Russell Larson

APOLLO GUIDANCE COMPUTER

Information Series

ISSUE 13

YUL PROGRAMMING SYSTEM

FR-2-113

5 December 1963

TABLE OF CONTENTS

TABLE OF CONTENTS (continued)

LIST OF ILLUSTRATIONS

LIST OF TABLES

AGC INFORMATION SERIES

ISSUE 13

YUL PROGRAMMING SYSTEM

FR-2-113

13-1. INTRODUCTION

13-2. This is the thirteenth issue of the AGCIS, published to inform the technical staff at MIT and Raytheon about the AGC and the Apollo G & N System. This issue is a description of the Yul system and contains most of the rules necessary for writing an AGC program.

13-3. The development of the Yul Programming System started several years ago. At that time the Instrumentation Laboratory of MIT worked on a spaceborne computer with a fixed and an erasable memory of which the fundamental efforts were supposed to be finished by Christmas 1959. The computer became known as the Christmas computer, and the programming system, created by Hugh Blair-Smith, as the Yul system. Later, the Yul system was further developed by Mr. Blair-Smith to become the programming system for the AGC. In a series of meetings held during August through November, 1963, Mr. Blair-Smith described and demonstrated the Yul system to Richard L. Volpi and assisted in writing this issue.

13-4. GENERAL SYSTEM DESCRIPTION

13-5. The Yul Programming System (Yul system) is a software system (i.e., a system consisting of programs or a set of programs) used with the Honeywell 800 Data Processing System. The purpose of the Yul system is to aid in the preparation of programs for the AGC and related computers.

The Mars Simulation System, a software system developed by Richard Warren of MIT/IL, is used to check out such programs.

13-6. HARDWARE ITEMS AND PROGRAMS INVOLVED

13-7. The Honeywell 800 Data Processing System (H-800 system) of the MIT Instrumentation Laboratory consists of an H-801 Central Processor with 32,768 forty-eight-bit words of memory and floating point option, ten tape drives, a five-megaword disc file, an H-823-2 Card Reader (650 cpm), an H-824-2 Card Punch (250 cpm), an H-822-3 High-Speed Printer (600 or 900 lpm), a paper tape reader (1000 cps), a paper tape punch (110 cps), a Calcomp 564 plotter, an on-line clock, and a console typewriter (cpm means cards per minute, lpm means lines per minute, and cps means characters per second). In December 1963 the H-800 system will be changed to an H-1800 system, which works three times as fast as the H-800 system. The H-1800 system has an H-1801 Central Processor instead of the H-801; otherwise the H-1800 system differs very little from the H-800 system described above.

13-8. Other hardware items which are or might be involved with the Yul system are the IBM-026 Printing Card Punch for punching input information, the IBM-519 Document-Originating Machine for reproducing and transcribing (numbering) punched cards, the IBM-557 Interpreter for printing cards, the IBM-082 Sorter for sorting card decks, and the IBM-407 Accounting Machine for printing out programs before assembly.

13-9. Most programs executed on the H 800 of the MIT/IL,including all Yul and Mars activities, are run under the control of the MIT/IL Monitor. The Monitor is a set of programs stored on magnetic tapes and discs which cause jobs to be run with a minimum of human intervention. The Monitor

maintains all the H-800 programs which are frequently used on the MIT/IL System Tape. Monitor cards (80-column cards with Monitor codes punched) are used to direct the Monitor to load particular programs into the main memory of the H 800.

13-10. When the H-800 system operates under the Yul system on AGC programs, the MIT/IL System Tape, the tape YULPROGS, and three work tapes have to be mounted on the tape drives. Any Yul programming task is executed by feeding task definitions and information (punched on 80-column cards supplied by a programmer) into the Yul system. The operator of the H-800 system supplies monitor cards, which are fed into the H-800 system first and cause the Monitor to bring in the Yul system. Normally, more than one Yul system task is carried out during one job, in which case the H-800 operator enters his monitor cards into the card reader of the H 800 first and then all the card decks related to Yul system tasks. The H-800 operator also provides an ENDJOB card, which signals the MIT/IL Monitor that all Yul tasks have been completed. As a Yul task is performed by the H-800 system, results are printed out by the H-822-3 printer and certain messages are typed out by the console typewriter. The H-800 system is able also to produce information on punched cards or punched tapes.

13-11. BASIC YUL SYSTEM ACTIVITIES

13-12. The discussions in this issue deal mainly with the creation of AGC 4 programs but apply similarly to other AGC models and similar computers. The fixed memory of the AGC stores various information and programs. In order to be able to wire a program into a core rope (storage device of fixed memory), it is necessary to create that program and to specify how it has to be wired into a certain core rope. Normally, a mathematician supplies the necessary equations or mathematical functions, an engineer specifies the

control functions of the Apollo system involved, and a programmer writes a particular program (in Yul language). Once the program has been written, it has to be assembled, checked, and tested, and information for manufacturing a core rope has to be produced. Most of these functions are normally performed by the Yul system.

13-13. The basic activities of the Yul system are:

a. Assembling an AGC program (or subroutine)
b. Maintaining tape YULPROGS
c. Miscellaneous functions
d. Simulating an AGC program, and
e. Manufacturing an AGC program.

At this time the Yul system is able to deal with AGC programs only. In the future the Yul system will be able to deal with AGC subroutines in a similar manner, as indicated by parenthetical expressions in the following discussions. An AGC program is defined as the entire content of F memory of an AGC. An AGC program might consist of several parts or sections. Also, AGC programs might vary with their purpose. Different programs might be written to develop or test different techniques, and they might be written by different groups of people before a program for a certain mission can be chosen. Program parts which are used frequently for the generation of (entire) programs will be classified as AGC subroutines, and the Yul system will be able to deal with AGC subroutines independently from (entire) programs. Established subroutines will be protected against revisions.

13-14. Assembling an AGC program (or subroutine) means translating it from Yul language (input information) to binary record (usable information). Inputs for an assembly are 80-column cards containing AGC programs (or subroutines) in Yul language. The outputs of an assembly are an assembly listing and a printout on the console typewriter. The assembly listing contains program information in Yul language with octal equivalents and various

tabulations (such as symbols defined, AGC storage used, etc). If the assembly of a program is judged "good" or "fair", a binary record of the program (including binary records of subroutines used by the program) is added to the assembly. (The expression "good" means without any error; "fair" means without serious errors.) Input information and binary records are stored on tape YULPROGS; intermediate results are stored temporarily on work tapes. The console typewriter supplies information in regard to the Yul and H-800 activity.

13-15. Maintaining tape YULPROGS means keeping the latest versions (revisions) of each AGC program (or subroutine) on tape YULPROGS. All routine work of this type is done automatically during assembly.

13-16. Miscellaneous functions which are of interest to an AGC programmer are deleting a program (or subroutine) and punching a Yul language deck of a program (or subroutine) from tape YULPROGS. More miscellaneous functions are provided, but most of them are of no interest to an AGC programmer.

13-17. Simulating an AGC program means testing it by causing the H-800 system to execute the program as an AGC would execute it and to print out intermediate and final results. Inputs for a simulation are 80-column cards specifying which AGC program is to be tested and the test conditions. Outputs are edited and annotated accounts of the AGC program actions. A very simplified simulation for AGC 4 programs exists as an integral part of the Yul system. All more complex simulations are usually done by a separate piece of software called the Mars system. When a Mars simulation is to be carried out, the Yul system transfers information to the Mars system, which completes the test and provides results without further help from the Yul system. The Mars system has the capability also of combining the test of an AGC program with the simulation of environment. This activity is run under

the control of the MAC system (MIT/IL's Algebraic Compiler and Executive, not to be confused with the Machine Aided Cognition of the MIT Computation Center).

13-18. Manufacturing an AGC program means translating a binary record of a correctly assembled program into some form suitable for further processing by means other than the Yul system. Inputs for manufacturing are 80-column cards specifying the program (or subroutine) and the type of manufacturing. Outputs are punched paper tapes, punched cards, and/or diagrams used in wiring core ropes. Similar outputs can be generated for loading data and programs into AGC test equipment (including the core rope simulator).

13-19. TAPE YULPROGS AND THE SYSTEM TAPE

13-20. Tape YULPROGS consists of various tables and catalogues, and three main files, known as SYPT, SYLT, and BYPT. Refer to figure 13-1. SYPT, or Symbolic Yul Program Tape, consists of card images of all active programs in the original symbolic Yul language. (SYLT, or Symbolic Yul Library Tape, will consist of card images of all active subroutines in the original symbolic Yul language.) BYPT, or Binary Yul Program Tape, consists of binary records generated by a successful assembly. This binary record (actually several physical records on tape YULPROGS) is the binary representation of a program. In the case of SYPT (and SYLT), card images are maintained only for the latest revisions of all programs (or subroutines). Records on tape BYPT exist only for programs (including binary records of subroutines involved) and only when the assembly of the latest revision was judged "good" or "fair". The directory of computers presently lists only the MOD 3C computer (AGC 3) and the AGC 4.

13-21. The MIT/IL System Tape holds all the H-800 programs which are frequently used (as mentioned in paragraph 13-9). The various Yul system

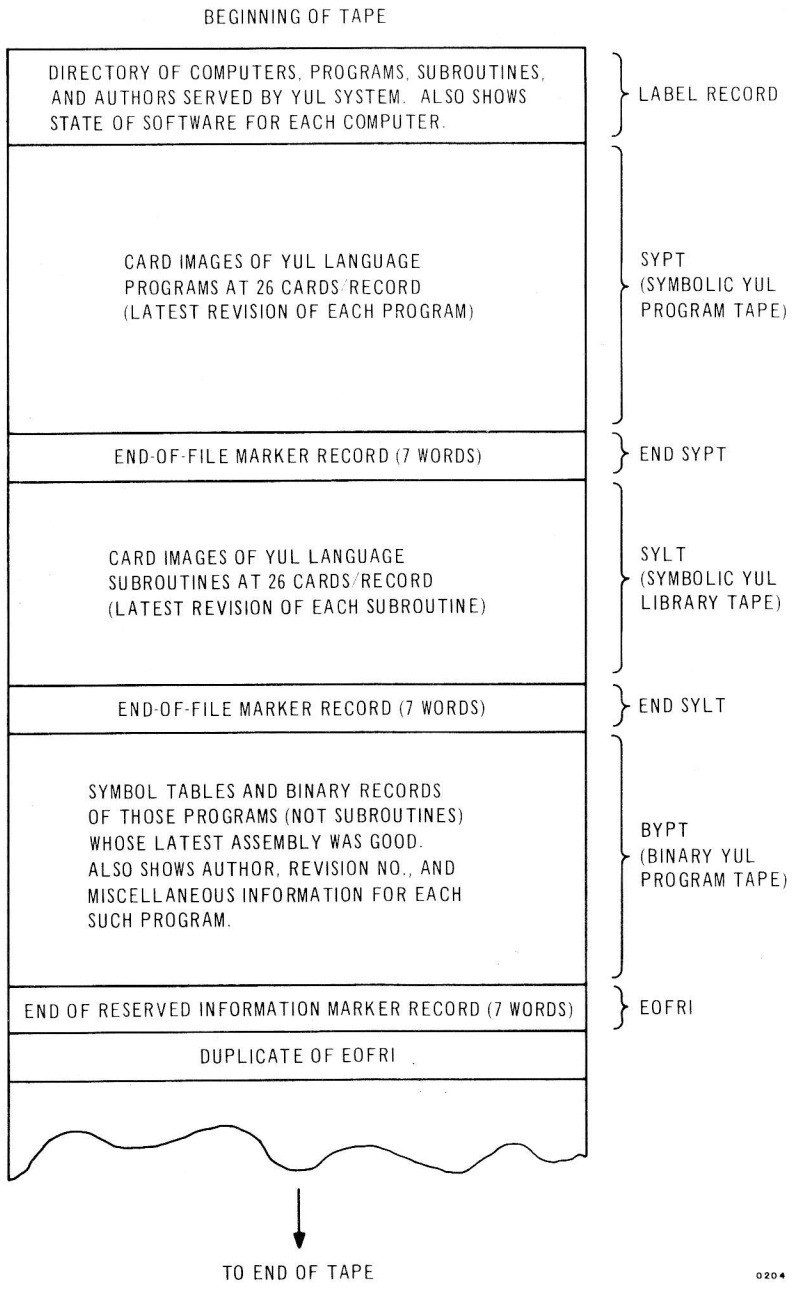


Figure 13-1. Tape YULPROGS

programs stored on the MIT/IL System Tape and the ways in which they function are indicated by figure 13-2. Presently, Yul system programs are provided only for the creation of AGC 3 and AGC 4 programs, but programs for the creation of programs for other AGC models might be added any time. The Yul system programs (PASS0 through MISCGA) are punched on 14,000 cards. Listings of the Yul system program can be printed in ARGUS (automatic routine generating and updating system) language or in MITIGUS language (MIT/IL dialect of ARGUS). The ARGUS language is for the H-800 system that which the Yul language is for the AGC.

13-22. PROGRAMMING EXAMPLE

13-23. Although Yul language cards and various printouts will be discussed in detail later, it is desirable to show samples now. For this purpose the demonstration program TRIVIUM was created, and copies were printed by the Yul system to generate one sample listing for each copy of Issue 13. Attachment 13-1 is one of the 25 Yul language cards used for feeding information into the Yul system. Attachment 13-2 is a printout of the IBM-407 Accounting Machine with the 80-80 Board inserted (see paragraph 13-64). This printout lists all information contained on a Yul director card (Y in column 1) and the 25 detail cards as punched for the sample program. The Yul director is needed to direct the Yul system to carry out a certain task. The detail cards are used for supplying information. Attachment 13-3 is a printout of the IBM 407 with the Honeywell Format Board inserted. This printout is similar to attachment 13-2, but some information is not printed and other information is printed in positions reflecting the format of a Yul listing. The Yul director card was not inserted with the card deck deliberately, because it would have been printed on a separate sheet since a log card (L in column 1) always skips to the next page before printout of its information content. Attachment 13-4 is a Yul assembly listing printed by the H-822-3 High Speed Printer. An assembly listing consists of:

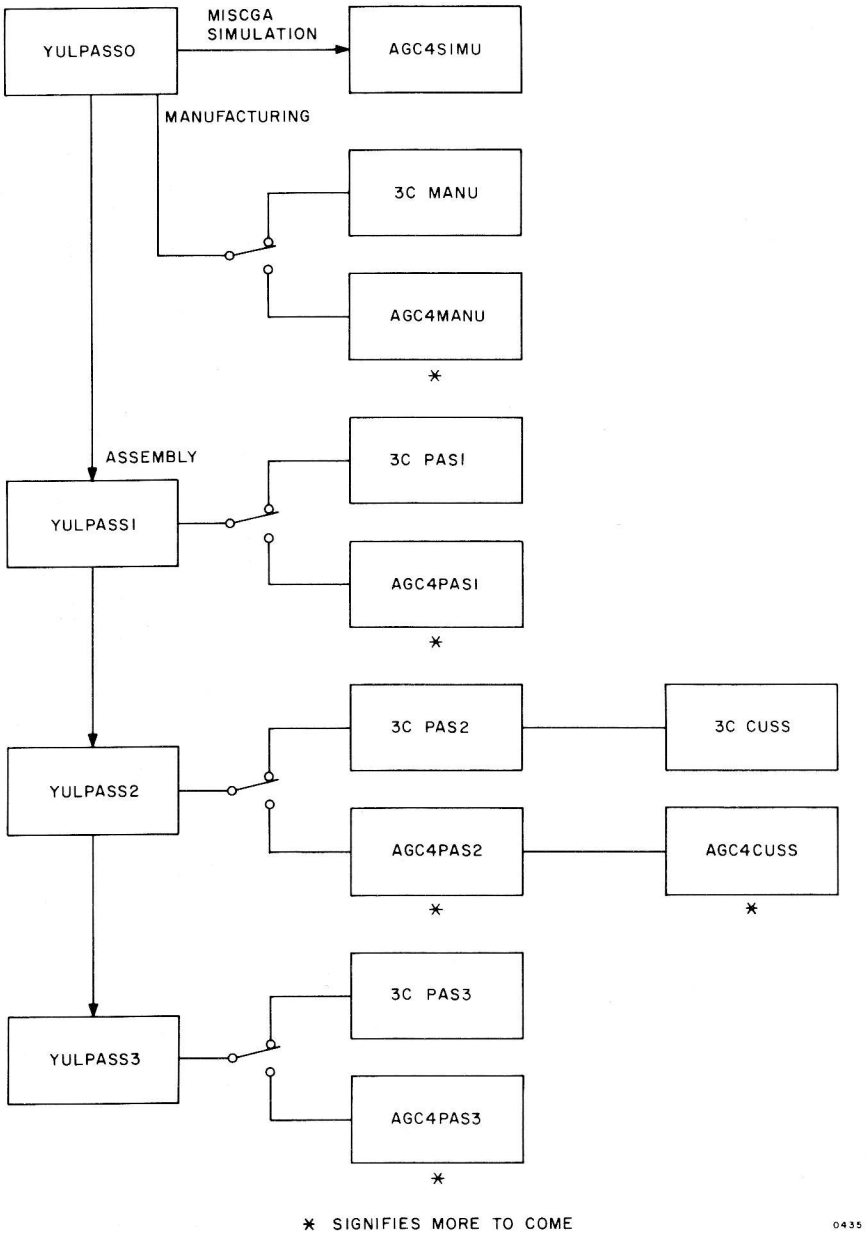


Figure 13-2. Yul System Programs on MIT/IL System Tape

a. A program or subroutine listing (only one sheet in the case of program TRIVIUM)

b. A list of symbols (only one sheet)
c. A summary of symbols (page 3)
d. A directory of AGC memory (page 4)
e. An index to occupied locations (page 5)
f. A list of paragraphs (page 6), and
g. A map of used AGC memory (page 7).

} Summary of listing

A paragraph is a set of 256 locations in AGC memory; only paragraphs of which at least one location is used are printed in the list of paragraphs. The sample assembly was judged "bad", as stated at the bottom of page 7. Attachment 13-5 is a photocopy of the printout of the console typewriter of the H-800 system made as program TRIVIUM was run the first time.

13-24. Assembling an AGC program (or subroutine) takes place in three Passes with the MIT/IL System Tape, tape YULPROGS, and three work tapes. Refer to figure 13-3. The input for Pass 1 consists of symbolic cards (detail cards) when a new program (or subroutine) is assembled. When a version of a program (or subroutine) is assembled from existing programs (or subroutines) or when an existing program (or subroutine) is revised, appropriate SYPT (or SYLT) records might also be used as input for Pass 1. Pass 1 forms tables in core memory of the H-800 system and writes symbolic and binary information on work tapes. The main purpose of Pass 1 is to determine the type of each detail card which supplied information, to assign the location in AGC memory for instructions and constants of a program, and to define symbols in the location field of instruction, constant, and clerical cards. An AGC programmer has the choice of letting the Yul system assign locations for instruction and constants during Pass 1 or of postponing such actions for selected instructions and constants (called leftover instructions and constants) until all regular memory reservations have been made. A programmer will see the advantage of this feature when working with CCS instructions. A one-,

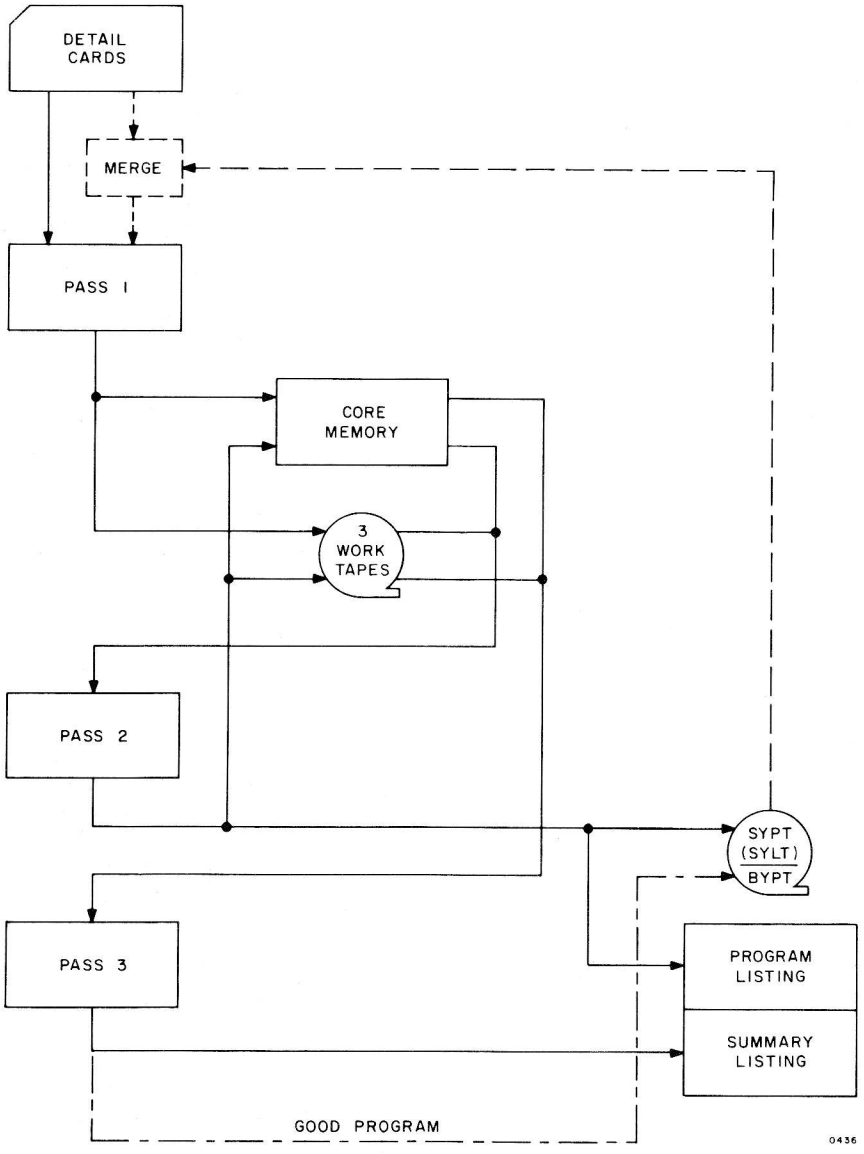


Figure 13-3. Assembly of a Program (or Subroutine)

two-,or three-word hole which can be filled later by one, two, or three leftover constants (or instructions) may occur. After Pass 1, every symbol which appears in the location field has a 15-bit definition, except symbols of leftover instructions, leftover constants, and leftover variables.

