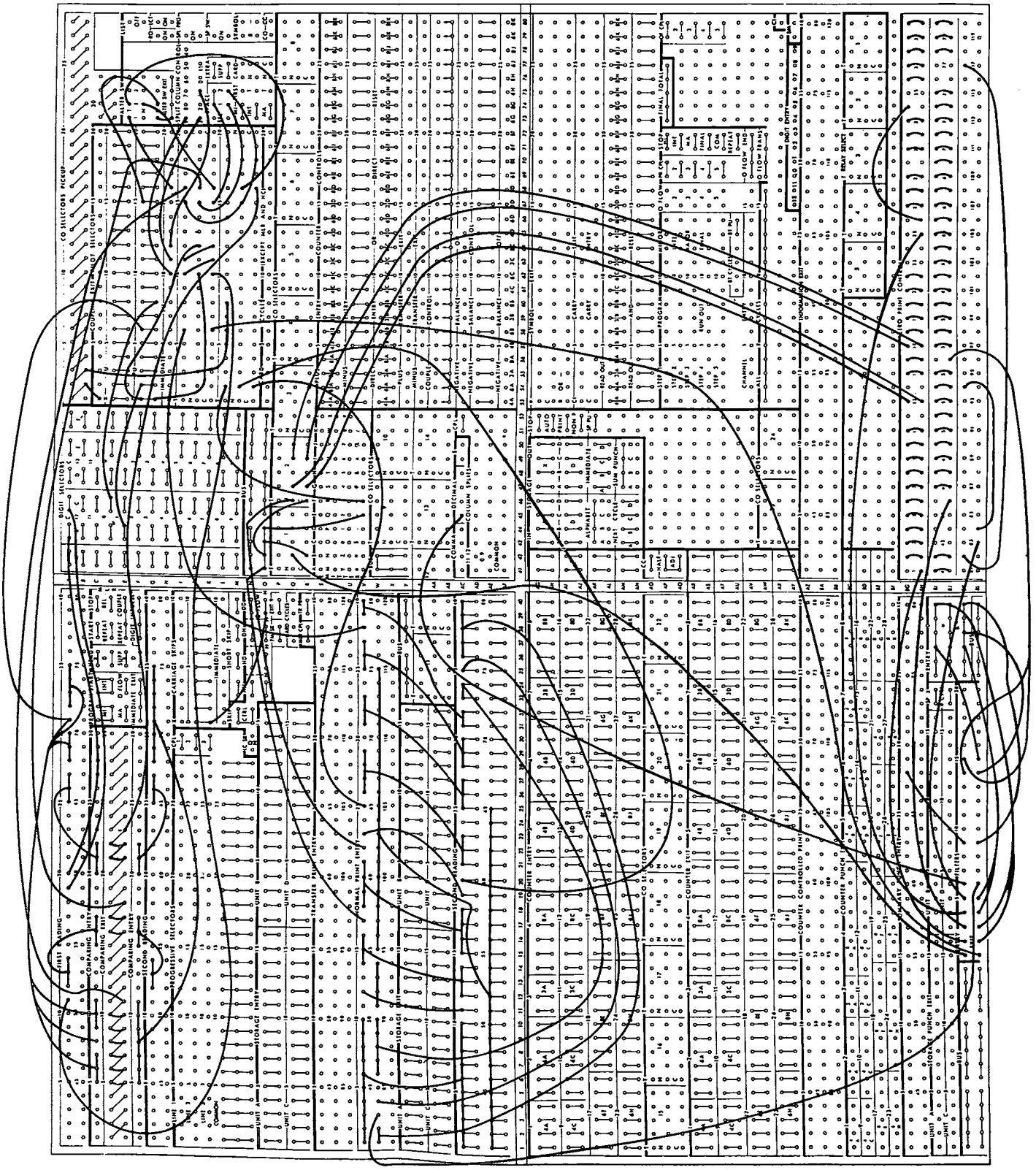


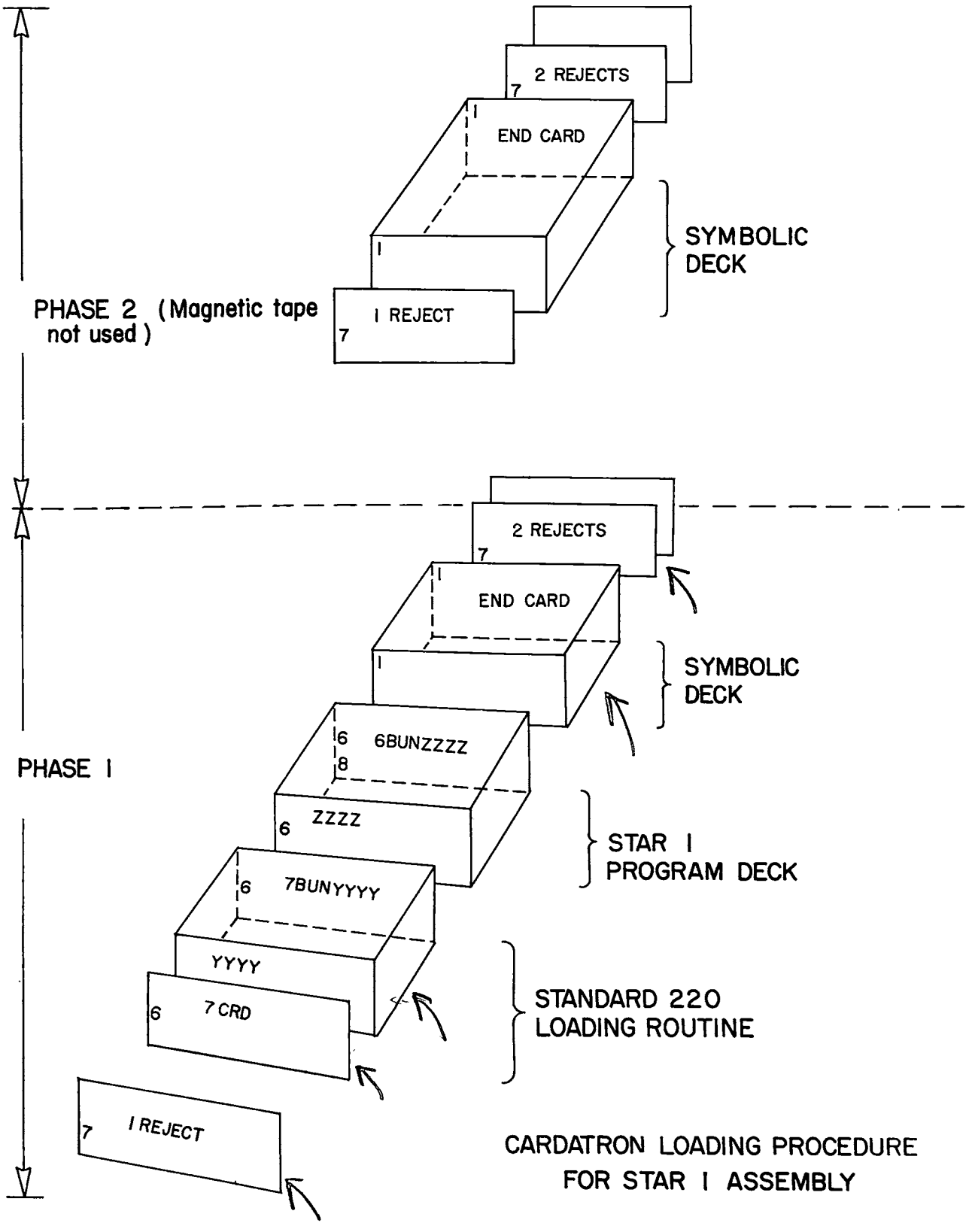
STAR 1 SYMBOLIC CARD LAYOUT