13-25. Pass 2 examines the address field of each instruction, evaluates relevant addresses represented by symbols defined during Pass 1, and defines leftover symbols as required. Pass 2 also evaluates constants, finishes the assembly (using and updating the tables in core memory), updates SYPT (or SYLT), makes up an unsorted binary record, and prints a program or subroutine listing. In this listing each detail card of Yul language is reproduced in full. Any pertinent binary equivalent is added (mostly in octal) on the same line, which is followed by as many lines of error or warning notices (cusses) as needed. Refer to line 0013 and the subsequent line of attachment 13-4. The vertical format originally punched in the cards is used to determine the vertical format of the listing.

13-26. Pass 3 prints a summary listing using the tables generated by Pass 1 and amended by Pass 2. On sheet 2 of attachment 13-4 the symbols used are listed in alphabetical order with the equivalent of each in octal notation and the condition under which it was defined or any error associated with it. On sheet 3 an account of defined symbols is given. Sheet 4 represents a directory of AGC memory showing the type of memory for each location and if the location is reserved or available. Sheet 5 relates the locations of words defined in the program to pages on which the words are defined. Sheet 6 lists the paragraphs generated for that program. Sheet 7 is a map of used AGC memory, printed from a binary record which was sorted according to locations during all the preceding Pass 3 printing. If the assembly is of a program and if the assembly has been judged "good" or "fair", the sorted binary record is inserted into BYPT.

13-27. The first line on each sheet of an assembly listing (attachment 13-4) is referred to as a main headline. This line is printed automatically on each sheet by the Yul system. The number at the beginning of the main headline is the log number, or serial number, of the job. This number and the date are supplied by the MIT/IL Monitor. The page number as well as the text of the main headline are maintained by the Yul system. The second line of sheet 1 of attachment 13-4 is referred to as a subheadline. A program or subroutine listing has no subheadline until the appearance of the first log card. From there on the content of the log card is printed as a subheadline on each page until the appearance of another log card. The words USER'S OWN PAGE NO. and the number are supplied automatically by the Yul system. Each new log card or subheadline breaks the sequence of user's own page number and starts a new count (with 1).

13-28. Most other lines of a program or subroutine listing (sheet 1 of attachment 13-4) start with a four-, five-, or six-digit decimal number identifying the card from which information was taken. The letter R preceding a card number indicates that information was taken from a (general) remark card. The letter A indicates that information was taken from an aligned remark card. The letter C has been entered by the Yul system to indicate that the content of a right-print remark card has been displaced into the next line. All types of remark cards contain explanations concerning a program but do not contribute to the operation of the Yul system. The letter E means error and identifies a complaint made by the Yul system. The six blots following the letter E are printed automatically instead of any card number so that a complaint can be noted easily.

13-29. Card numbers not preceded by a letter usually indicate that information has been taken from an instruction card, a constant card, or a clerical card. These cards contain information pertinent to the creation of a program. Line 0011, for instance, is a printout of information

contained on an instruction card plus information generated by the Yul system. The symbols WERNER (name of a certain location but undefined as yet), CAF (instruction code), and ONE (relevant address expressed by the quantity stored at it) were punched into card 0011. During Pass 1 the Yul system assigned and reserved the location in F memory (of AGC) where instruction CAF ONE will be stored. The symbol WERNER in the location field had been defined as meaning location 5510. The order code CAF in the operation field was translated to 3. During Pass 2 the symbol ONE in the address field was examined; the Yul system recognized that the address field contained a symbol, looked up the definition of the symbol, and formed value 3 5776 for instruction CAF ONE. (The symbol ONE was defined during Pass 1 to mean address 5776, as specified by numeric constant card 0008.) During Pass 2 the parity (1) for word 3 5776 was computed and the octal equivalent 3 5776 1 for instruction CAF ONE was printed in front of the symbol WERNER, together with location 5510 assigned earlier.

13-30. Program or subroutine listings printed from tape YULPROGS show location values for each word. Location values of central registers, E memory, and FF memory (table 1-4) are always printed in four octal digits (0000 through 7777). Location numbers of FS memory are printed with their octal bank number in front of the real address and separated by a comma (for instance 03, 6510). The Yul system uses octal bank numbers 03 through 14 and 21 through 34 when working on AGC 4 programs (table 1-4). Originally, decimal bank numbers 0 through 9 were used.

13-31. CREATION OF AN AGC PROGRAM

13-32. Figure 13-4 illustrates in a very general way the creation of an AGC program. A new program must first be assembled. When a "good" or "fair" assembly has been made, the program can be tested either by a simulation program in the H-800 system (with or without environment

simulation), or by running the program on an AGC with a core rope simulator, or both. Once a program has been found satisfactory, all the information needed to produce a core rope can be generated by the Yul system.

13-33. Figure 13-5 indicates how a new program (or subroutine) is assembled. First one Yul director card, the subdirector cards desired, and all necessary detail cards have to be punched. The Yul director card directs the Yul system to assemble a new program and defines the new program. Subdirector cards can be added to specify how many copies of an assembly listing shall be printed, how many lines per page shall be printed, etc. Normally, one copy of an assembly is printed and 54 lines per page, including headlines and skipped lines. The detail cards contain all program information. Printing the contents of the cards on the IBM 407 allows proofreading. (See attachment 13-3.) When all cards are correctly punched and in correct order, sequential numbers are inserted in columns 2 through 5 of all detail cards. Thereafter, the card deck with the Yul director card on top is ready for processing by the H-800 system.

13-34. Figure 13-6 illustrates how an existing program (or subroutine) can be revised. This figure is similar to figure 13-5, except that automatic numbering of new cards is generally not feasible. The numbers of new cards are punched into cards as the cards are produced. The Yul system is able to replace the card numbers of a revised program (or subroutine) by inserting subdirector RENUMBER, in which case the assembly listing and any card deck punched from tape YULPROGS show the new numbers. Whenever a program is revised, its revision number, which is stored in the Yul system, is incremented by one. A new program has revision number 0.

13-35. Figure 13-7 shows how a version of a program (or subroutine) is

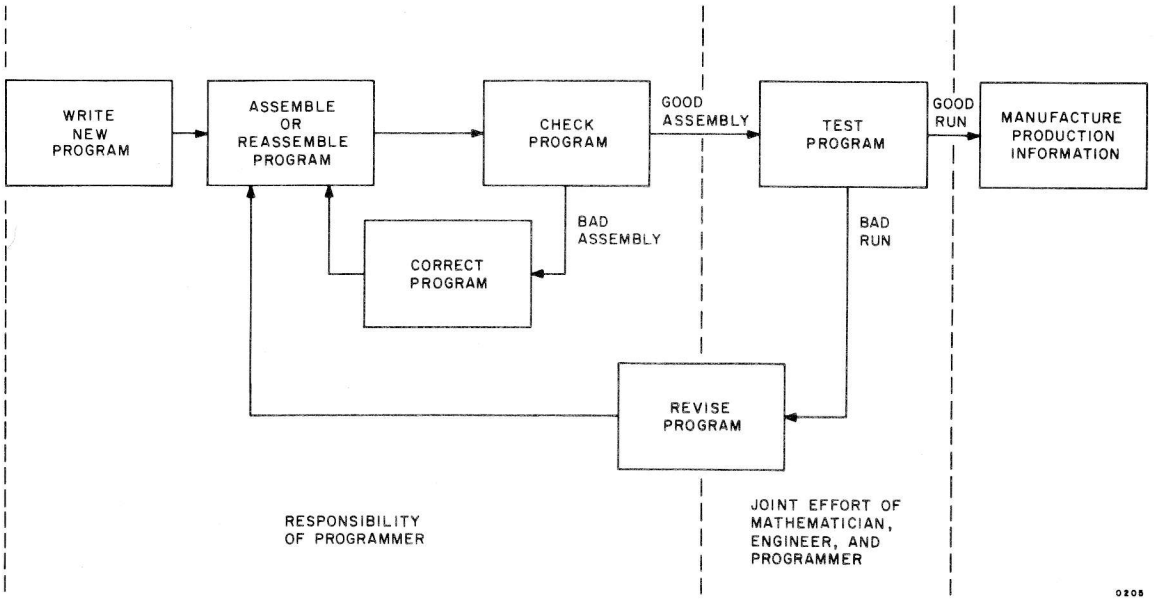


Figure 13-4. Creation of an AGC Program

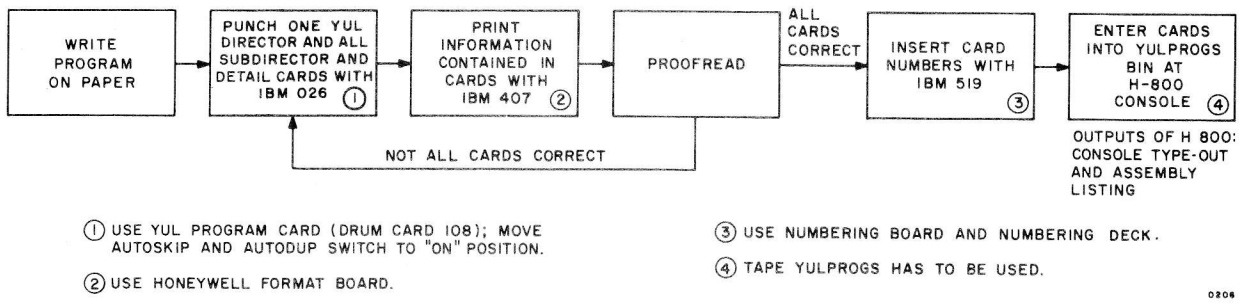


Figure 13-5. Assembly of a New Program (or Subroutine)

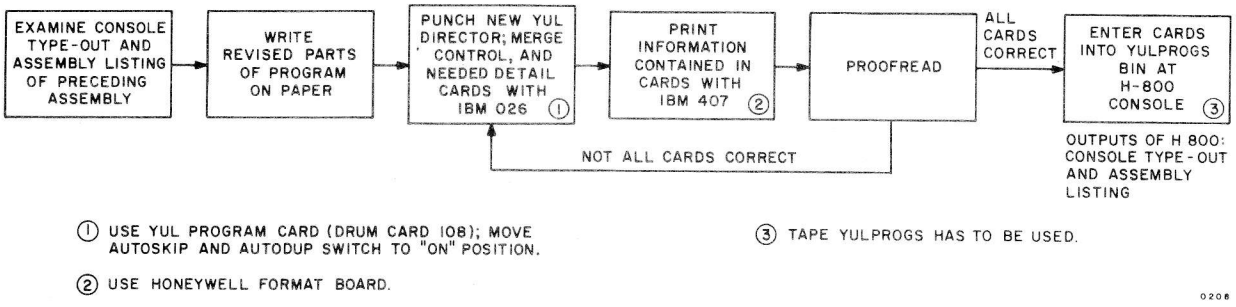


Figure 13-6. Revision of Program (or Subroutine)

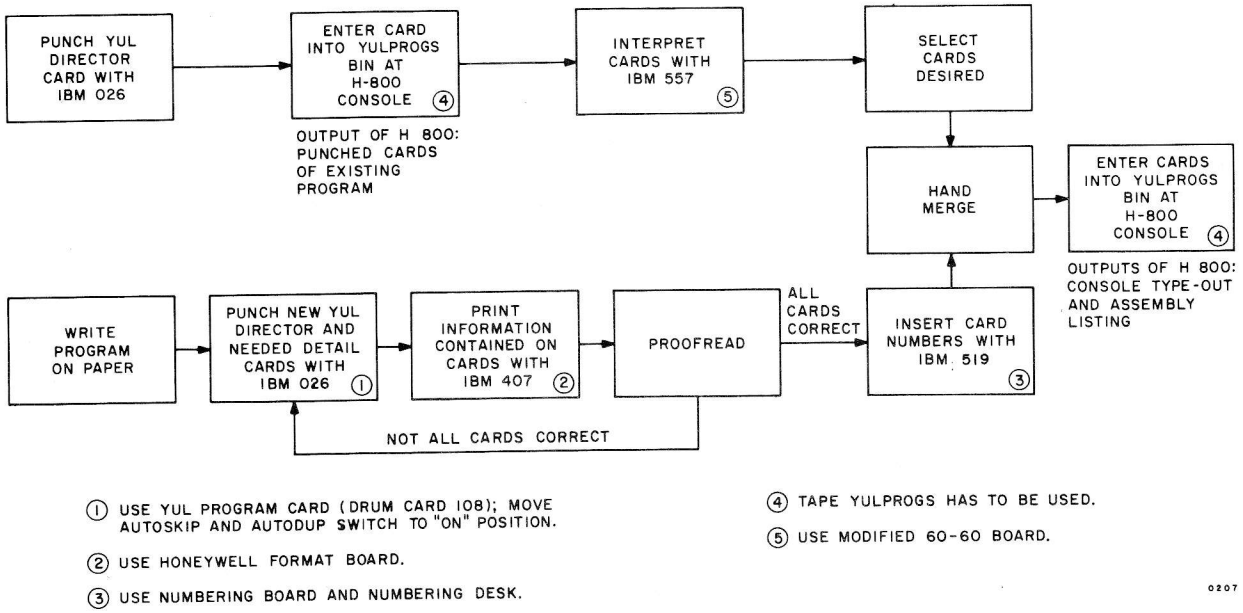


Figure 13-7. Assembly of a Version of a Program (or Subroutine), Old Process

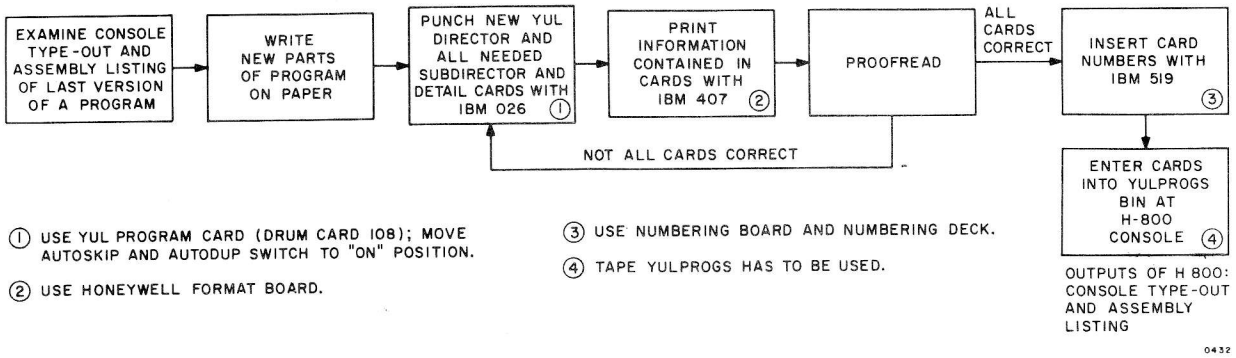


Figure 13-8. Assembly of a Version of a Program (or Subroutine), New Process

presently assembled. The new program might consist of parts of an existing program and of some new parts. A Yul director card specifying that existing program for which cards shall be punched by the H-800 system has to be punched. After the cards are printed, those which are needed can be selected. The new parts of a program are generated in a similar way, as indicated in figure 13-5. After the various program parts are merged, the combined card deck with a new Yul director card on top is ready for processing. Note that cards of the new program portions are numbered independently of previously existing cards.

13-36. Figure 13-8 shows a more convenient Yul process, called Version Assembly, which will be available soon. This process will combine card images of an existing program (or subroutine) on tape YULPROGS with input cards to form a new program (or subroutine) in one operation. First, one Yul director, one or more subdirectors, and all needed detail cards have to be punched. The Yul director causes the Yul system to assemble a new version of a program (or subroutine) and defines the name of the new program. Subdirectors specify which existing program is to be used, how many copies shall be printed, etc.

13-37. Figure 13-9 illustrates how the Yul system can be directed to reprint a program (or subroutine). Only a Yul director which directs the Yul system to print an assembly listing and specifies for which program (or subroutine) the listing is to be printed is needed. The content of tape YULPROGS is not changed during process Reprint. Subdirectors can be used to specify the number of copies to be printed, lines per page, etc. Normally more than one copy of an assembly can be printed only if the last assembly was judged "good" or "fair", no matter how many copies were required by a subdirector. In case of Reprint any required number of copies is printed, no matter whether the last assembly was "good" or "fair" or not. Subdirector RENUMBER must not be used with Reprint.

13-38. Figure 13-10 indicates how the Yul system is used to punch a Yul language card deck for a program (or subroutine) stored on tape YULPROGS. Only a Yul director which directs the Yul system to punch cards and specifies the program (or subroutine) for which a card deck is to be punched is needed. The content of tape YULPROGS is not changed during process Punch.

13-39. Figure 13-11 indicates how an obsolete program (or subroutine) can be deleted from tape YULPROGS. The Yul director card specifies which program is to be deleted.

13-40. In case a deck of numbered cards has been dropped, the deck can be rearranged with the help of the IBM-082 Sorter. Care must be taken that cards belonging to different numbering sequences are separated first. The process of figure 13-7, for instance, produces a program consisting of at least two numbering sequences. Using cards of different colors for different numbering sequences makes separating them easier.

13-41. An assembly listing (output of procedures illustrated in figures 13-5 through 13-8) can be grammatically correct or incorrect and semantically correct or incorrect. If an assembled program concurs with all rules of the Yul language, it is called grammatically correct. When a program contains errors in grammar, these errors are indicated in the assembly listing, and revising or rewriting the program is necessary. When a program has been assembled correctly, this is indicated by the words "good assembly" contained in the console type-out and by a similar message at the end of the assembly listing.

13-42. If an assembled program performs those functions requested, it is called semantically correct. Before a program is tested, one has to verify that the console type-out and the assembly listing of the last revision

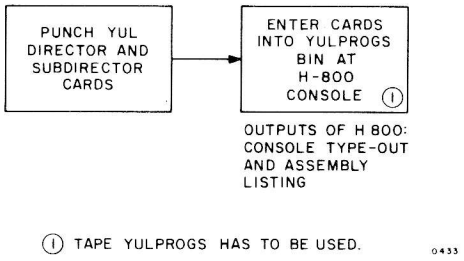


Figure 13-9. Reprinting of a Program (or Subroutine)

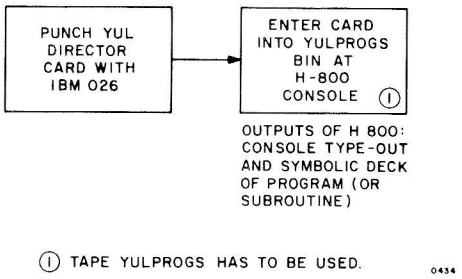


Figure 13-10. Punching of a Program (or Subroutine)

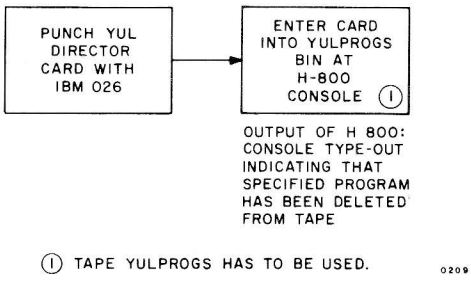


Figure 13-11. Deletion of a Program (or Subroutine)

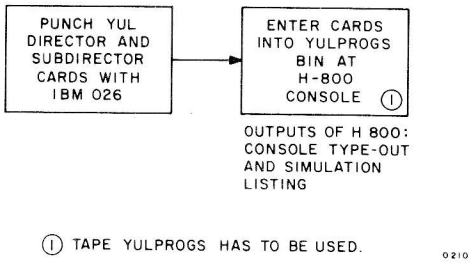


Figure 13-12. Testing a Program by MISCGA

contain the words "good assembly", "fair assembly", or a similar statement. Four different means are presently provided for testing a program:

a. Program MISCGA (AGCSIM spelled backwards) can be used to simulate an assembled program. This is a simplified test which does not consider computer inputs and outputs or program interrupts. Program MISCGA mainly checks subroutines of an AGC program.

b. The Mars simulation bench test is a moderate test which does not consider environmental conditions.

c. The Mars simulation flight test is a sophisticated test with environmental conditions.

d. The AGC prototype test is carried out with an AGC computer using a core rope simulator which is set by means of a core rope simulator load tape. The latter can be generated by the Yul system.

13-43. Figure 13-12 illustrates how a program is tested by means of program MISCGA. The Yul director and subdirector cards specify that this particular test shall be carried out with a certain program. The two types of Mars simulations are temporarily outside the Yul system and must be run as separate jobs.

13-44. Figure 13-13 indicates how a program is tested on the AGC 4 with a core rope simulator. The Yul director and subdirector cards specify that a punched tape of a certain program is to be produced. This tape is used to load the core rope simulator of the AGC 4.

13-45. Once a program has been tested and found to be satisfactory, it can be "manufactured"; i.e., a master deck can be punched and/or all information which is needed for wiring a core rope can be generated. Figure 13-14 shows how the Yul system can produce a master deck which contains all information in regard to the program to be wired into core ropes but not the way in which the ropes are to be wired. The latter information can be derived from the card deck by other processing methods.

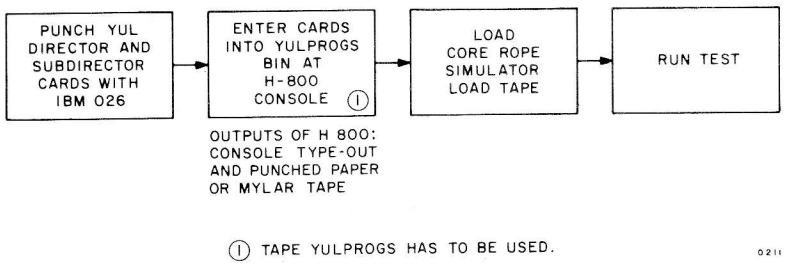


Figure 13-13. Testing a Program on an AGC with a Core Rope Simulator

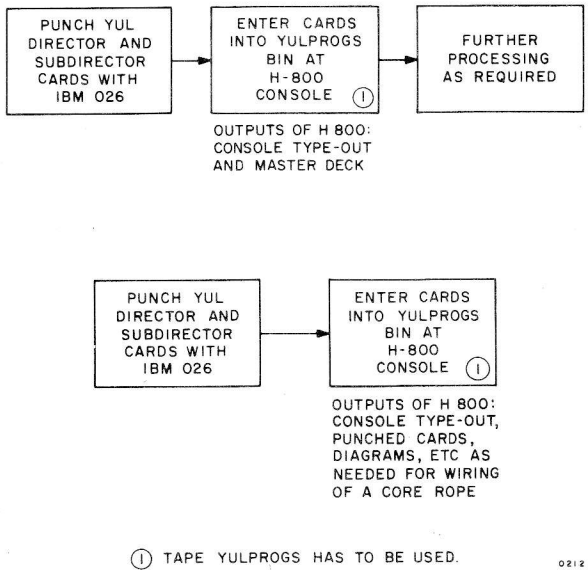


Figure 13-14. Manufacturing a Program

Figure 13-14 also shows that this Yul system is able to generate information on how the ropes are to be wired. In this case, the Yul director and subdirector cards specify what kind of information is to be generated for which program.

13-46. Table 13-1 is a summary of the various processes.

13-47. PUNCHING, PRINTING, AND READING CARDS

13-48. THE IBM-026 PRINTING CARD PUNCH

13-49. The IBM-026 Printing Card Punch is used to punch numeric, alphabetic, and special character information on 80-column cards (IBM cards). The equipment is able also to print information on an IBM card as it is punched. A program card (same type of IBM card) mounted onto the program drum of the IBM 026 is able to define fields and to control such operations as skipping, duplicating, and printing.

13-50. The maximum number of positions provided for punching holes into one card is 960: 80 columns of 12 positions or 12 rows of 80 positions. Refer to attachment 13-1. Normally, one hole, two holes, or three holes in a column represent one character. Consequently, 80 characters can be stored (and printed) on one card. The columns are numbered 1 through 80 from left to right. The first and second rows at the top of the card are called "R" and "X", respectively. The other rows are labeled "0" through "9."

13-51. Figure 13-15 shows the keyboard and the switches of the IBM-026 Printing Card Punch as used for punching cards for the Yul system. Each key is numbered for purposes of description. Keys 0 through 32 are light gray and engraved with dark blue symbols. Keys 33 through 45 are dark

TABLE 13-1
YUL PROCESSES

Process	Inputs	Outputs	Remarks
Assembly of a new program (or subroutine) Figure 13-5	Yul director (directing Yul system to assemble new program and defining new program), subdirectors as desired, and detail cards.	Console type-out and assembly listing	Tape YULPROGS stores symbolic and binary records of all AGC programs according to last assembly, revision, or deletion of a program. These seven processes were created by Hugh Blair-Smith of MIT/IL.
Assembly of a revision of a program (or subroutine) Figure 13-6	Yul director (directing Yul system to revise a program and specifying which program has to be revised), subdirectors as desired, merge control cards (specifying which parts of the program have to be revised), detail cards replacing existing cards, additional detail cards, and detail card images from tape YULPROGS.		
Assembly of a version of a program (or subroutine), old process Figure 13-7	Yul director(s) [specifying punchout of existing program(s)], punched cards of this (these) program(s), new Yul director (directing Yul system to assemble new program and defining new program), and detail cards of new program sections.		
Assembly of a version of a program (or subroutine), new process Figure 13-8	Yul director (directing Yul system to assemble a version of a program and defining the new program), subdirectors as required and desired, merge control cards, detail cards, and detail card images from tape YULPROGS. (The required subdirector specifies from which program a version is to be assembled.)		
Reprinting of a program (or subroutine) Figure 13-9	Yul director (directing Yul system to reprint a program and specifying for which program the assembly listing is to be printed) and subdirectors as desired. Do not use subdirector RENUMBER.		
Punching of a program (or subroutine) Figure 13-10	Yul director (directing Yul system to punch a Yul language card deck and specifying for which program the card deck has to be punched by the H-800 system).	Console type-out and deck of punched cards	
Deletion of an obsolete program (or subroutine) Figure 13-11	Yul director (directing Yul system to delete a program and specifying which program has to be deleted).	Console type-out	
Testing a program by simulation Figure 13-12	Yul director (directing Yul system to carry out a certain type of test and specifying which program shall be tested), subdirector (specifying which portion of a program is to be tested), and binary records of the specified program (or portions) as taken from tape YULPROGS. The above statement is true only for programs which are to be simulated by MISCGA. Mars simulation programs require different inputs.	Console type-out and simulation listing. Listing shows action of program	Presently available test program developed for AGC: MISCGA. Programs developed for Mars: Mars simulation Bench Test and Mars simulation flight test. These test functions of the Yul system were developed by Charles Muntz and Richard Warren assisted by Hugh Blair-Smith, all of MIT/IL.
Testing a program by running it on an AGC with a core rope simulator Figure 13-13	Yul director (directing Yul system to manufacture a program and specifying for which program information is to be provided) and subdirector (specifying that a load tape has to be punched). The tape, which is an output of the H-800 system, is used as an input when the core rope simulator of an AGC is being loaded.	Indication on DSKY of AGC, on display panel of computer test set, and on associated instruments	The core rope simulators are designed and built by Raytheon.
Manufacturing a program Figure 13-14	Yul director (directing Yul system to manufacture a program and specifying for which program information is to be produced), subdirectors (specifying what information is to be produced), and binary records of the specified program stored on tape YULPROGS.	Printed wiring diagrams, and/or punched cards, and/or punched tape to govern wiring of a core rope	This function has been designed by Katherine Lloyd and Hugh Blair-Smith of MIT/IL.
Maintenance of tape YULPROGS	Yul director cards referring to YULPROGS and detail cards.	Updated YULPROGS	For Yul system programs see MITIGUS—format listing of YULPASS0. The one and only one who knows all tricks: Hugh Blair-Smith.

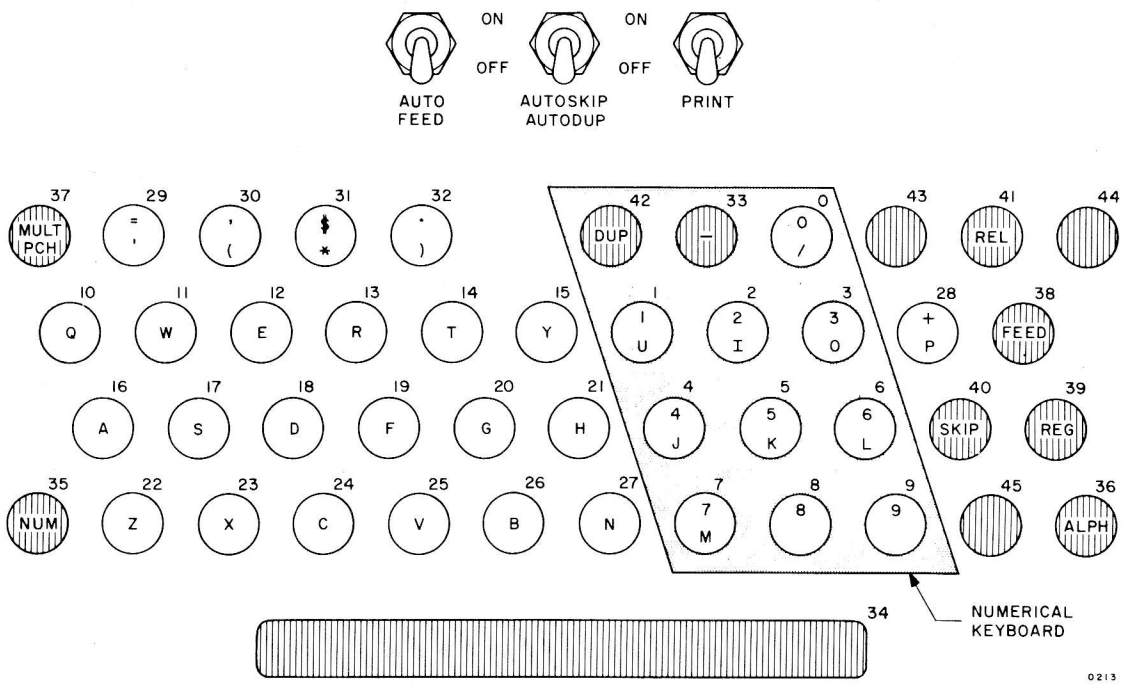


Figure 13-15. IBM-026 Printing Card Punch Keyboard and Switches

blue and carry light gray symbols. The numeric keyboard is identified by a dark blue background. The use of each key and switch is defined as follows:

Keys 0 through 9 can be depressed when the machine is in either numeric (NUM) or alphabetic (ALPH) shift.

Keys 10 through 28 can be depressed only when the machine is in shift ALPH. Otherwise the machine locks. Operation can be resumed by depressing key 36 or releasing the card.

Keys 29 through 32 can be depressed when the machine is in either NUM or ALPH shift. Key 29 might be marked - instead of ' but punches code (holes) 4, 8 in either case when in shift ALPH.

Key 33 is marked -SKIP/- but the machine at MIT/IL is wired such that it punches the code for - no matter whether in NUM or ALPH shift and never skips.

Key 34, the space bar, causes spacing over one column when the key is depressed.

Key 35, NUM (numeric shift), shifts the machine into numeric mode for the period the key is depressed. The key is normally used to punch numbers in an otherwise alphabetic field.

Key 36, ALPH (alphabetic shift), shifts the machine into alphabetic mode for the period the key is depressed. The key is normally used to punch letters in an otherwise numeric field.

Key 37, MULTPCH (multipunch), shifts the machine into numeric mode; moves the card for punching the next column and prevents further moving of the card for the period the key is depressed. The key is normally used when more than one hole has to be punched for codes which cannot be punched by depressing one of keys 0 through 34. When the key is released, the card is moved for punching another column.

Key 38, FEED (card feed), causes a card-feed cycle; feeds a card from the hopper, registers a card at the punching and reading stations, and stacks a card at the left of the machine.

Key 39, REG (register card), is used primarily when a card is inserted manually; registers a card at the punching and reading stations and stacks a card at the left of the machine.

Key 40, SKIP, causes skipping of the field for which it is depressed. It is normally used for skipping the unused right-hand portion of a field.

Key 41, REL (release), causes the cards at the punching and reading stations to be advanced completely past those stations. Fields programmed for automatic duplication beyond the point of release are punched into the card before release is completed.

Key 42, DUP (duplicate), with program control causes duplication of the field for which it is depressed at the rate of 20 columns per second. Without program control it duplicates at the rate of 10 columns per second only as long as the key is depressed.

Keys 43, 44, 45 are marked CORR, ALT PROG, and AUX DUP, but cause no effect.

Switch AUTO FEED (automatic feed). When this switch is turned ON and column 80 of the card passes the punching station, a new card is fed automatically. The card in the left of the card bed is stacked, the one in the center is registered at the reading station, and the one at the right is registered at the punching station. The automatic feeding occurs when column 80 of the card passes the punching station by any one of three possible operations: punching, skipping, or releasing.

Switch AUTO SKIP AUTO DUP (automatic skip and duplicate). When this switch is turned ON, the program punching for automatic skipping and automatic duplicating is effective. When this switch is turned OFF, commands start automatic skip (hole X in program card) and start automatic duplication (hole 0) are nullified.

Switch PRINT. When this switch is turned ON, symbols are printed in one line on top of the IBM card as their codes are punched. Which symbols are printed or suppressed is controlled by commands punched in the program card. When the switch is turned OFF, nothing is printed.

Back Space Key is located below the center of the card bed. As long as this key is depressed, the card in the bed moves to the right.

13-52. Table 13-2 is a list of the various effects of the keys of the IBM 026 when punching and printing cards for the Yul system. The table also indicates how other equipment interpret Yul cards. The first three columns of the table refer to keys being depressed. The first column shows that symbol engraved in a key which can be punched into a card if the machine is in the mode indicated in the second column. The third column identifies the keys of the first column by the numbers shown in figure 13-15. The table indicates that 57 different characters can be punched by depressing just one key. A single space between punched codes can be provided by depressing the space bar once. Another 16 characters can be punched by depressing two or three different keys sequentially if key MULTPCH is depressed during that period. The fourth column indicates at which rows of a column holes are punched into the IBM card as a consequence of actions described in the earlier columns. In the fifth column the symbols which might be printed in a card are shown. Whenever more than one symbol is shown, this means that these symbols are superimposed. Whenever a punched card is duplicated and printed at the same time, the print might vary, as indicated in the sixth column.

TABLE 13-2
IBM AND H CODES

IBM-026 Printing Card Punch as Used for Yul						Read Prints				
Key Depressed			Punched Card Code	Print	Duplication Print	IBM 407	IBM 557	H 822-3	H Console	H 800 Code
Symbol	Shift	Identification Number								
0	NUM	0	0	0	0	0	0	0	0	00
1	NUM	1	1	1	1	1	1	1	1	01
2	NUM	2	2	2	2	2	2	2	2	02
3	NUM	3	3	3	3	3	3	3	3	03
4	NUM	4	4	4	4	4	4	4	4	04
5	NUM	5	5	5	5	5	5	5	5	05
6	NUM	6	6	6	6	6	6	6	6	06
7	NUM	7	7	7	7	7	7	7	7	07
8	NUM or ALPH	8	8	8	8	8	8	8	8	10
9	NUM or ALPH	9	9	9	9	9	9	9	9	11
2, 8	MULTPCH	2, 8	2, 8	2, 8	6 ⚠1	9	2	'	'	12
=	NUM	29	3, 8	=	=	=	=	=	=	13
' or -	ALPH	29	4, 8	' or -	' or -	-	-	:	:	14
Space bar	NUM or ALPH	34	Blank	Blank	Blank	Blank	Blank	Blank	Blank	15
6, 8	NUM	6, 8	6, 8	6, 8	6 ⚠1	=	6	■ ⚠2	¢	16
7, 8	NUM	7, 8	7, 8	7, 8	Blank ⚠1	-	7	&	&	17
+	NUM	28	R	+	+	+	+	+	+	20
A	ALPH	16	R, 1	A	A	A	A	A	A	21
B	ALPH	26	R, 2	B	B	B	B	B	B	22
C	ALPH	24	R, 3	C	C	C	C	C	C	23
D	ALPH	18	R, 4	D	D	D	D	D	D	24
E	ALPH	12	R, 5	E	E	E	E	E	E	25
F	ALPH	19	R, 6	F	F	F	F	F	F	26
G	ALPH	20	R, 7	G	G	G	G	G	G	27
H	ALPH	21	R, 8	H	H	H	H	H	H	30
I	ALPH	2	R, 9	I	I	I	I	I	I	31
+, 2, 8	MULTPCH	28, 2, 8	R, 2, 8	+, 2, 8	F ⚠1	I	B	;	;	32
.	NUM	32	R, 3, 8	.	.	.	.	.	.	33
)	ALPH	32	R, 4, 8	)	)	)	)	)	)	34
+, 5, 8	MULTPCH	28, 5, 8	R, 5, 8	+, 5, 8	Blank ⚠1	)	E	%	%	35
+, 6, 8	MULTPCH	28, 6, 8	R, 6, 8	+, 6, 8	6 ⚠1	.	F	■	■	36
+, 0	MULTPCH	28, 0	R, 0	+, 0	⊄ ⚠1	0	+	■ ⚠2	Δ	37

IBM-026 Printing Card Punch as Used for Yul						Read Prints				
Key Depressed			Punched Card Code	Print	Duplication Print	IBM 407	IBM 557	H 822-3	H Console	H 800 Code
Symbol	Shift	Identification Number								
-	NUM or ALPH	33	x	-	-	-	-	-	-	40
J	ALPH	4	x, 1	J	J	J	J	J	J	41
K	ALPH	5	x, 2	K	K	K	K	K	K	42
L	ALPH	6	x, 3	L	L	L	L	L	L	43
M	ALPH	7	x, 4	M	M	M	M	M	M	44
N	ALPH	27	x, 5	N	N	N	N	N	N	45
O	ALPH	3	x, 6	O	O	O	O	O	O	46
P	ALPH	28	x, 7	P	P	P	P	P	P	47
Q	ALPH	10	x, 8	Q	Q	Q	Q	Q	Q	50
R	ALPH	13	x, 9	R	R	R	R	R	R	51
-, 2, 8	MULTPCH	33, 2, 8	x, 2, 8	-, 2, 8	0 ⚠1	R	K	#	#	52
$	NUM	31	x, 3, 8	$	$	$	$	$	$	53
*	ALPH	31	x, 4, 8	*	*	*	*	*	*	54
-, 5, 8	MULTPCH	33, 5, 8	x, 5, 8	-, 5, 8	Blank ⚠1	*	N	"	"	55
-, 6, 8	MULTPCH	33, 6, 8	x, 6, 8	-, 6, 8	6 ⚠1	$	0	■ ⚠2	∤	56
-, 0	MULTPCH	33, 0	x, 0	-, 0	⊄ ⚠1	0	-	■ ⚠2	?	57
5, 8	MULTPCH	5, 8	5, 8	5, 8	Blank ⚠1	-	5	■ ⚠2	◊	60
/	ALPH	0	0, 1	/	/	/	/	/	/	61
S	ALPH	17	0, 2	S	S	S	S	S	S	62
T	ALPH	14	0, 3	T	T	T	T	T	T	63
U	ALPH	1	0, 4	U	U	U	U	U	U	64
V	ALPH	25	0, 5	V	V	V	V	V	V	65
W	ALPH	11	0, 6	W	W	W	W	W	W	66
X	ALPH	23	0, 7	X	X	X	X	X	X	67
Y	ALPH	15	0, 8	Y	Y	Y	Y	Y	Y	70
Z	ALPH	22	0, 9	Z	Z	Z	Z	Z	Z	71
0, 2, 8	MULTPCH	0, 2, 8	0, 2, 8	0, 2, 8	W ⚠1	Z	S	@	@	72
,	NUM	30	0, 3, 8	,	,	,	,	,	,	73
(	ALPH	30	0, 4, 8	(	(	(	(	(	(	74
0, 5, 8	MULTPCH	0, 5, 8	0, 5, 8	0, 5, 8	Blank ⚠1	(	V	C_R	C_R	75
0, 6, 8	MULTPCH	0, 6, 8	0, 6, 8	0, 6, 8	6 ⚠1	,	W	■ ⚠2	□	76
0, 7, 8	MULTPCH	0, 7, 8	0, 7, 8	0, 7, 8	Blank ⚠1	(	X	■ ⚠2	⊗	77

⚠1 Not necessarily. ⚠2 Blots are forced in by the Yul system for symbols the H-822-3 system is unable to print.