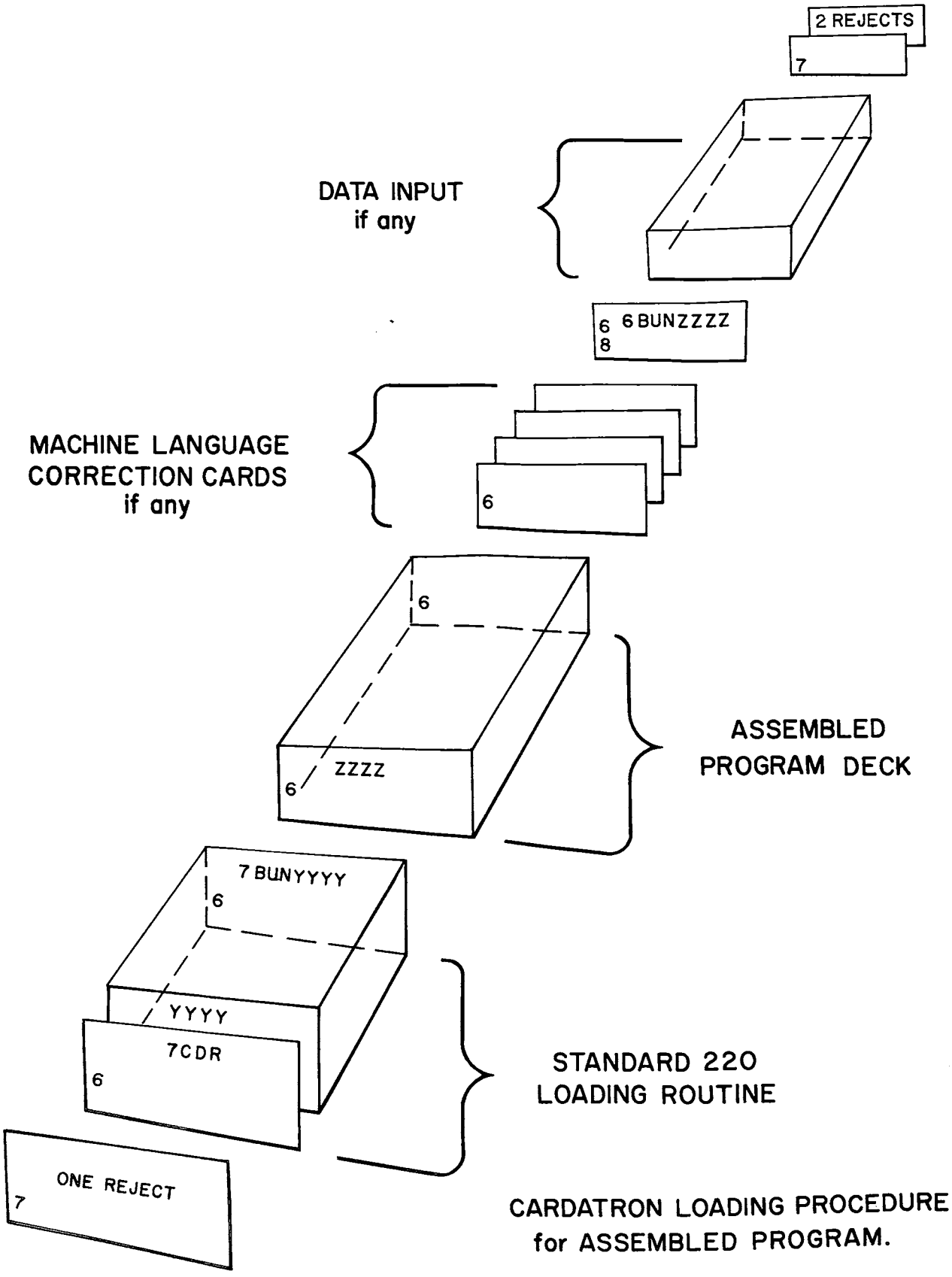
STANDARD 220 PROGRAM CARD LAYOUT

WIRING DIAGRAM OF STANDARD 407 PRE-ASSEMBLY BOARD





CARDATRON LOADING PROCEDURE FOR STAR 1 ASSEMBLY



CARDATRON LOADING PROCEDURE
for ASSEMBLED PROGRAM.