13-53. In the subsequent columns of table 13-2, various printouts are listed. Column IBM 407 indicates what the IBM-407 Accounting Machine prints when Yul cards are fed into it. Column IBM 557 lists what the IBM-557 Interpreter prints on Yul cards fed into it. It is of interest to note that the machine, as presently wired, prints the characters punched in columns 1 through 60 in one line and the characters punched in columns 61 through 80 in a second line beneath the right portion of the first line. The character of column 8 is printed underneath the character of column 1. Column H 822-3 lists what the Honeywell 822-3 high-speed printer prints when Yul cards are read. Column H Console shows what the console of the Honeywell 800 system prints. The last column of table 13-2 lists the Honeywell 800 codes (6 bits) in octal numbers.

13-54. The previous paragraphs describe how a single character can be punched and how the card can be moved by the space bar. When punching in accordance with certain card formats (this is similar to typing tables on a regular typewriter), the IBM 026 can be programmed so it jumps automatically to the next field (column of a table) when the SKIP key is depressed. The machine can be programmed also to shift into ALPH mode, to skip, to duplicate, and to print automatically. The basic part of the program unit is a program card (IBM card punched for this purpose) which is mounted on the program drum. After the program control lever is moved to the left, the program punched into the program card (drum card) has control over the punching operation.

13-55. An "R" hole is punched in every column except the first (left-hand) column of each field to be skipped, duplicated, or manually punched. Figure 13-16 represents a program card used for punching Yul language cards. Holes "R" are punched in all columns except columns 1, 2, 8, 9, 18, 25, and 41. This means that the 80 columns of each card punched under program control are grouped in seven fields, as listed in table 13-3.

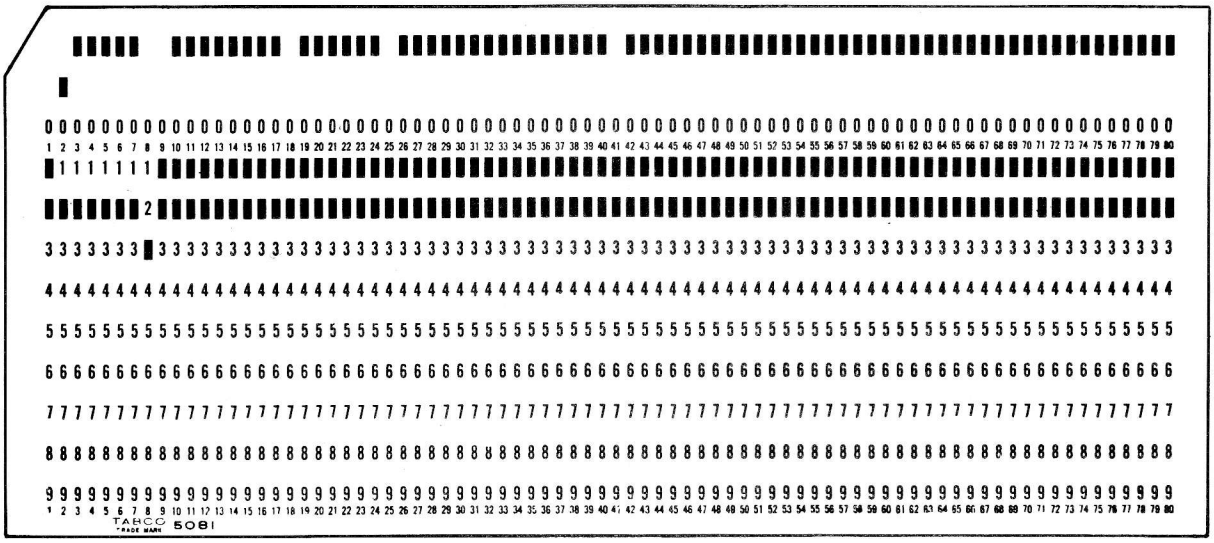


Figure 13-16. Drum Card

TABLE 13-3

THE SEVEN FIELDS OF A DRUM CARD

Name of Field	Columns
Card type	1
Card number	2 through 7
Spacing	8
Location	9 through 16 and 17*
Operation	18 through 23 and 24*
Address	25 through 40
Remark	41 through 80

*The Yul system does not recognize columns 17 and 24 as part of fields location and operation. The IBM 026 is programmed such that SKIP is always the way to advance to the fields operation and address.

13-56. An "X" hole causes the IBM 026 to skip to the first column of the next field. The "X" punched into column 2 of the Yul program card causes the punches to skip over the field card number. Normally, all numbers are punched into this field later; i.e., if a deck is complete.

13-57. A "1" hole causes the IBM 026 to punch in the ALPH mode. The Yul program card has 1's punched in columns 1 and 9 through 80, which hold the puncher in the alphabetic shift most of the time.

13-58. A "3" hole causes print suppression. The 3 punched into column 8 of the Yul program card prevents printing any character which might be punched in column 8 of any Yul card being punched. All other characters punched into a Yul card are printed when the PRINT switch is in the ON position.

13-59. With program control, zeros (not to be confused with the letter O) to the left of the first significant digit in a numeric field are automatically suppressed (i.e., 00102 is printed 102). Printing zeros to the left in a field is caused by "2" holes in the program card. The Yul program card has "2" holes punched in all columns except column 8. This has the effect that 0's are printed for all characters 0 punched in a Yul card except for the one in column 8, which is not printed anyway.

13-60. Whenever a text to be punched into the remark field needs more than 40 locations, the remaining part has to be punched into a right print remark card in columns 9 through 48. A "9" has to be punched in column 8 to cause the IBM-407 printer or the H-822-3 high-speed printer to print the characters of the right print remark card on the same line as and after the characters of the preceding Yul card.

13-61. USE OF OTHER EQUIPMENT

13-62. When the IBM-519 Document Originating Machine is used for transcribing (numbering) Yul cards, the Numbering Board must be inserted into the machine. A deck of cards to be numbered plus a card with a punch in column 2 (normally 0) as the first card are entered into the

right hopper. A Numbering Deck (cards with numbers 000 through 999 in columns 3 through 5) is put into the left hopper. If more than 999 cards are to be numbered, a higher number is punched into column 2 by using another first card (for instance 1, 2, etc).

13-63. When the IBM 519 is used for reproducing Yul cards, the Reproducing and Compare Board (with a toggle switch added by MIT) or the General Purpose Board have to be used. With the first board all information contained in cards is reproduced. With the second board cards can be reproduced without reproducing the numbers by setting the 80 toggle switches properly.

13-64. When the IBM-407 Accounting Machine is used, the Honeywell Format Board has to be inserted. (Refer to paragraph 13-23.) The 80-80 Board is a subset of the Honeywell Format Board, selected by transferring the ALTERATION 1 switch on the 407.

13-65. YUL LANGUAGE

13-66. PUNCHED CARDS USED WITH YUL SYSTEM

13-67. The 80-column cards used as input cards for the Yul system can be grouped according to their information content as follows:

- Yul Directors and Subdirectors
  - Yul directors, Type A
  - Yul directors, Type B
  - Yul directors, Type C
  - Subdirectors
- Yul Detail Cards
  - Types used for assembling or revising a program
    - Instruction cards
    - Address constant cards

Numeric constant cards
Clerical cards
Remark cards

Type used for revising a program only

Merge control cards

Types used for manufacturing a program

Master deck cards
Raytheon wiring cards, Type A (temporary).

13-68. A Yul task is performed by feeding a deck of Yul language cards (Yul deck) into the Yul system. Each Yul deck (normally supplied by a programmer) consists of one and only one Yul director, which must be the first card of the deck. Depending on the type of task, some subdirectors and detail cards might follow the Yul director. Yul director and subdirector cards specify a Yul task. Detail cards are used to feed information into the Yul system as required. The end of a Yul deck is indicated by the appearance of the Yul director of another Yul deck or by the appearance of a monitor card.

13-69. Before any Yul task can be performed, it is necessary to make the Yul system available to the H-800 system. This is done by first feeding two monitor cards (normally supplied by the operator of the H-800 system) into the card reader of the H-800 system. A job is concluded by another monitor card. All monitor cards carry the character * punched in column 1. Figure 13-17 represents an example of how a Yul job is organized. As the MIT/IL Monitor recognizes the first two monitor cards, it brings in the Yul system. The console typewriter types out the code of the job: 55 indicating NASA, 191 indicating Apollo Contract, and 23 indicating Space Guidance Analysis Group. The end of a job is signaled by monitor card ENDJOB. If the Yul system finds any card with the character * in column 1, it assumes the end of a job and does not accept any subsequent cards.

```
*   JOB  55-191-23  YUL SYSTEM
*   YUL
-------------------------------------------------
STACK OF ONE OR MORE YUL TASKS (YUL DECKS)
-------------------------------------------------
*   ENDJOB                                  0411
```

Figure 13-17. Yul Job

13-70. YUL DIRECTORS AND SUBDIRECTORS

13-71. Yul directors, Type A, are needed to perform maintenance functions on the Yul system. Refer to the last row of table 13-1. These directors are not of interest to an AGC programmer, for which reason this type of card is not further discussed. Yul directors, Type B, request an action on an AGC program (or subroutine). Yul directors, Type C, are Yul directors which are not of Type A or B. The director MESSAGE is of interest to AGC programmers. The MESSAGE cards (character Y in column 1, word MESSAGE in columns 9 through 80 followed by a message) can be used by AGC programmers to "talk" to the H-800 operator. The content of any MESSAGE card is printed out only by the H-800 system and has no effect on the Yul or H-800 system. The message might be USE 3-PLY PAPER IF POSSIBLE, or PLEASE CALL ME BEFORE RUN, etc. The MESSAGE card is not part of any Yul programming task; it represents an independent task.

13-72. Yul directors, Type B, request an action on a particular AGC program (or subroutine). Yul subdirectors give further detail in regard to an action or specify a particular part of a program. Refer to other rows of table 13-1. The format of Yul directors B and subdirectors is indicated in figure 13-18. Yul directors are identified by a Y punched in column 1, subdirectors by an S. The contents of columns 2 through 8 are ignored by the Yul system. Columns 9 through 80 contain the directive in

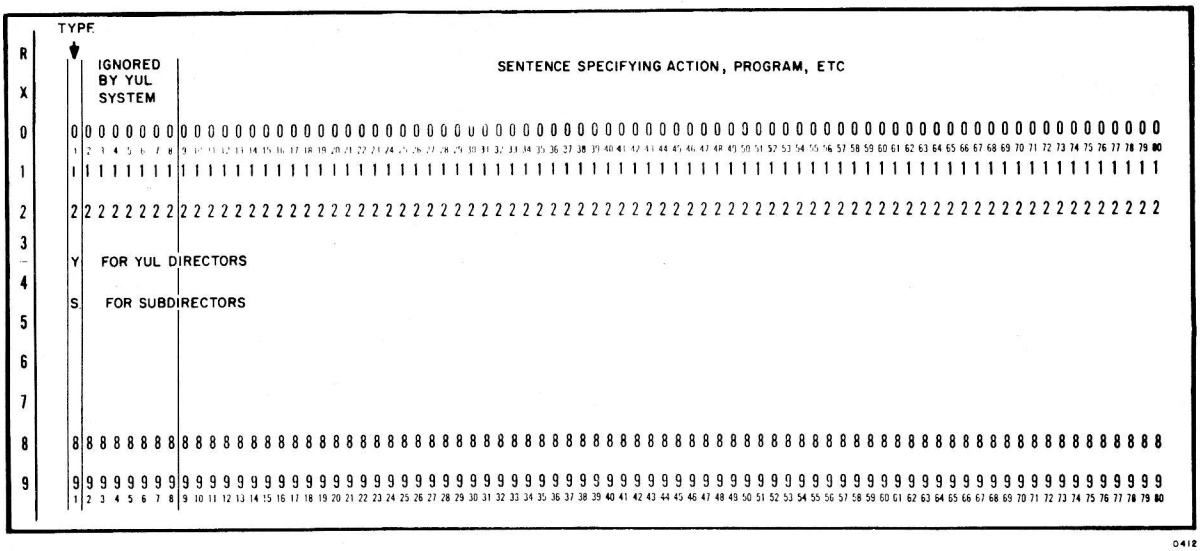


Figure 13-18. Yul Directors (B) and Subdirectors

the form of a Yul sentence. The contents of directors and subdirectors are not printed out as part of an assembly listing.

13-73. A Yul sentence is defined as a combination of words, and a word is defined as a combination of characters. The particularities of sentences in the Yul language are explained in figure 13-19 by means of a metalanguage (language about a language; see "Report on the Algorithmic Language ALGOL 60," by Peter Nauer, Vol. 3, Issue 3, of "Communications of the Association for Computing Machinery," March 1963). Definitions 1 and 2 define which characters of the Yul system are called nonterminating characters and which are called nonblank terminators. Definition 3 indicates that a terminating character might be a nonblank terminator or a blank. According to definition 4, a terminator can be a terminating character or a blank followed by another terminator. Definition 5 says that a nonblank character can be a nonblank terminator or a nonterminating character.

13-74. A word is defined as being one nonterminating character, or

C ≡ CHARACTER
TC ≡ TERMINATING CHARACTER
NTC ≡ NONTERMINATING CHARACTER
T ≡ TERMINATOR
NBT ≡ NONBLANK TERMINATOR
NBC ≡ NONBLANK CHARACTER
b ≡ BLANK

DEFINITION 1 <NTC> :: = 0|1|2|3|4|5|6|7|8|9|'|&|A|B|C|D|E|F|G|H|I|
;|.|)|%|■|J|K|L|M|N|O|P|Q|R|≠|$|"|S|T|U|V|
W|X|Y|Z|@|(|C_R|

DEFINITION 2 <NBT> :: = =|:|+|-|*|/|,

DEFINITION 3 <TC> :: = <NBT> | b

DEFINITION 4 <T> :: = <TC> | b <T>

DEFINITION 5 <NBC> :: = <NBT> | <NTC>

DEFINITION 6 <WORD> :: = <NTC> | <WORD> <NTC> | b <WORD>

MAXIMUM LENGTH OF A WORD IS 16 <NTC>.

DEFINITION 7 <SENTENCE> :: = <WORD> | <SENTENCE> <T> <WORD> | <SENTENCE> b

MAXIMUM LENGTH OF A SENTENCE IS 13 <WORDS>.

<SENTENCE> MUST FIT INTO COLUMNS 9 THROUGH 80

AN ≡ AUTHOR'S NAME AFTER PROCESSING

8 <AN> :: = <WORD> | <AN> <TC> <WORD>

MAXIMUM LENGTH OF <AN>, FIRST <NTC> TO LAST <NTC>, IS 16 C.
UNNECESSARY BLANKS ARE DROPPED.

EXAMPLE: DR.bbTERRYbbb-bbbbTHOMAS BECOMES DR.bTERRY-THOMAS
<TC> <TC>

0457

Figure 13-19. Yul Sentence

another word (one or more nonterminating characters) followed by a nonterminating character, or a blank followed by another word. The word of a Yul director or subdirector card shall never be longer than 16 nonterminating characters. A sentence is defined as consisting of one word, or another sentence (one or more words) followed by a terminator and a word, or a sentence followed by a blank. Note that a blank can be considered as a part of a sentence, a word, or a terminator. The sentence of a Yul director or subdirector card must fit into columns 9 through 80 and shall never consist of more than 13 words. Definition 8 indicates the composition of a name.

13-75. The sentence punched in a Yul director type B is an imperative sentence which is normally constructed as follows:

> verb / adjective / computer name / PROGRAM (or SUBROUTINE) / program (or subroutine) name / BY / author's name.

In case of version assembly, the imperative sentence is punched in a Yul director B and a subdirector:

> ASSEMBLE / VERSION / name of new program (or subroutine) / BY / name of author of new program (or subroutine) // FROM / adjective / computer name / PROGRAM (or SUBROUTINE) / name of existing program (or subroutine) / BY / name of author of existing program.

(The word SUBROUTINE as well as the word PROGRAM will be recognized.)

13-76. A sentence can be composed of the following words:

Verbs	ASSEMBLE
	REPRINT
	PUNCH
	DELETE
	ETALIMUS
	MANUFACTURE

Adjectives	NEW REVISION N OF ($0 \leq N \leq 999$; see also paragraph 13-78) VERSION
Computer names	MOD 3C or 3C AGC4 or MOD AGC4 (See also paragraph 13-79)
Program (or subroutine) names, or noun	For instance, BUGFULL
Author's name	For instance, DR. TERRY-THOMAS (See also figure 13-19, definition 8)

13-77. Some examples of sentences are:

a. ASSEMBLE NEW MOD 3C PROGRAM BUGFULL BY DR. TERRY-THOMAS
b. ASSEMBLE REVISION 15 OF AGC4 PROGRAM HIKE BY JOVE
c. ASSEMBLE VERSION WIENER BY NORBERT
FROM REVISION 46 OF AGC4 PROGRAM HIKE BY JOVE
d. REPRINT REVISION 46 OF AGC4 PROGRAM HIKE BY JOVE
e. PUNCH NEW AGC4 PROGRAM BIMBO BY LULU
f. DELETE REVISION 0 OF 3C PROGRAM BUGFULL BY DR. TERRY - THOMAS
g. ETALIMUS REVISION 2 OF AGC4 PROGRAM BUGS BY BUNNY
h. MANUFACTURE REVISION 11 OF AGC4 PROGRAM ZAP BY JFB

13-78. The expression REVISION 0 OF has the same meaning and effect on the Yul system as the word NEW, except when used with the word ASSEMBLE. With the word ASSEMBLE only the word NEW can be used; the use of expression REVISION 0 OF is illegal. Whenever a new or previously revised program has to be revised, the number of the new revision has to be inserted. For instance, if the last revision was the fourteenth, then it has to say REVISION 15 OF.

13-79. The word MOD may optionally precede any computer name in a Yul director or subdirector. The Yul system eliminates the word MOD

during the card-analyzing routine and stores the name without MOD. When a computer name so stored is printed or punched by the Yul system and if the name starts with a numeral, the name is preceded by MOD.

13-80. ASSEMBLY OF A NEW PROGRAM (OR SUBROUTINE)

13-81. Let us consider the example

ASSEMBLE NEW MOD 3C PROGRAM BUGFULL BY DR. TERRY-THOMAS

As the Yul system reads the Yul director B, it goes through the following operations:

Step 1. The Yul system recognizes the word ASSEMBLE and prepares itself to assemble a program (or subroutine).

Step 2. It recognizes the word NEW and prepares itself to introduce a new program (or subroutine).

Step 3. It recognizes the code MOD 3C, ignores the word MOD, and verifies that 3C is a computer name known to the Yul system. It verifies also that a program can be assembled for this computer.

Step 4. It recognizes the word PROGRAM, prepares itself to assemble a new program, and provides revision number 0 for this program.

Step 5. It recognizes the name BUGFULL and verifies that this name is not in use for any 3C program (or subroutine).

Step 6. It recognizes the word BY.

Step 7. It recognizes the name DR. TERRY-THOMAS, puts the name in standard format (see figure 13-19), associates the name with program BUGFULL, and checks whether the author's name is known to the Yul system; if not, the name is added to the author list.

Step 8. It assembles the new program and stores it on tape YULPROGS.

13-82. REVISION OF A PROGRAM (OR SUBROUTINE)

13-83. Let us consider the example

ASSEMBLE REVISION 15 OF AGC4 PROGRAM HIKE BY JOVE

Step 1. The Yul system recognizes the word ASSEMBLE and prepares itself to assemble a program (or subroutine).

Step 2. It recognizes the word REVISION and prepares itself to revise an existing program (or subroutine).

Step 2A. It recognizes the expression 15 OF and prepares itself to check on the revision number.

Step 3. It recognizes the word AGC4 and verifies that AGC4 is a computer name known to the Yul system. It verifies also that a program can be assembled for this computer.

Step 4. It recognizes the word PROGRAM and prepares itself to revise an existing program.

Step 5. It recognizes the name HIKE, verifies that this program does exist for the AGC4, verifies that its last known revision number is 14, and replaces the 14 by a 15.

Step 6. It recognizes the word BY.

Step 7. It recognizes the name JOVE and checks that this name is the name of the author who originated that program.

Step 8. It assembles a revised program by merging the unchanged parts of the program (taken from SYPT or SYLT) with the new parts of the program (on cards) under control of merge control cards. (Paragraphs 13-184 through 13-194 explain the use of merge control cards.) During revision of a program the unrevised program is deleted from tape YULPROGS and replaced by the revised program, unless a merging error was detected. In this case, the revision number is restored to its original value and the content of tape YULPROGS is kept as if the revision had never begun. The assembly is said to be rejected.

13-84. VERSION ASSEMBLY OF A PROGRAM (OR SUBROUTINE)

13-85. Let us consider the example

ASSEMBLE VERSION WIENER BY NORBERT
FROM REVISION 46 OF AGC4 PROGRAM HIKE BY JOVE

Step 1. The Yul system recognizes the word ASSEMBLE and prepares itself to assemble a program (or subroutine).

Step 2. It recognizes the word VERSION and prepares itself to assemble a version of the program (or subroutine) defined by the first subdirector.

Step 3. It takes WIENER, the name of the new program (or subroutine), and puts it aside.

Step 4. It recognizes the word BY.

Step 5. It takes the new author's name, NORBERT (puts it in standard form when necessary), and puts it aside.

Step 6. It recognizes the word FROM as it starts reading the first subdirector.

Step 7. It recognizes the expression REVISION 46 OF and prepares itself to check on the revision number.

Step 8. It recognizes the name AGC4 and verifies that AGC4 is a computer name known to the Yul system. It verifies also that a program can be assembled for this computer.

Step 9. It recognizes the word PROGRAM and prepares itself to assemble a new program.

Step 10. It recognizes HIKE, the name of the existing program, verifies that this program does exist for AGC4, and verifies that the last known revision number is 46.

Step 11. It recognizes the word BY.

Step 12. It recognizes JOVE, the name of the author of the existing program, and checks that this name is the name of the author who originated program HIKE.

Step 13. It takes the name WIENER, put aside during Step 3, and verifies that this name is not in use for any AGC4 program (or subroutine).

Step 14. It takes the name NORBERT, put aside during Step 5, associates the name with program WIENER, and checks whether the author's name is known to the Yul system; if not, the name is added to the author list.

Step 15. It takes Yul card images of the specified existing program from tape YULPROGS (without disturbing it), merges them with information brought in on cards under the control of merge control cards, and assembles program WIENER.

13-86. REPRINTING OF A PROGRAM (OR SUBROUTINE)

13-87. Let us consider the example

REPRINT REVISION 46 OF AGC4 PROGRAM HIKE BY JOVE

Step 1. The Yul system recognizes the word REPRINT and prepares itself to print an assembly listing for a program (or subroutine) stored on tape YULPROGS.

Step 2. It recognizes the expression REVISION 46 OF and prepares itself to check on the revision number.

Step 3. It recognizes the code AGC4 and verifies that AGC4 is a computer name known to the Yul system.

Step 4. It recognizes the word PROGRAM and prepares itself to print an assembly listing for an existing program.

Step 5. It recognizes the name HIKE, verifies that this program does exist for AGC4, and verifies that the last known revision number is 46.

Step 6. It recognizes the word BY.

Step 7. It recognizes the name JOVE and checks that this name is the name of the author who originated the program.

Step 8. It punches a copy of the assembly listing for program HIKE from tape YULPROGS.

13-88. PUNCHING A PROGRAM (OR SUBROUTINE)

13-89. The sentence to be punched on a Yul director for punching a

program is the same as for reprinting a program, except for the first word. A completely different sentence was chosen as an example to demonstrate the meaning of NEW. Let us consider the example

PUNCH NEW AGC4 PROGRAM BIMBO BY LULU

Step 1. The Yul system recognizes the word PUNCH and prepares itself to punch a deck of Yul language cards for a program (or subroutine) stored on tape YULPROGS.

Step 2. It recognizes the word NEW and prepares itself to check on the revision number (in this case 0).

Step 3. It recognizes the code AGC4 and verifies that AGC4 is a computer name known to the Yul system.

Step 4. It recognizes the word PROGRAM and prepares itself to punch a card deck for an existing program.

Step 5. It recognizes the name BIMBO, verifies that this program does exist for the AGC4, and verifies that the last known revision number is 0.

Step 6. It recognizes the word BY.

Step 7. It recognizes the name LULU and checks that this name is the name of the author who originated the program.

Step 8. It punches a copy of the Yul director and Yul language card deck for AGC4 program BIMBO.

13-90. DELETION OF A PROGRAM (OR SUBROUTINE)

13-91. Let us consider the example

DELETE REVISION 0 OF 3C PROGRAM BUGFULL BY DR. TERRY - THOMAS

Step 1. The Yul system recognizes the word DELETE and prepares itself to delete a program (or subroutine) from tape YULPROGS.

Step 2. It recognizes the expression REVISION 0 OF (the word NEW could have been used instead) and prepares itself to check on the revision number.

Step 3. It recognizes the code 3C and verifies that 3C is a computer name known to the Yul system.

Step 4. It recognizes the word PROGRAM and prepares itself to delete an existing program.

Step 5. It recognizes the name BUGFULL, verifies that this program does exist for the 3C, and verifies that the last known revision number is 0.

Step 6. It recognizes the word BY.

Step 7. Similar to step 7 of paragraph 13-81.

Step 8. It deletes program BUGFULL from tape YULPROGS.

13-92. TESTING A PROGRAM BY MISCGA

13-93. Let us consider the example

ETALIMUS REVISION 2 OF AGC4 PROGRAM BUGS BY BUNNY

Step 1. The Yul system recognizes the word ETALIMUS ("simulate", backwards) and prepares itself to test a program by means of program MISCGA. (Subroutines cannot be simulated or manufactured.)

Step 2. It recognizes the expression REVISION 2 OF and prepares itself to check on the revision number.

Step 3. It recognizes code AGC4 and verifies that AGC4 is a computer name known to the Yul system.

Step 4. It recognizes the word PROGRAM.

Step 5. It recognizes the name BUGS, verifies that this program does exist for AGC4, and verifies that the last known revision number is 2.

Step 6. It recognizes the word BY.

Step 7. It recognizes the name BUNNY and checks that this name is the name of the author who originated the program.

Step 8. It verifies that a binary record exists for the program (which is true only if the last assembly was good).

Step 9. The test is carried out with the help of one or more

subdirectors which might contain the following sentences:

START AT 2537	Necessary card, always first subdirector.
PRINT LIMIT 19000	Optional, always second card if present; if not provided, no more than 5000 lines are printed out.

The four following card types have the purpose of inhibiting detail printouts for selected sections. None or as many as desired can be used. If no card of this group is used, everything is traced (i.e., each instruction simulated shows a line or print giving the state after execution).

TRACE OFF 2777	Turns trace off just after execution of an instruction at octal location 2777.
TRACE ON 2713	Turns trace on just before execution of an instruction at octal location 2713.
TRACE OFF R:0110	Turns trace off just after execution of an instruction that refers to octal location 0110.
TRACE ON R:0113	Turns trace on just before execution of an instruction that refers to octal location 0113.

The last 300 instructions simulated in a run are always traced and are tagged CORONER'S REPORT. This feature overrides any trace order or print limit given. (During a test the execution results of the last 300 instructions are stored in the Yul system.)

13-94. MANUFACTURING A PROGRAM

13-95. Let us consider

MANUFACTURE REVISION 11 OF AGC4 PROGRAM ZAP BY JFB

Step 1. The Yul system recognizes the word MANUFACTURE and prepares itself to generate production information for a program.

Step 2. It recognizes the expression REVISION 11 OF and prepares itself to check on the revision number.

Step 3. It recognizes code AGC4 and verifies that AGC4 is a computer name known to the Yul system.

Step 4. It recognizes the word PROGRAM and prepares itself to generate production information for a program.

Step 5. It recognizes the name ZAP, verifies that this program exists for AGC4, and verifies that the last known revision number is 11.

Step 6. It recognizes the word BY.

Step 7. It recognizes the name JFB and checks that this name is the name of the author who originated the program.

Step 8. It verifies that a binary record exists for the program (which is true only if assembly was good).

Step 9. Production information is generated with the help of one or more subdirectors. At present four different subdirectors are provided:

PUNCH MASTER DECK: The only card of the Yul task card deck besides the Yul director.

PUNCH CORE ROPE SIMULATOR LOAD TAPE: The only card besides the Yul director.

PUNCH RAYTHEON WIRING CARDS: Raytheon wiring cards, Type A, are required to define the Type B cards to be punched.

PUNCH RAYTHEON WIRING TAPE: Requires further subdirectors defining paragraphs or substrands for which tape has to be generated.

For instance, the sentence PUNCH MASTER DECK punched on a subdirector is recognized by the Yul system and causes it to punch a master deck of the specified program.

13-96. SUBDIRECTORS

13-97. The words punched in a subdirector might be part of a sentence, a complete sentence, or any imperative statement. Some subdirectors can be used with a certain Yul director B only, others with various Yul directors.

13-98. The sentences

PRINT 13 COPIES

and/or

PRINT 36 LINES PER PAGE

punched in a subdirector can be used with Yul directors

ASSEMBLE NEW (paragraph 13-80)
ASSEMBLE REVISION (paragraph 13-82)
ASSEMBLE VERSION (paragraph 13-84), and
REPRINT (paragraph 13-86).

Normally, one copy (plus carbon copies) of a Yul listing is printed during assembly. Whenever subdirector PRINT nn COPIES follows a Yul director B listed above and if the assembly was judged "good" or "fair", the Yul system prints nn copies ($2 \leq nn \leq 63$), including all carbon copies. Normally, the Yul system prints 54 lines in each sheet, including headlines and skipped lines. (The distance between the first and last line is 9 inches.) Whenever subdirector PRINT nn LINES PER PAGE or PRINT nn LINES follows a Yul director B listed above, the Yul system prints nn lines per page ($10 \leq nn \leq 63$). Both subdirectors PRINT must follow immediately the director ASSEMBLE NEW, the director ASSEMBLE, the director REPRINT, or the subdirector FROM of director ASSEMBLE version. When both subdirectors are used with a Yul director, either can be the first of the two.

13-99. The word RENUMBER punched on a subdirector is normally used with Yul director ASSEMBLE REVISION or ASSEMBLE VERSION in order to renumber the sequence of detail cards. Whenever the card RENUMBER is used, the assembly listing does not show the card numbers punched originally on the cards (except merge control cards) but new numbers. Also, if a merge error is discovered, renumbering stops immediately. The old card numbers on tape YULPROGS are replaced (after merging) by new numbers and are used from there on. In case of emergency (for instance IBM 519 out of order), subdirector RENUMBER can be used also with Yul director ASSEMBLE NEW. In this case the Yul system ignores the card numbers punched on detail cards or missing from them and inserts card numbers into tape YULPROGS and the assembly listing. Whenever a log card appears during operation RENUMBER, a new numbering sequence is started with 0001.

13-100. Subdirector FROM must always follow the Yul director ASSEMBLE VERSION immediately and was discussed in paragraph 13-84.

13-101. Subdirectors START AT, PRINT LIMIT, and TRACE are used with Yul director ETALIMUS and were discussed in paragraph 13-92.

13-102. Various subdirectors PUNCH are used with Yul director MANUFACTURE and were mentioned in paragraph 13-95.

13-103. YUL DETAIL CARDS

13-104. Detail cards are used to feed information into the Yul system. Detail cards are all cards which do not carry the character * or Y or S in column 1. Of the detail cards used for assembly of a program (or subroutine), some types have an identification character in column 1 (various remark cards and one type of merge control card); one type has an identification

character in column 8 (right-print remark cards). All other types are identified by operation codes in columns 18 through 23. An operation code can be an order code (instruction cards), a constant code (address constant cards and numeric constant cards), a clerical code (clerical cards), or a merge control code (merge control cards). Of the detail cards used for manufacturing a program, no identification codes are needed, because only one type of detail card is required for each task.

13-105. Information contained in column 1 of a Yul language card (attachment 13-1) normally identifies the card type, as listed in table 13-4.

TABLE 13-4

MEANING OF COLUMN 1

Content of Column 1	Identifies	
*	Monitor card	
Y	Yul director	
S	Subdirector	
blank	Instruction card	Five-field detail cards
blank	Address constant card	
blank	Numeric constant card	
J	Leftover instruction or leftover constant card	
blank	Clerical card	
blank	Merge control card	
=	Acceptor, a type of merge control card	Two-field detail cards
T	Tins card, a type of merge control card	
L	Log card	
R	General remark card	
blank	Right-print remark card (identified by character 9 in column 8)	
A	Aligned remark card	
P	Page remark card	

13-106. Detail cards containing program information are divided into five fields, as indicated in figure 13-20:

Card number field	Columns 2 through 7
Location field	Columns 9 through 16
Operation field	Columns 18 through 23
Address field	Columns 25 through 40
Remark field	Columns 41 through 80

The location, operation, and address fields contain program information. Numbers defining the sequence of cards within a Yul deck are punched into the first field. Any remarks, if needed, are punched into the last field. Detail cards containing no program information but remarks only are divided into two fields, as indicated in figure 13-21:

Card number field	Columns 2 through 7
Remark field	Columns 9 through 80

13-107. The character punched in column 1 of a Yul language card also determines how the content of that card is printed out by the IBM 407 (attachment 13-3) and by the Yul system (sheet 1 of attachment 13-4). Only the digit part (rows "0" through "9" but not rows "R" and "X", see paragraph 13-50) of column 1 is examined by the IBM 407, but the mnemonic letters listed in table 13-4 must be used if the card is to be processed by the Yul system.

13-108. The standard format for the printout of the IBM 407 with the Honeywell Format Board inserted is as follows:

Columns 1 through 7	Printed in print positions 1 through 7
Column 8	Not printed
Columns 9 through 80	Printed in print positions 9 through 80

The IBM 407 prints out the content of a card according to the standard format if column 1 is blank or if column 1 contains any character listed in table 13-4 except L, T, and =. When a card is processed by the Yul system and its content is printed out by the H-822-3 High Speed Printer and

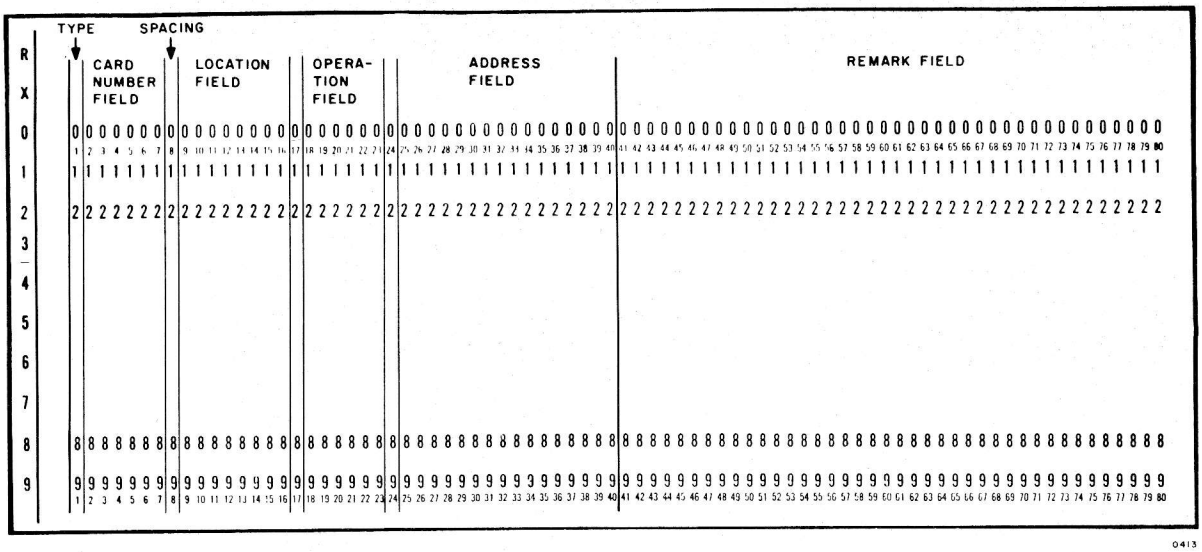


Figure 13-20. Yul Detail Card with Five Fields

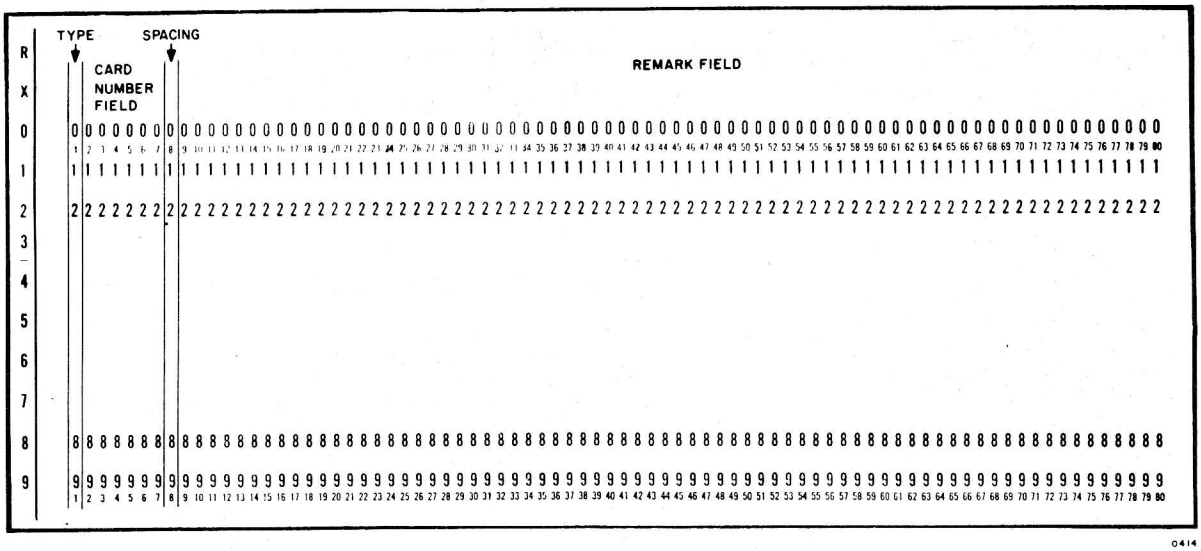


Figure 13-21. Yul Detail Card with Two Fields

if column 1 contains the character R or P, the content of the card is printed out also according to the standard format. The printout of cards with characters L or T in column 1 will be described when these types of detail cards are discussed. If column 1 is blank or contains the character A or J, the contents of columns 1 through 7 are printed in print positions 1 through 7, and the contents of columns 9 through 80 in print positions 49 through 120. Refer to sheet 1 of attachment 13-4. Octal translations and complaints are printed into print positions 9 through 47.

13-109. The contents of all detail cards except the contents of merge control cards are written into SYPT (or SYLT). The contents of two-field detail cards have no effect on a program; these cards affect only the printout of an assembly listing. Five-field cards are recognized by the Yul system as containing program information.