4500 4501 4502 4503 3178 3177 3176 3175 3174 3173 3172 3171 3170 3169 3168 3167 3166 3165 3164 3163 3162 3161 3160 3159 3158 3157 3156 3135 3134 3133 3132 3131 3130 3129 3128 3127 3126

BURROUGHS CORPORATION
ELECTRODATA DIVISION

BURROUGHS 220 STAR I CODING FORM

UNPUNCHED REMARKS	CLASS	LOCATION REGION	LOCATION SEQUENCE	IN	S	INSTRUCTION CONTROL	OP	ADDRESS REGION	ADDRESS SEQUENCE	IN	Increment CONSTANTS/REGISTERS/REMARKS CONSTANTS/A-REGISTER	R-REGISTER	B-REG.	SCALE	REF
	11	12 – 14	15 – 18	19	20	1 2 3 4	25 – 27	28 – 30	31 – 34	35	36 S 1 2 3 4 5 6 7 8 9 0 46	47 S 1 2 3 4 5 6 7 8 9 0 57	58 7 8 9 0 61	62 ± D/P 64	65
Calling Sequence to Region 11.		10	0				STP	11	0						
			1				BUN	11	1						
			2			1234	56	45	0		SINE-49 WORDS				
			3			7531	97	193	0		MATRIX INVERSION-212 WORDS				
11.0 →			4		-	9012	34	870	0		SQUARE ROOT-24 WORDS				
Continuation of Region 10.			5												
10.1 →		11	0				BUN				EXIT OF SUBR				
Subroutine to read subroutines on cards.			1				LDB	3	0				10.2		
			2		-		LDR		0000		12345645.0				
			3		-		LDB		0000				45.0		
			4				STP		0000		EXIT OF LOADING ROUTINE				
Loading Routine Return →			5				BUN		0001		ENTRANCE OF LOADING ROUTINE				
			6				LDB	3	0				10.2		
			7		-		CAD		0000						
			8			0401	IFL	3	0						
			9			1	BSA	3	0		TO EXIT IF LAST ENTRY WAS				
			10				BUN	3	1		JUST PROCESSED.				
Storage reservation and regional-symbolic identification.	6	45	0		·				0049		SINE-ID 123456				
	6	193	0						0212		MATRIX INV-ID 753197				
	6	870	0						0024		SQUARE ROOT-ID 901234				