13-110. The information contained in column 8 of a detail card normally refers to upspacing: it is the space between the printout of that card and the next line. Information in column 8 might have a different meaning for the IBM 407 or the Yul system, as indicated in table 13-5.

13-111. As indicated earlier, detail cards may contain card numbers in columns 2 through 7. When a new card deck is punched, card numbers do not have to be punched by hand; they can be inserted by the IBM-519 Document-Originating Machine. Whenever cards are revised or added, the card numbers have to be punched on the IBM 026. All card numbers except the one of right-print remark cards following a general remark (R) or a page remark card (P) are printed into an assembly listing. The card number of a right-print remark card following a five-field detail card is preceded by the character C when printed out. This indicates that the content of the right-print remark card occupies a new line, as can be seen from sheet 1 of attachment 13-4. The characters 0 and blank are equivalent

TABLE 13-5

MEANING OF COLUMN 8

Content of Column 8	Yul System	IBM 407
Blank	Normal upspace	
0	Suppress upspace	
1	Normal upspace	
2	Double upspace	
3	Triple upspace	Double upspace
4	Quadruple upspace	
5	Quintuple upspace	Quadruple upspace
6	Sextuple upspace	Quadruple upspace
7	Septuple upspace	Quadruple upspace
8	Skip to head of next page	
9	Identifies right-print remark card	

in a card number. The character 0 or a blank in columns 2 through 5 is always printed as a 0. A 0 is printed in position 6 only if any digit between 1 and 9 is present in position 7. The character 0 is never printed in position 7.

13-112. The numbers punched for a new card deck are punched into columns 2 through 5, starting with 1 or any number larger than 0. Each subsequent card must have a number greater (any amount) than the number of the preceding card unless the number sequence is broken. Normally, a log card is used to break a sequence and to specify the start of a new sequence. Another way to break a sequence is to punch one or more nonnumericals into columns 2 through 7 or the number 999999. The largest number that can be punched into columns 2 through 5 is 9999. When new cards which are supposed to fall between existing cards (for example cards 0006 and 0007 of attachment 13-4) have to be added, the numbers 1, 2, 3, etc are punched in addition into position 6. In a similar way numbers can be punched into position 7 when it is necessary to number another set of additional cards.

13-113. The location and operation fields of a detail card (figure 13-20) consist of one subfield each. The address field normally consists of one or two subfields but may also contain special formats in certain cases. The fundamental units of meaning (words) to be used in the location, operation, and address fields are the subfields. A subfield is a word of a certain length and composition. Four types of subfield are provided:

Blank subfield
Numeric subfield
Signed numeric subfield, and
Symbolic subfield.

13-114. A blank subfield consists entirely of blanks and is therefore detectable only when it is alone in a field. A numeric subfield consists of numerals, optionally followed by the character D. Leading, embedded, and trailing blanks of numeric and signed numeric subfields are ignored by the Yul system during assembly. A numeric subfield may not contain more than eight nonblank characters. This type of subfield is interpreted as an octal integer unless the D is included, in which case the subfield is interpreted as a decimal integer. If a numeric subfield contains character 8 or 9 but no D, it is considered to represent a decimal integer, but a complaint is printed. A signed numeric subfield has either the symbol + or - as its leftmost nonblank character; otherwise it does not differ from a numeric subfield. A symbolic subfield is any other collection of characters which has not more than eight characters (nonblank characters included) counted from its leftmost to its rightmost nonblank character. Leading blanks are ignored, but embedded blanks are significant (paragraph 13-131), and some trailing blanks can be significant if a subfield consists of less than eight characters (eight as defined before). For this reason it is suggested not to use symbolic subfields with embedded blanks.

13-115. INSTRUCTION CARDS

13-116. An instruction card is a five-field detail card which contains the order code of an instruction in the operation field. The card may contain in the location, operation, and address fields the information listed in table 13-6.

TABLE 13-6

CONTENTS OF INSTRUCTION CARDS

Location Field	Operation Field	Address Field
Blank subfield ( )	Symbolic subfield (CAF)	Blank subfield ( )
Symbolic subfield (WERNER)	Numeric subfield (4)	Symbolic subfield (ONE)
Numeric subfield (5510)		Numeric subfield (5776)
Signed numeric subfield (-1)		Signed numeric subfield (-1)
		Symbolic and signed numeric subfield (BRAUN +1)
		Numeric and signed numeric subfield (5521 +1)
		Signed numeric and signed numeric subfield (+1 -2)

13-117. The location field of five-field detail cards is examined during Pass 1, as stated in paragraph 13-24. The location field usually contains a symbolic subfield or is blank (sheet 1 of attachment 13-4). It may also contain a signed numeric subfield, which is ignored by the Yul system. (For instance, the location field of line 0016 could contain the expression

-1, used in the address field of the next line for the convenience of the human reader.) A numeric subfield overrides the setting of the location counter of the Yul system and resets it to the quantity represented. (Replacing the word WERNER by 6510 would set the location counter in a similar way as the clerical code SETLOC 3400D would do it. The octal address 6510 is equal to 3400D.) If the location field is blank or contains a signed numeric, the instruction word contained in the operation field and the address field is assigned to the location defined by the content of the location counter. If the location field contains a symbol, the same is true (unless the symbol has been defined by an EQUALS clerical code), and the symbol becomes defined as equivalent to the content of the location counter. If the symbol has been defined by an EQUALS card earlier in the card deck, this has the same effect as replacing the symbol with a numeric subfield.

13-118. Instruction cards may contain Regular Machine Instructions (table 1-1) or Interpretive Instructions (tables 1-2 and 6-1). The order code of a Regular Instruction is punched into the operation field, and its relevant address is punched into the address field. The first order code of an Interpretive Instruction Word is punched into the operation field. The second order code, or number identifying how many IIW's immediately follow the first IIW (see Issue 2), is punched into the address field. All information of an instruction card except the content of column 8 is printed into an assembly listing. As the Yul system recognizes an order code, the code is translated into a binary equivalent as part of an instruction word (normally stored on tape YULPROGS) and an octal equivalent printed into an assembly listing.

13-119. The following order codes of Basic Instructions can be punched into an operation field of an instruction card:

TC or 0 (zero)

TCR	TC intending to return; recognized by Yul system as identical to TC; used for the benefit of human readers.
CCS or 1	The Yul system checks whether its relevant address refers to F memory or not. In case it does refer to F memory, the Yul system complains. No complaint is made if an instruction is indexed.
INDEX, NDX or 2	-
XCH, CAF or 3	For CAF refer to paragraph 2-20. The Yul system complains if CAF does not refer to a location in F memory. No complaint is made if an instruction is indexed.
CS or 4	-
TS or 5	-
AD or 6	-
MASK, MSK or 7	-

13-120. The following order codes of Extra Code Instructions can be punched into an operation field, but these cards have to be preceded by a card with instruction EXTEND or INDEX 5777 (NDX 5777). Location 5777 contains the quantity 47777. (Refer to paragraph 2-49.)

MP or 4
DV or 5
SU or 6

13-121. The order codes listed in paragraphs 13-119 and 13-120 are followed by meaningful expressions in the address field. The following Implied Address Order Codes can be used as well. The address field of an Implied Address Order Code Instruction is not used by the Yul system. The Yul system knows the relevant addresses of these instructions and supplies them automatically.

XAQ	(= TC A)	Refer to paragraph 2-82.
RETURN	(= TC Q)	
NOOP	(= XCH A)	Refer to paragraph 2-83.
COM	(= CS A)	Refer to paragraph 2-84.
OVSK	(= TS A)	Refer to paragraph 2-85.
TCAA	(= TS Z)	
DOUBLE	(= AD A)	Refer to paragraph 2-87.
EXTEND	(= NDX 5777)	Refer to paragraph 2-88.
RESUME	(= NDX 0025)	
INHINT	(= NDX 0017)	
RELINT	(= NDX 0016)	
SQUARE	(= MP A)	Has to be preceded by EXTEND. Refer to paragraph 2-92.

13-122. The order code

OVIND (= TS blank) can be used when an overflow indication is desired without disturbing E memory. As the Yul system recognizes the word OVIND, it inserts the code 5 (for TS) as the order code and the location number of OVIND as the relevant address. Instruction OVIND shall be written in F memory only; it shall never appear in E memory.

13-123. The character - punched into column 17 of an instruction card causes the Yul system to complement the translated binary and octal equivalents of the instruction word. For instance, instruction -CCS 0136 becomes - 1 0136 as a result of translation and 67641 as a result of complementation. The character - punched in column 17 (and stored in SYPT or SYLT) flags the instruction word (in BYPT) as a constant for simulation purposes. The simulator has the additional feature of being able to distinguish between an instruction and a constant and executes instructions only. In this case, for instance, the simulator refuses to execute 67641 as AD 7641.

13-124. Order codes of Interpretive Instructions are punched as shown in tables 1-2 and 6-1. In the case of indexing operations, the rightmost

nonblank character may be an *. (Refer to paragraph 6-21.) No other codes are recognized by the Yul system or the Interpreter. As the Yul system translates the symbolic order codes into binary and octal equivalents, it increments (by one) and complements the Interpretive Instruction Word (paragraph 6-29). For instance, the word DAD DMP is translated to 034 and 054 (refer to table 1-2), or 000111000101100. After incrementing and complementing, 111000111010010 is stored in the Yul system, and 70722 is printed in the program or subroutine listing.

13-125. Various examples of codes which might be punched into the address field of an instruction card appear on sheet 1 of attachment 13-4. Whenever the operation field contains an Implied Address Order Code (paragraph 13-99), the address field is neglected by the Yul system. Whenever the operation field contains an Interpretive Instruction Code, the address field might also contain an Interpretive Instruction Code, or a number, or it might be blank.

13-126. Whenever the operation field contains the order code of a Basic Machine Instruction (paragraph 13-119) or the order code of an Extra Code Instruction (paragraph 13-120), the Yul system examines the content of the address field. If the address field is blank, the Yul system takes the content of its location counter and uses it as the relevant address. If the address field contains a signed numeric subfield, the algebraic sum of the content of the location counter plus the content of the address field is used as the relevant address. If the address field contains a numeric subfield, this quantity is used as a relevant address. (The Yul system considers a numeric subfield as a pseudo octal address; refer to table 1-4.) If the address field contains a numeric and a signed numeric subfield (the two subfields must be separated by at least one blank), the algebraic sum of both is used as the relevant address. If the address field contains two signed numeric subfields (separated by at least one blank),

the algebraic sum of the content of the location counter plus the two signed numeric subfields is used as relevant address. If the address field contains a symbolic subfield, the numeric equivalent of the symbol (as defined by an earlier or later card in the deck) is used as the relevant address. If the address field contains a symbolic and a signed numeric subfield (separated by at least one blank), the algebraic sum of the equivalent of the symbol plus the signed numeric value is used as the relevant address. A negative address (up to 7777) in the address field is permitted for any order code except 4.

13-127. In the example of sheet 1, attachment 13-4, various symbols representing an address are shown. For instance, HOUSTON is equivalent to address 0014, LP to 0003, BRAUN to 5521, and VON to 5513. Consequently, BRAUN +1 means 5522. The relevant address in line 0018 means 5515 -1, which is 5514. The example shows also a numeric constant in location 5776, which is represented by the symbol ONE.

13-128. It is suggested that AGC programmers give names to constants which indicate their octal value. It is suggested that the symbol OCT00000 be used instead of ZERO, OCT00001 instead of ONE, etc; OCT77777 instead of NEG0, OCT77776 instead of NEG1, etc; OCT40000 instead of NEGSIGN, OCT00177 instead of LOW7, etc. These expressions are ordinary symbols as far as the Yul system is concerned and have to be defined in the same way as any other symbol. In cases where it is inconvenient to use names which indicate the octal value of a constant, symbols such as 1/2, 7/8, .4397, etc which contain at least one nonnumeric character can be used.

13-129. The "head" (or "tail) character of a symbolic subfield in the location or address field can be set such that the Yul system is able to distinguish between symbols of identical spelling introduced by different

programmers. A head character can be any character, including a blank. A blank normally indicates that the head character has not been set for programmer identification. Any other character can be used to identify the programmer who introduced a symbol to which the head character is attached. At the beginning of an assembly, the head character for each symbolic subfield in the location and address fields is a blank. At the appearance of a HEAD (or TAIL) card, the Yul system starts to set the specified head character into each symbolic subfield in the location and address fields and does so until another HEAD card appears. The other HEAD card may request setting another programmer identification or none, i.e., a blank as head character.

13-130. At present, a HEAD card causes the Yul system to check the eighth character (counted from the leftmost nonblank character and including blanks in the count) of each symbolic subfield taken from the location or address field of a detail card. If the eighth character is a blank, the Yul system replaces it by the specified (and prevailing) head character. If the eighth character is a nonblank character, the Yul system leaves it unchanged. Thus, programmers can avoid conflicts in their use of symbols by restricting symbols to seven characters or less and using different heads.

13-131. Figure 13-22 illustrates the decoding of address fields. A symbolic subfield is treated in a special way to allow an instruction (or address constant or clerical code) in a section of coding under one prevailing head to refer to a symbol defined under a different head. In case an instruction has to refer to a symbol (of seven or less characters) defined under a different head, the programmer has to add the head character of that symbol at the end of the symbolic subfield in the address field of the instruction. The head character has to be separated from the nonblank characters of the symbolic subfield by at least one blank to enable the Yul

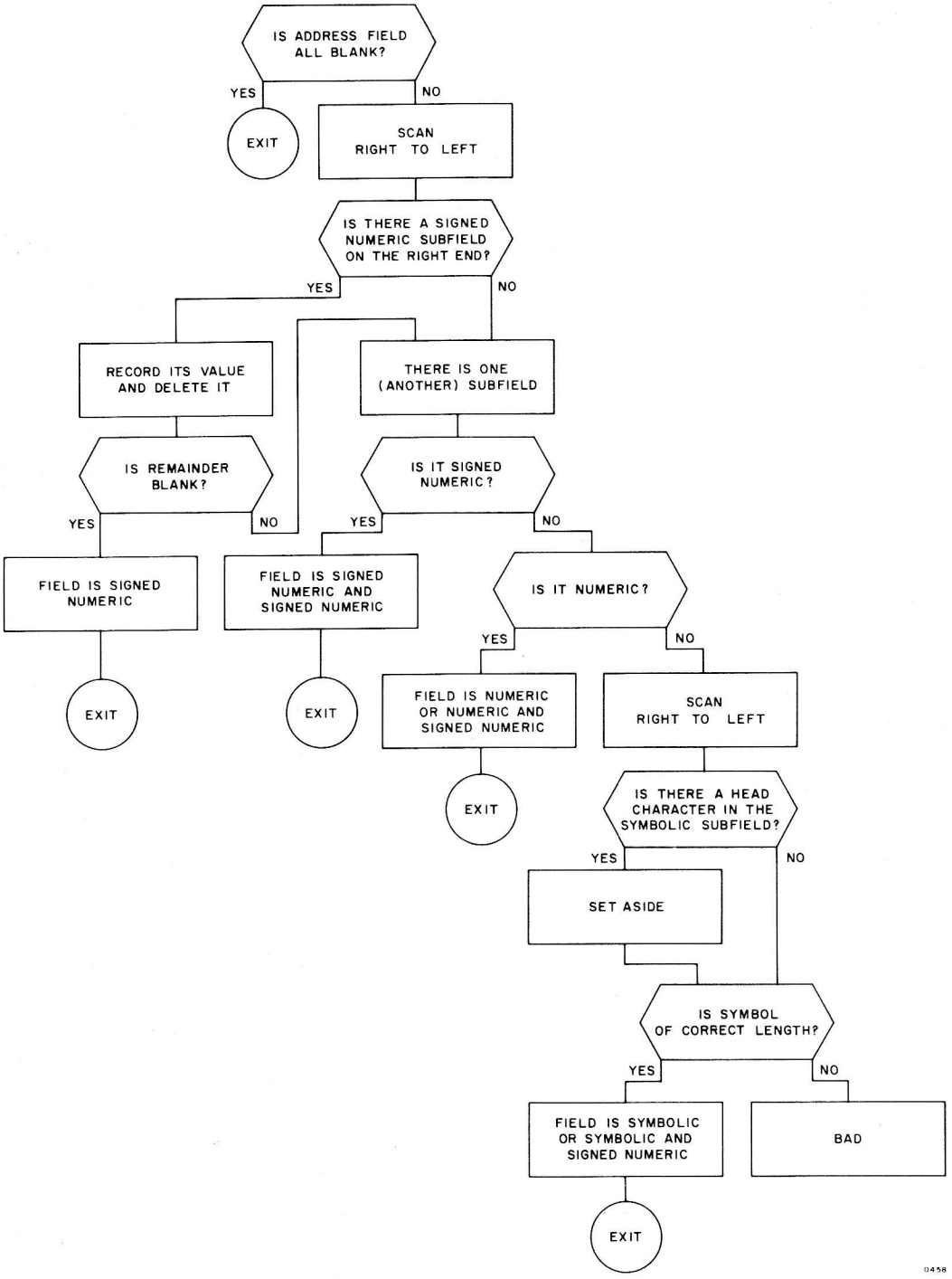


Figure 13-22. Decoding of an Address Field

system to recognize it as a head character. As the Yul system decodes the address field and finds a lone character (single nonblank character) at the tail, it assumes that this is a head character and shifts it into the head (eighth) position. For this reason no symbol of which the last nonblank character is preceded by a blank that separates the last nonblank character from the remainding nonblank characters should be used. (Compare the suggestion at the end of paragraph 13-114.)

13-132. Assume the last HEAD card processed applied the head character M. Then the Yul system will store on tape YULPROGS the symbolic subfields shown to the right for the symbolic subfields punched into the address field of a card and shown below to the left (b for blank):

BRAUN	BRAUNbbM
BRAUNbH	BRAUNbbH
BRAUNbbbH	BRAUNbbH
VONbBbM	VONbBbbM
VONbB	VONbbbbB

The fourth example shows how one can use — if he insists — a symbol with a single nonblank character at the end. He has to punch the head character into the card. Not doing so would let B appear to be a head character. The head characters are not printed into an assembly listing but into the list of symbols (paragraph 13-23).

13-133. The Yul system will be modified such that the head character will be placed in position 9 (instead of position 8) of a symbolic subfield to allow the use of eight-character symbols with a head character. Subroutines will be headed automatically.

13-134. ADDRESS CONSTANT CARDS

13-135. An address constant card is a five-field detail card which contains an address code in the operation field. The operation field of an

address constant card can contain a symbolic subfield but no numeric subfield. Otherwise, table 13-6 applies for address constant cards also. The following five operation codes can be punched into the operation field:

ADRES	Generate 12-bit address as entered into register S.
CADR	Generate complete 15-bit address.
XCADR	Generate extra code address; i.e. ADRES plus 47777. Refer to paragraph 13-120.
P or blank	Generate Interpretive Address Word. Refer to paragraph 6-28.
STORE	Generate Store Code Address Word. Refer to paragraphs 6-32 and 6-34.

13-136. As the Yul system recognizes an address code, it examines the address field following the code, generates that type of address word specified by the operation code, flags the address word stored on tape YULPROGS as a constant for simulation purposes (paragraph 13-123), and prints the octal equivalent of the address word into the assembly listing. The use of a negative address value in the address field is forbidden since the character - punched in column 17 achieves the same effect. The location and the address fields of an address constant card are interpreted in the same way as the location and address fields of an instruction card. A few examples are given as they would appear in an assembly listing:

```
00047  1              ADRES   CDU
00136  0   XVER       ADRES   PRIORITY
00156  1              ADRES   PRIORITY +16D
10071  0              CADR    MARK
00611  0              ADRES   SFTEMP 1
50610  0              XCADR   SFTEMP 1
```

The next four lines represent an interpretive string consisting of one IIW, two IAW's, and one Store Code Address Word:

```
70776  0          DAD      0
00747  0                   SUCCESS
17073  0          P        DBLOW1
32747  1          STORE    SUCCESS
```

13-137. When the Yul system translates addresses for interpretive operations (P, blank, or STORE in operation field), it increments (by one) the Interpretive Address Word (paragraph 6-29). For instance, the address SUCCESS, or 00746, becomes 00747 after translation; 00747 is printed into an assembly listing, and its binary equivalent is stored on tape YULPROGS. The word STORE SUCCESS is generated by adding 32000 to SUCCESS (paragraph 6-32).

13-138. The character - punched into column 17 of an address constant card has a similar effect as the character - punched into column 17 of an instruction card (paragraph 13-123): it causes the Yul system to complement the translated address word after converting it to the binary and octal equivalents. In the case of P or STORE address with a character in column 17, the incrementation is done after the complementation.

13-139. NUMERIC CONSTANT CARDS

13-140. A numeric constant card is a five-field detail card which contains a numeric constant code in the operation field. The operation field of a numeric constant card can contain a symbolic subfield but no numeric subfield. The numeric constant is punched into the address field, which does not consist of subfields in the usual sense. The following four codes can be punched into the operation field (SP means single precision; DP means double precision):

OCT or OCTAL	Generate an SP constant word for the quantity given in octal notation.
DEC	Generate an SP constant word for the quantity given in decimal notation.

2OCT or 2OCTAL	Generate DP constant words for the quantity given in octal notation.
2DEC	Generate DP constant words for the quantity given in decimal notation.

(For the definitions of SP and DP quantities, refer to paragraph 1-48.)

13-141. As the Yul system recognizes a numeric constant code, it examines the address field, translates the numeric constant into one or two constant words, flags the constant word(s) stored on tape YULPROGS as a constant for simulation purposes (paragraph 13-123), and prints the octal equivalent of the constant word into the assembly listing. Any character punched into column 17 of a numeric constant card is jeered at by the Yul system; negative quantities have to be expressed by a sign in the address field. The location field of a numeric constant card is treated the same as the location field of an instruction card, except that in the case of a DP constant, the next location is also reserved and holds the low-order part of the DP constant. A few examples are given as they would appear in an assembly listing:

```
00001  0   ONE         OCT     1
00012  1               OCT     12
77765  0               OCT     -12
77765  0   MINUS12     OCT     77765
00123  1   LOSEFRAC    OCT     12.34 B3
00247  0   OKAYFRAC    OCT     12.34 B+4
40000  0   BIT15       OCT     B14

00000  1               DEC     0
00011  1   9DEC        DEC     9
77766  0               DEC     -9
10440  0               DEC     4384
27713  1               DEC     23.896 B-5
25405  1   EXP(10)     DEC     .22026 E5 B-15
23420  0   NEG 10K     DEC     -E4

00000  1               2OCT    0
00000  1
00000  0               2OCT    4. B+3
00040  1
```

```
00001  0    HIGHONE    2OCT     B15
00000  0
76543  1               2OCT     76543 21012    (This is a
21012  1                                        logical word)
53064  0               2OCT     -1234567076
50701  1

24000  1               2DEC     20. B-5
00000  1
14631  0               2DEC     .4
23146  0
00020  0               2DEC     .001
14223  1
00020  0               2DEC     E-3
14223  1
05174  0               2DEC     E-5 B14
13261  0
21717  0    10 RAD     2DEC     572.957795 B-10
12310  0
43264  0    -4 RAD     2DEC*    -229 183 118 E-6 B+20*
50762  1
```

The first and second words of a DP constant carry the same card number; the second card number printed is preceded by the character C in column 1 (C for continuation line created by the Yul system).

13-142. The address field of a numeric constant card may not be blank and does not consist of subfields in the usual sense. One SP constant (consisting of one 16-bit word, bit 0 being the parity bit) or one DP constant (consisting of two 16-bit words) is formed from the expression in the address field. If the address field is not long enough to hold the expression for a constant, it might be extended by inserting the character * into the operation field after the operation code and at the end of the expression. In this case, the address field begins as usual at column 25 and ends at the next character *, which might be placed anywhere in what normally is the remark field.

13-143. The expression in the address field may be unsigned or may begin with a plus or a minus sign. All blanks in the address field are ignored.

Any number of digits may be used, with or without a decimal or octal point. At most, 10 significant digits (decimal constant) or 14 significant digits (octal constant) are examined, and a warning is printed by the Yul system whenever more digits are present in the address field. These are neglected by the Yul system. If no decimal or octal point is given, the Yul system assumes that it lies to the right of the last digit of the mantissa given. The expression for a decimal constant may contain a decimal exponent (for instance E3 or E+3 for 10^3 or E-5 for 10^{-5}), and the expression for either type of constant may contain a binary scaling factor (B14 or B+14 for 2^{14} or B-1 for 2^{-1}).

13-144. An SP integer has 14 value bits, its sign in bit position 15 of the constant word, and an implied binary point between bits 1 and 0. An SP fraction also has 14 value bits, its sign in bit position 15, and an implied binary point between bits 15 and 14. An SP logical word has 15 value bits, no sign, and an implied binary point between bits 1 and 0. A DP integer has 28 value bits, its sign in bit position 15 of both constant words, and an implied binary point between bits 1 and 0 of the second word. A DP fraction has 28 value bits, its sign in bit position 15 of both constant words, and an implied binary point between bits 15 and 14 of the first word. A DP logical word has 30 value bits, no sign, and an implied binary point between bits 1 and 0 of the second word. Whether a word represents an integer, a fraction, or a logical word depends on the intention of the programmer who builds it into a program.

13-145. Operation code OCT (or OCTAL) is used to specify a binary SP integer or logical word (but no fraction) to be formed from the octal expression in the address field. A warning cuss is printed into the assembly listing if a nonzero fractional part is lost by truncation. The result is stored as an integer if the expression is signed; otherwise it is stored as a logical word. The expression in the address field consists of as

many as three parts:

a. The sign, which is set aside during conversion and is used to decide whether to complement the result. If the sign is omitted, the result is a logical word and is not complemented.

b. The mantissa (assumed to be 1.0 if omitted), consisting of octal digits and an optional octal point.

c. The binary scaling factor (assumed to be B0 if omitted), consisting of the character B, an optional sign, and a one- or two-digit decimal integer: $B \pm n = 2^{\pm n}$.

The absolute value of the expression must be less than octal 40000 for an integer and less than 100000 for a logical word.

13-146. Operation code 2OCT (or 2OCTAL) is used to specify a binary DP integer or logical word (but no fraction) to be formed from the octal expression in the address field. The absolute value of the expression must be less than octal 20000 00000 for an integer or 1 00000 00000 for a logical word. Otherwise all remarks about OCT apply to 2OCT.

13-147. Operation code DEC is used to specify a binary SP integer or fraction (but no logical word) to be formed from the decimal expression in the address field. If the absolute value of the expression is less than 1.0, the result is stored as a rounded fraction. Otherwise, the result is stored as a truncated integer, a warning cuss being printed if a nonzero fractional part was lost by truncation. The expression consists of as many as four parts:

a. The sign (assumed to be plus if omitted), which is set aside during conversion and is used to decide whether to complement the result.

b. The mantissa (assumed to be 1.0 if omitted), consisting of decimal digits and an optional decimal point.

c. The decimal exponent (assumed to be E0 if omitted), consisting of the character E, an optional sign, and a one- or two-digit decimal integer: $E \pm n = 10^{\pm n}$. Parts 1 through 3 constitute the true value of the number.

d. The binary scaling factor (assumed to be B0 if omitted), consisting of the character B, an optional sign, and a decimal integer of one to three digits: $B\pm n = 2^{\pm n}$. The result of multiplying the true value by the binary scaling factor is the value of the expression.

The absolute value of the true value must be no less than 1E-64 and no more than .99999 99999 E63. The absolute value of the expression must be less than 16384 (decimal).

13-148. Operation code 2DEC is used to specify a binary DP integer or fraction (but no logical word) to be formed from the decimal expression in the address field. The absolute value of the expression must be less than 268 435 456 (decimal). Otherwise all remarks about DEC apply to 2DEC.

13-149. LEFTOVER INSTRUCTION AND CONSTANT CARDS

13-150. An AGC programmer has the choice of letting the Yul system assign locations for instructions or constants (including DP constants) during Pass 1 or of postponing such actions until all regular memory reservations have been made (paragraph 13-24). For any instruction or constant taken from an instruction or constant card of which column 1 is blank, the location is assigned by the Yul system during Pass 1. These types of cards are referred to as regular instruction cards and regular constant cards.

13-151. For any instruction or constant taken from an instruction or constant card which contains the character J in column 1 and a symbol in the location field which has not been defined by a card earlier in the deck, a location not assigned during Pass 1 (leftover location) is assigned after Pass 1. These types of cards are called leftover instruction cards and leftover constant cards.

13-152. For any instruction or constant taken from an instruction or constant card which contains the character J in column 1 and either an unsigned numeric subfield in the location field or a symbol which has been defined by a card earlier in the deck, the location indicated is assigned during Pass 1. This assignment has no effect on the setting of the LC. These types of cards are called bound leftover instruction cards and bound leftover constant cards.

13-153. Any instruction or constant taken from an instruction or constant card which contains the character J in column 1 and either a blank location field or a signed numeric subfield in the location field is treated as any regular instruction or constant, and a complaint is printed into the assembly listing.

13-154. CLERICAL CARDS

13-155. A clerical card is a five-field detail card which contains a clerical code in the operation field. A clerical code is an instruction to the Yul system that does not result in the assignment of words to fixed memory. Some clerical codes need information in the location and address fields; others neglect their contents. The address field of clerical cards is decoded (if needed) in most cases in the same way as the address field of instruction cards is decoded. The clerical codes listed below can be punched into the operation field:

SETLOC or LOC	Set the content of the Yul location counter to that address in E or F memory which is specified in the address field. (Paragraphs 13-161 through 13-163.)
BLOCK	Set the content of the location counter to that address in E or F memory which is specified (by means of a block number) in the address field. If this location is not available, set the location

	counter to the address of the next available location of the same block. (Paragraphs 13-164 through 13-166.)
ERASE	Reserve those locations in E memory whose addresses are specified in the address field. Define any symbol present in the location field. (Paragraphs 13-167 through 13-172.)
EQUALS or = or IS	Define the symbol in the location field (which is the location symbol) as meaning the same as the address given in the address field. (Paragraphs 13-173 through 13-176.)
HEAD or TAIL	Use the character given in the location field as HEAD (or TAIL) for each symbol read from a card until the appearance of another HEAD (or TAIL) card. (Paragraphs 13-177 and 13-178.)
MEMORY	During the current assembly, consider addresses given in the address field as belonging to the memory type given in the location field. (Paragraphs 13-179 through 13-181.)
SUBRO	Include the subroutine whose name is given in the address field in the current assembly. This clerical card can be used as soon as the Yul system will be able to deal with AGC subroutines. (Refer to paragraphs 13-13, 13-182, and 13-183.)

13-156. The Yul system classifies AGC 4 memory in three types:

Special and nonexistent memory	0000 through 0077
Erasable (E) memory	0100 through 1777
Fixed (F) memory	2000 through 5777 and 6000 through 7777 in all Banks

It is of interest to compare this classification with the AGC 4 addresses in table 1-4. The Yul classification permits detection of such errors as

the assignment of a constant to erasable memory.

13-157. The Yul system maintains a table during an assembly showing which locations in E and F memory are reserved and which are available. A location in F memory is reserved by assigning an instruction or constant to it; a location in E memory is reserved by assigning a variable to it. This availability table permits the Yul system to detect conflicts in the use of E and F memory. Special or nonexistent memory can never be reserved.

13-158. Instructions and constants (except leftover instructions and constants) are assigned to locations in F memory with the help of the location counter (LC) of the Yul system during Pass 1. The content of the LC is set for the start of a location sequence, as described in paragraph 13-160, and incremented (by one) whenever a word is assigned to an F memory location. (See attachment 13-4.) Leftover instructions and leftover constants are assigned to locations in F memory without the help of the LC after all regular assignments have been made. (Refer to paragraphs 13-24 and 13-50.)

13-159. Variables are assigned to locations in E memory in three different ways. An assignment can be made by means of an ERASE card, which specifies a certain location or a set of consecutive locations, in which case the assignment is made independently of the LC during Pass 1. Assignments also can be made by means of a SETLOC card and one or more ERASE cards. The SETLOC card specifies the location in which the first variable of a group is to be stored, and the ERASE cards specify the number of locations to be reserved for each request. The Yul system assigns the first variable to the first location and the other variables of the group to the subsequent locations with the help of the LC also during Pass 1. Leftover variables (variables not assigned as described above) are assigned

to leftover locations after all regular assignments have been made. These assignments are made independently of the LC.

13-160. The LC can be set by means of a SETLOC card (paragraphs 13-161 through 13-163), a BLOCK card (paragraphs 13-164 through 13-166), or an instruction or constant card with a numeric subfield or a symbolic predefined by equals in the location field (paragraph 13-117).

13-161. SETLOC CARDS

13-162. The function of the SETLOC (or LOC) code is to set the LC. The location field of a SETLOC (or LOC) card must be blank. The address field of a SETLOC card is decoded by the Yul system in the same way as the address field of an instruction card. A few examples are given as they would appear in an assembly listing; two of them were taken from attachment 13-4. Note that the content which has been set into the location counter is printed in print positions 40 through 46.

```
   5776              SETLOC  5776
   5510              SETLOC  2888D
05,6000              LOC      12000
05,6000                LOC   5120D
05,6054              SETLOC  12000 +54
04,7700              SETLOC  12000 -100
05,6054              SETLOC  5120D + 44D
04,7700              SETLOC  5120D -64D
05,6054              SETLOC  5120D +54
05,6054              SETLOC  12000 - 64D
   0205              SETLOC  DISTANCE + 22
   0205              LOC     DISTANCE +18D
   0205              SETLOC
   0206              SETLOC  +1
   0166              SETLOC  -20
   0265              SETLOC  +200D -211
```

13-163. Location values printed by the Yul system (first column) are expressed in real octal addresses with the proper bank numbers in front,

but addresses punched into the address field are always expressed in pseudo addresses (table 1-4). The address field of a SETLOC card is decoded in a similar way as the address field of an instruction card except that a symbol must be defined by an earlier card in the deck. For the examples it was assumed that DISTANCE is the symbol for location 0163. Whenever the address field is blank, the location counter is reset to its last content. Whenever the address field contains only one or two signed numeric subfields, their total is added to the last content of the location counter to set it for a new location. Whenever the address field contains a negative value, a value of an address in special and nonexistent memory, or the value of a nonexisting address, the content of the LC is not set to the illegal location but left as it was.

13-164. BLOCK CARDS

13-165. A block of locations is defined as 256 (decimal) consecutive locations beginning with a location whose address is a multiple of 256. (For AGC 4 a block is identical with a paragraph as a unit of memory.) The location field of a BLOCK card must be blank, as for SETLOC cards. The address field must contain an unsigned block number, which may be followed by a positive signed numeric subfield whose value is less than 256. A block number is the pseudo address of the first location in the block divided by 256 (decimal) or 400 (octal). For this reason each bank of AGC 4 memory consists of four blocks (table 1-4). The following bank and block numbers are given in octal numbers:

Bank 00 (CP through E memory)	Blocks 0 through 3
Bank 01 (FF memory, first half)	Blocks 4 through 7
Bank 02 (FF memory, second half)	Blocks 10 through 13
Bank 03	Blocks 14 through 17
Bank 04	Blocks 20 through 23
Bank 05	Blocks 24 through 27
Bank 06	Blocks 30 through 33
Bank 07	Blocks 34 through 37
Bank 10	Blocks 40 through 43

his table can be continued simply by multiplying (in octal numbers) the bank numbers by 4, which results in the first block number of that bank.

13-166. The function of the BLOCK code is to set the LC to the address of an available location in the specified block. First, the Yul system derives a tentative address, which is equal to the first address in the specified block plus the positive signed numeric subfield if one is present. This address must be in E or F memory. Thereafter, consecutive locations are tested for availability, and the LC is set to the first available location. If the end of the block is found before an available location is found, the content of the LC is left as it was and the Yul system complains. A few examples are given as they would appear in an assembly listing. Note the similarity in form with the printout of SETLOC cards. In all cases, it was assumed that the tentative location was available.

```
    0205        BLOCK   0     +205
    5510        BLOCK   13    +110
    5510        BLOCK   11D   +110
    5510        BLOCK   13    +72D
05,6000         BLOCK   24
05,6000         BLOCK   20D
05,6054         BLOCK   24    +54
```

If a BLOCK card is followed by a set of instruction or constant cards, care must be taken that the location(s) in F memory found by the BLOCK card is not a hole (leftover location or locations) in a CCS pattern.

13-167. ERASE CARDS

13-168. The ERASE code causes the Yul system to reserve locations in E memory. Three types of ERASE cards are provided: the absolute, the relative, and the leftover ERASE cards. Only the second type advances the content of the LC. In the following three paragraphs a few examples are given as they would appear in an assembly listing. Note that the

printout of location fields is not preceded by octal equivalents, as for instructions and constants, but by two location columns containing the first and last locations for each reservation (the same location if only one location was reserved).

13-169. Absolute ERASE cards reserve specified locations in E memory independently of the LC:

```
0150        0150                ERASE   150
0163        0163    CAMBRIGE    ERASE   163
0200        0202    VIENNA      ERASE   200 +2
0217        0220                ERASE   217 -220
```

The expression +n has the meaning "and the next n locations," and -n means "through location n". (For instance, 200 +2 has the same meaning as 200 -202.) CAMBRIGE is being defined as 0163, and VIENNA as 0200.

13-170. Relative ERASE cards reserve locations in E memory with the help of the LC, advancing it just past the locations reserved:

```
            0100                SETLOC  100
0100        0100                ERASE
0101        0101    BONN        ERASE
0102        0103                ERASE   +1
0104        0107    ROMA        ERASE   +1  +2
0110        0112    PARIS       ERASE   +4  -2
0113        0120                ERASE   -120
0121        0146    LONDON      ERASE   -100  -46
```

The two signed numeric subfields are combined by the Yul system, and the net sign determines their meaning.

13-171. Leftover ERASE cards cause the Yul system to reserve leftover locations in E memory:

```
0147        0147    SUDBURY     ERASE   ANYWHERE
0151        0153    WAYLAND     ERASE   ANYWHERE +2
```

ANYWHERE as used here can be followed only by blanks or a positive subfield. The symbol ANYWHERE as used here is a special symbol which is not affected by a prevailing head character.

13-172. A certain location (for instance 0150) can be reserved by punching the address (150) of that location into the address field of an ERASE card. A certain variable (for instance CAMBRIGE) can be assigned to a certain location (0163) by punching the address (163) of that location into the address field. More than one location can be assigned or reserved by punching the address of the first location followed by a signed numeric subfield. Whenever a symbol is contained in the location field of an ERASE card, it is defined as meaning the first (or only) location of that assignment. Whenever no address is given in the address field (relative ERASE card), the Yul system reserves or assigns that location whose address is contained in the LC and also further consecutive locations if one or two signed numeric subfields are punched into the address field. The content of the LC is advanced to the address of the next location. Leftover variables are identified by the code ANYWHERE in the address field and must contain a symbol in the location field.

13-173. EQUALS CARDS

13-174. The EQUALS (or = or IS) code is used to define a symbol without reserving a location. The location field of an EQUALS card must be symbolic and contain the symbol to be defined. The address field of an EQUALS card is decoded in the same way as the address field of an instruction card and must yield a value in the range of legal address values. A few examples are given as they would appear in an assembly listing:

```
0000    A       EQUALS  0
0001    Q       IS      1
0002    Z       =       2
0003    LP      EQUALS  1  +2
```

```
0004   IN0        =        0 +4
0005   IN1        =        5
0006   IN2        =        0006
0010   OUT0       EQUALS   10
0007   IN3        =        OUT0 -1
0011   OUT1       IS       BNK -4
0012   OUT2       IS       OUT1 +1
0013   OUT3       IS       OUT1 +2
0014   OUT4       IS       OUT1 +3
0015   BNK        EQUALS   15
0015   BANKREG    =        BNK
```

None of these operations makes use of the LC.

13-175. During Pass 1 all symbols are defined by the Yul system which are expressed by a numeric subfield in the address field, by a numeric and a signed numeric subfield in the address field, or by a symbolic subfield which has already been defined with or without a signed numeric subfield in the address field. (IN3 is described by OUT0 -1; OUT0 is defined by an earlier card in the deck. BANKREG is defined by BNK, which is defined by an earlier card as 15.) A symbol which is expressed by a symbolic subfield which has not been defined yet cannot be defined during Pass 1. (OUT1 is defined by BNK -4; BNK has not been defined yet.) At the beginning of Pass 2 the Yul system knows the definition BNK = 15 and is able to define OUT1 through OUT4 during Pass 2. The Yul system makes two attempts to define an equated symbol (not every symbol); one during Pass 1, and failing, then another at Pass 2.