PROGRAMMER A.R.Friedenheit CHARGE NO. 7440-82 DATE _____ PAGE _____ OF _____

ROUTINE NAME SAMPLE CODING for SUBROUTINE LOADING IDENT. NO. 1 REGION NO. _____

Printed in U.S.A. 1420-A

"SAMPLE" ASSEMBLY LISTING, WITH NOTES.

```
3126  0000130012                609.0058          SUB    1.0002  MINUS PARTIAL FICA
3127  0001126239                609.0059          ADA  850.0039  ADD DEDUCTION 3
3128  0001490002                609.0059.4        SLT     .0002   XXXXXXXXYY Y
3129  1105373134                609.0059.5  1105  BFR    3.1000
3130  0000160000                609.0059.7        RND     .0000                ALL CLEARED
3131  8810400075                609.0060    8810  STA    2.0015  XXXXXXXXYY STORE TAX      08
3132  0000407851                609.0061          STA  930.0011  STORE FOR PRINT OUT
3133  0000300000                609.0999          BUN     .0000   EXIT OF REGION.
3134  2401266212 11             609.1000    2401  IFL  850.0012  STEP CTR. FOR 5 ROUNDING
3135  0000303130                609.1001          BUN    3.0059.7
                              6 609.9999                 .0020  LEAVE 20 EXTRA CELLS.

                              9 610.0000                        *ALPHA BETA
                              9 610.0000                        *SUBROUTINE
3156  0000300000                610.0000          BUN     .0000  EXIT
3157  4400283158                610.0001    4400  DLB    3.0002                      0009
3158  0009190076 02             610.0002    0009  ADL    2.0016
3159  10000407940               610.0003   -0000  STA  930.0100  STORE ARGUMENT TEN
3160  0001213159                610.0004    0001  DBB    3.0003   TIMES IN OUTPUT REGION
3161  3000100065 01 33          610.0005    3000  CAD    2.0005.2  00XXXXXXXX              07
3162  0000143178                610.0006          MUL    3.0900  -00000ZZZZZ-ZZZZZZZZZZ  11
3163  0000400080                610.0007          STA    2.0020
3164  0001400081                610.0008    0001  40     2.0021
3165  0000360000 30             610.0009          BFA  611.0000
3166  0000410074                610.0020          LDR    2.0014  CODE WORD  SWWWHHHEEEE
3167  3611183174 10             610.0021    3610  CFR    3.0040  TEST WHETHER HHH BEYOND 799
3168  0000343171                610.0022          BCH  610.0024
3169  0000354500                610.0022.5        BCE  612.0001  HHH EQ 799-SPECIAL PROCESSING
3170  0031093175 20             610.0023    0031  SPO    3.0050
3171  0001407952                610.0024          STR  930.0112  STORE FOR OUTPUT
3172  0401266212                610.0030    0401  IFL  850.0012  STEP NNNN, ENTRIES-IN CTR.
3173  0000303156                610.0031          BUN    3.0000  TO EXIT
3174  0007990000              1 610.0040                         0007990000 HHH LOWER LIMIT
3175  24848480063             2 610.0050                         HHH T
3176  25656006254             2 610.0051                         00 SM
3177  24153530016             2 610.0052                         ALL CR
3178  10003145926             1 610.0900                        -0003145926 PI            04

                              5 612.0000    0001                .4500  *SPECIAL PROC. HHH EQ 799
4500  0000410074                612.0001          LDR    2.0014                SWWWHHHEEEE
4501  3300001973 21             612.0003    3300  BFB  62.0008.5  BRANCH IF WWW ALL ZEROS
4502  0000420065                612.0005          LDB    2.0005  CODE 0075-0174          CODE
4503  10000307765             3 612.0007   -0000  BUN  930.0000  -0075 TRANSFER TO TABLE ENTRY
```

Note the following points:

The origin of Region 001 is 0010, since 0010 + 0002 = 0012.
The origin of Region 850 is 6200, since this too is normally a Type A region.
Variation designation digit of 1 automatically inserted at left of op code 49.
Address 3.1000 refers to 609.1000, assigned absolute location 3134.

The origin of Region 002 is 0060.
The origin of Region 930 is 7840.
Prestored address; might be noted by reference flag digit in column 65 (optional).
11 Error Flag: Erroneous sLnn

Class 6 Control Card which steps location counter by 20.
Spaces between regions.
Two lines of heading using Class 9 cards.

STP Exit in first word of subroutine.
Note that using the control digits field of the following entry for a B register setting results in "Possible Error" Flag 02: iiii not equal to 0000.

01: iiiX control field not equal to 000X.
33: Decimal insert digit of Type A region address not equal to zero.

Numeric operation code, complete with applicable variation designator digit.
30: Region specified in symbolic address not in table.

10: Erroneous sLf; here L greater than S + 1.
Region 003 notation not used in this address.

20: Wrong variation digit for this alphanumeric operation code; left as is.

Remember that all 407 output must be in CAPITAL letters in Remarks Field.

Numeric constant (Class 1) entered.
Alphanumeric constants (Class 2) entered.

CR stands for Carriage Return symbol.
Numeric constant; decimal point 4 places from left. See instruction in 3162, note record of decimal point (07 + 04 = 11).
Class 5 Control Card specifying new origin, Type B region.

21: Operation code specified not in table; stored as 00 in absolute word.

7765 + 0075 = 7840, origin of Region 930 (see 3132 for confirmation).

Burroughs Corporation

ELECTRODATA DIVISION
PASADENA, CALIFORNIA

BULLETIN 5054

Printed in U.S.A.