13-176. An EQUALS card may refer to the LC but does not change its content. Assuming the LC is set to 0400, the following examples apply:

```
0377   LAST       =        -1
0400   THIS       IS
0401   NEXT       EQUALS   +1
0405   ANOTHER    =        -3 +8D
```

13-177. HEAD OR TAIL CARDS

13-178. The use of HEAD (or TAIL) cards is described in paragraphs 13-129 through 13-133. A HEAD card contains the operation code HEAD or TAIL in the operation field and the selected head (for instance M) in the location field. In the example it is assumed that programmer M wants to use the symbol OVCTR to symbolize location 0548 in his program and that symbol OVCTR is already used to note location 0034:

```
          M       HEAD
0548    OVCTR     =          548
```

Note that the assembly printout does not show the head character with symbol OVCTR. The list of symbols (similar to sheet 2 of attachment 13-4) may include the statement:

```
OVCTR    M          0548        DEFINED BY EQUALS
```

13-179. MEMORY CARDS

13-180. The MEMORY code causes the Yul system to change the classification of memory types from the ones shown in paragraph 13-156 for the duration of the current program only. Its main use is to legalize the assembly of programs which are to be loaded into erasable memory for test purposes. A MEMORY card contains the code MEMORY in the operation field and the word FIXED, ERASABLE, or SPEC/NON in the location field. The addresses to be redefined as F, E, or special and nonexistent locations are specified in the address field. A few examples are given as they would appear in an assembly listing. Note the similarity with the print format of ERASE cards:

```
0300        1777    FIXED       MEMORY   300  -1777
0040        0077    ERASABLE    MEMORY   40   +37
0030        0030    SPEC/NON    MEMORY   30   -30
```

13-181. The first example causes locations 0300 through 1777 to be regarded as F memory without disturbing the status of locations 0000 through 0277 and 2000 through 7777 in all Banks. The address field must always contain a numeric and a signed numeric subfield. For the last example expression 30 +0 could have been used as well as 30 -30. (Compare with paragraph 13-169.) No attempt is made to check the set of MEMORY cards submitted for consistency as a set; each card is obeyed as it appears. All MEMORY cards of a deck must be placed before any card which is not a remark card or another MEMORY card. The reason for this is that consistent error checking demands that all changes in memory configuration be known before any words are assigned or locations are reserved.

13-182. SUBRO CARDS

13-183. The code SUBRO (subroutine) will be used to specify that the subroutine whose name is given in the address field is to be included in the program or subroutine being assembled. An example is given as it would appear in an assembly listing:

```
        SUBRO       INTPRET
```

This means that the subroutine INTPRET shall be used in the current program. When the Interpreter will be assembled as a subroutine (as defined in paragraph 13-13), it will probably be called INTPRET. More details on the use of subroutines will be given later. (See the note after paragraph 13-201.)

13-184. MERGE CONTROL CARDS

13-185. Merge control cards are used with the operations ASSEMBLE REVISION and ASSEMBLE VERSION. Four different types of merge control cards are provided for the Yul system: The DELETE card, a five-

field detail card with the code DELETE in the operation field, is used to delete information from tape YULPROGS. The CARDNO card, a five-field detail card with the code CARDNO in the operation field, is used to change card numbers on tape YULPROGS. The acceptor card, a two-field detail card with the character = in column 1, is used to get to any desired section of a program when a program consists of two or more card number sequences (paragraph 13-112). The tins (tuck in new section) card, a two-field detail card with the character T in column 1, allows the insertion of a new section between two existing card number sequences.

13-186. The three modes of merging (during operations ASSEMBLE REVISION or ASSEMBLE VERSION) are replacement, insertion, and deletion. A detail card (following a Yul director ASSEMBLE REVISION or ASSEMBLE VERSION) which has a card number equal to that of a card image of the same program stored on tape YULPROGS causes the Yul system to replace the stored card image by the content of the new detail card. In case the card number exists two or more times within the particular program, only the first card image with that card number is replaced. In order to get into another numbering sequence, an acceptor card has to precede the new detail card.

13-187. A detail card (following a Yul director ASSEMBLE REVISION or ASSEMBLE VERSION) which has a card number unequal to that of any card image of the same program stored on tape YULPROGS causes the Yul system to insert the information contained in the detail card between card images with lower and higher card numbers on tape YULPROGS. In case the program consists of two or more card number sequences, the insertion is made into the first sequence. If information is to be inserted into another numbering sequence, an acceptor card has to precede the additional detail card.

13-188. A DELETE card (following a Yul director ASSEMBLE REVISION or ASSEMBLE VERSION) causes the Yul system to delete the specified card image(s) of the particular program from tape YULPROGS. The location field is either blank or OUTOFSEQ (out of sequence), and the address field is either blank or THROUGH nnnnnn, where nnnnnn is a card number. If the location field is blank, this card causes the Yul system to delete from tape YULPROGS that well-ordered card image of which the card number is given in the card number field of the DELETE card. If the location field is OUTOFSEQ, this card causes the Yul system to delete from tape YULPROGS that ill-ordered card image of which the card number is given in the card number field. In either case, the content of the DELETE card is printed in the assembly listing in place of that of the deleted card but is not kept as part of the symbolic record of the program on tape YULPROGS. In case the program consists of two or more card number sequences, an acceptor card has to be used to define the sequence in which a card image is to be deleted.

13-189. If the address field of the DELETE card is blank, just one card image is deleted. If the address field of the DELETE card contains the word THROUGH (or THRU), at least one blank, and a card number, deletion continues until the card image whose number is given in the address field has been deleted. The word SEQBREAK instead of a number may follow the word THROUGH, in which case deletion continues until the next sequence break. The number punched after THROUGH is interpreted by the Yul system as follows:

Punched	Interpreted	Punched	Interpreted
1	000100	.1	000010
12	001200	1.2	000120
123	012300	12.34	001234
1234	123400	1234.	123400
12345	123450	1234.5	123450
123456	123456	1234.56	123456

13-190. Any of the following conditions causes the Yul system to reject a DELETE card and not to carry out the revision or version:

a. If the Yul system detects a meaningless location or address field in a DELETE card, including the self-contradictory case in which the number in the card number field is a sequence break and the location field contains OUTOFSEQ.

b. If the Yul system detects a sequence break before it is able to find a card image with a card number equal to that contained in the card number field of the DELETE card.

c. If the Yul system detects that the number given in the address field of a DELETE card is not larger than the number given in the card number field.

d. If the Yul system detects a sequence break before it is able to find a card image with a card number equal to that contained in the address field of the DELETE card or if the Yul system detects a card image with a card number larger than that given in the address field of the DELETE card.

13-191. A CARDNO card (following a Yul director ASSEMBLE REVISION or ASSEMBLE VERSION) causes the Yul system to change a card number of a particular card image on tape YULPROGS. A CARDNO card contains the code CARDNO in the operation field, CORRECT or CHANGE in the location field, and TO nnnnnn in the address field. The number punched after TO is interpreted as the number after THROUGH, explained in paragraph 13-189. A CORRECT CARDNO card may be ill-ordered and can change only an ill-ordered card number on tape YULPROGS. A CHANGE CARDNO card has to be well-ordered and can change only well-ordered card numbers. The card

```
1243    CORRECT   CARDNO  TO  1235
```

causes the Yul system to correct the card number (of a card image stored on tape YULPROGS) 1243 to 1235. The card

```
06201   CHANGE    CARDNO  TO  620.5
```

causes the Yul system to change the card number 06201 to 06205. The

content of a CARDNO card is printed in the assembly listing but is not stored on tape YULPROGS. In case the program consists of two or more card number sequences, an acceptor card has to be used to define the sequence in which a card number has to be corrected or changed. CARDNO cards are subject to error conditions a and b listed in paragraph 13-190 except that the word OUTOFSEQ has to be replaced by CORRECT.

13-192. An assembly listing contains an indication everywhere a change was made in the program during the last revision. However, this indication does not appear in the assembly listing of the next revision. The indications of changes make it possible to determine what was done during a revision; however, it is not possible (in general) to reconstruct the preceding assembly. The Yul system prints the character * in position 8 of every line when a card image has been changed or inserted. The deletion of a card or set of cards can be recognized by the printout of a DELETE card. The correction or change of a card number can be recognized by the printout of a CARDNO card. The content of an acceptor card is not printed into an assembly listing, but the Yul system prints the character @ in position 8 of each line containing an "accepted" card. For instance, if a log card is "accepted", the character @ is printed on each page with the content of the log card until another log card appears. Therefore, each page of a section can show whether any revision was made in this section.

13-193. The acceptor card is provided to get to any desired section of the program being revised. An acceptor card contains the character = in column 1 and causes the Yul system (when following a Yul director ASSEMBLE REVISION) to accept revision cards (cards described in paragraphs 13-186 through 13-191) for that section of the program which follows a card image (on tape YULPROGS) with identical contents in the card number field and columns 9 through 80. If a log card is chosen to mark a section for revision (as is done in most cases), the card number field of the acceptor card must contain the word LOG. Most AGC programs

consist of separately numbered sections, each headed by a log card. A revision deck for such a program might consist of acceptor cards for each log card heading a section in which changes are to be made. Each acceptor card is followed by the revision cards for that section. All sections of the revision deck must be in the same order as the sections of the program. However, not all sections need be represented. The content of an acceptor card is not printed in an assembly listing.

13-194. The tins card is provided to insert a new section (independent card number sequence) into a program. A tins card is organized in the same way as a log card except that the character T instead of the character L is punched into column 1. During merging, a tins card is merged ahead of a log card or other sequence break. The images of detail cards following immediately the tins card are inserted also (on tape YULPROGS) ahead of the log card images or other sequence breaks mentioned before. Once merged, a tins card becomes an ordinary log card and the content of column 1 is changed to character L.

13-195. LOG CARDS

13-196. A log card usually contains the title of a program section and is identified by the symbol L punched in column 1. A log card is a two-field card which may contain a date in columns 68 through 75, as indicated by figure 13-23. The symbol L is not printed into an assembly listing. The contents of columns 2 through 7 and 9 through 67 are printed as punched. The contents are stored in the Yul system and printed on each page of an assembly listing as a page subhead (second line) until another log card takes over. The content of column 8 governs the upspacing as the content of the card is printed the first time; thereafter, upspacing 2 is used. (The content of column 8 is never printed, as with all cards.) If any one of columns 68, 69, 71, 72, 74, and 75 is not blank, the character - is inserted

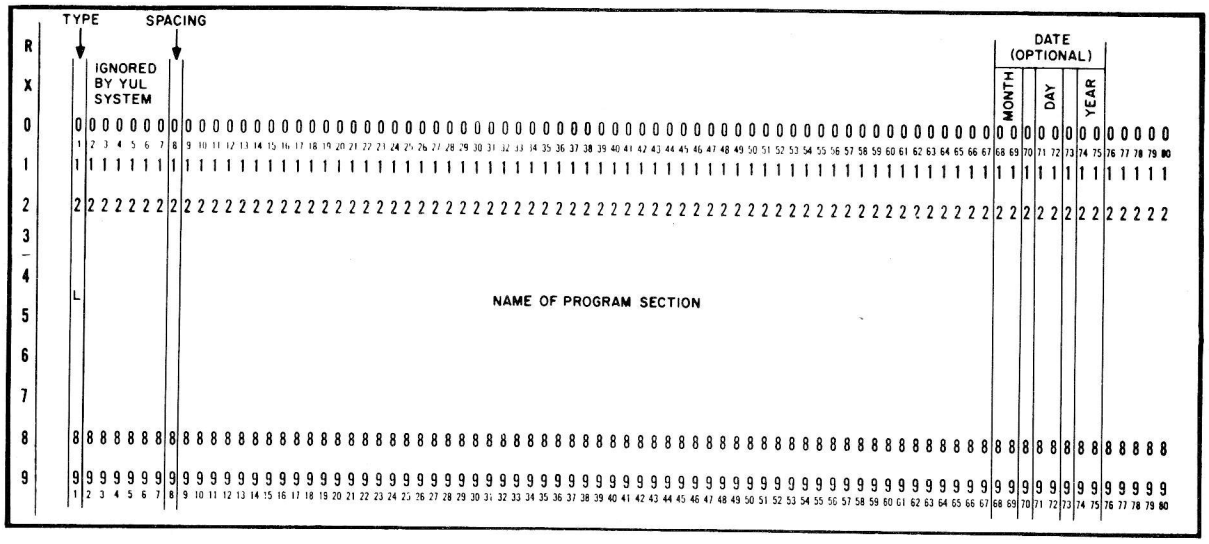


Figure 13-23. Log Card

into positions 70 and 73 of the print. USER'S OWN PAGE NO. nnn is inserted into print positions 81 through 104 of each subheading: $1 \leq nnn \leq 999$. On the first appearance of a log card in a listing, nnn is equal to 1. Log cards break a sequence and begin a new count in card numbering.

13-197. REMARK CARDS

13-198. Four types of remark cards are provided for the Yul system: the general remark card (also called remark card), the right-print remark card, the aligned remark card, and the page remark card (table 13-4). Remark cards are two-field detail cards consisting of the card number field and the remark field (figure 13-21). They are used to store information on tape YULPROGS which does not affect the Yul system and which is printed in an assembly listing mainly for explanatory purposes.

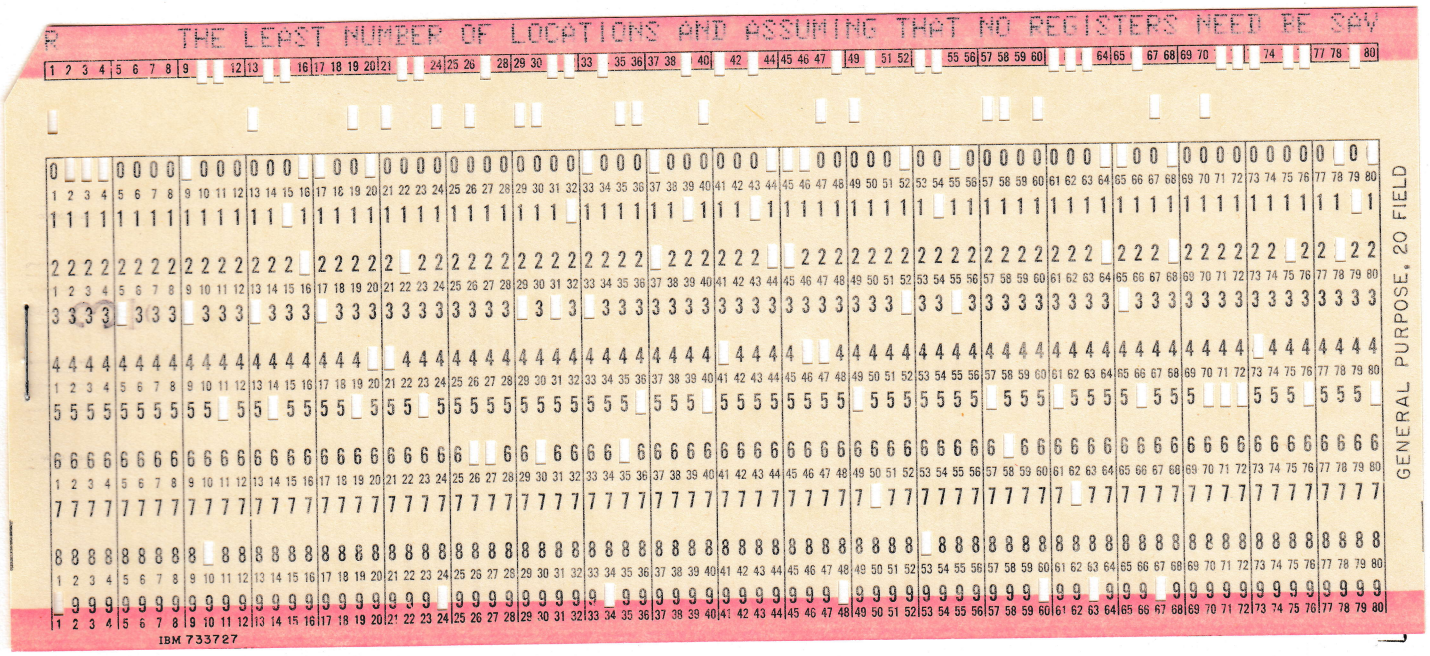
13-199. Remarks to be printed in print positions 9 through 80 of an assembly listing are punched into the remark field of general remark cards.

Their card numbers, with the character R in front of them, are printed in an assembly listing (attachment 13-4). Remarks to be printed in positions 81 through 120 of an assembly (on the line of the preceding general remark card) are punched into the remark field of right-print remark cards. Their card numbers are not printed in an assembly listing. A right-print remark card is identified by the character 9 punched into column 8.

13-200. Remarks to be printed in positions 49 through 120 of an assembly listing are punched into the remark field of aligned remark cards. Their card numbers, together with the character A, are printed into an assembly listing.

13-201. Remarks to be printed at the top of a page, after the system-supplied page head and after the programmer-supplied subhead (by means of a log card, if any), are punched into the remark field of a page remark card. Their card numbers, together with the character P, are printed in an assembly listing.

NOTE: Descriptions of high-speed printer outputs, Yul system complaints, console typewriter outputs, and the use of subroutines will be added during 1964.



Attachment 13-1. Yul Language Card

```
Y        A
L        3S
R0001
0002  9M
R0003  T
0004  9E
R0005  4S
R0006  2M
00061  1
00062  2H
0007  2
0008
0009  4
0010  2
0011
0012
0013  2
0014
0015  9
0016
A0017
0018
0019
0020
0021
0022
```

Attachment 13-2. Printout of IBM 407
with 80-80 Board Inserted

```
Y       ASSEMBLE NEW AGC4 PROGRAM TRIVIUM BY IMA NIT-PICKER
L       3SECTION 1 AND ONLY OF SAMPLE PROGRAM
R0001             CANONICAL PROBLEM FOR AGC4- TO PLACE IN REGISTER HOUSTON THE NU
 0002  9MBER OF ONES IN REGISTER HUNTSVIL, USING
R0003   THE LEAST NUMBER OF LOCATIONS AND ASSUMING THAT NO REGISTERS NEED BE SAV
 0004  9ED.  THIS SOLUTION, REQUIRING 10 IN-
R0005  4STRUCTIONS AND TWO CONSTANTS, IS THE BEST KNOWN TO DATE.
R0006  2MISCELLANEOUS DEFINITIONS AND CONSTANTS HERE.  ENTER ROUTINE AT WERNER.
 00061 LP       =      3
 00062 2HOUSTON  EQUALS 14
 0007  2        SETLOC 5776
 0008   ONE     OCT    1
 0009  4        4      7777
 0010  2        SETLOC 2888D
 0011   WERNER  CAF    ONE
 0012           EXTEND                    NOTICE THAT THIS IS INDEX 5777.
 0013  2        MP     HUNTSVIL           ZERO TO A, C(HUNTSVIL) TO LP.
 0014     VON   TS     HOUSTON            ZERO COUNT (INITIALLY), ACCUMULATE COUNT
 0015  9 (SUBSEQUENTLY).
 0016           CCS    LP                 EXAMINE AND CLEAR BIT 15 (FIRST TIME).
A0017                                     SUBSEQUENTLY, DO BITS 1-14 IN ORDER.
 0018             TC   -1                 IF BIT=0 BUT C(LP) NON-ZERO, LOOK AGAIN.
 0019           TC     BRAUN +1           EXIT WHEN C(LP) EXHAUSTED.
 0020           XCH    HOUSTON            SET UP INCREMENT IF BIT=1.
 0021           AD      ONE               C(LP) CAN BE -0 FIRST TIME ONLY.
 0022  3BRAUN   TC     VON
```

R0001
R0003
R0005

R0006

00061
00062

0007

0008
0009

0010

0011
0012
0013

0014
0016
A0017
0018
0019
0020
0021
0022

Attachment 13-3. Printout of IBM 407 with
Honeywell Format Board Inserted

```
R0001                CANONICAL PROBLEM FOR AGC4- TO PLACE IN REGISTER HOUSTON THE NUMBER OF ONES IN REGISTER HUNTSVIL, USING
R0003    THE LEAST NUMBER OF LOCATIONS AND ASSUMING THAT NO REGISTERS NEED BE SAVED.  THIS SOLUTION, REQUIRING 10 IN-
R0005    STRUCTIONS AND TWO CONSTANTS, IS THE BEST KNOWN TO DATE.


R0006    MISCELLANEOUS DEFINITIONS AND CONSTANTS HERE.  ENTER ROUTINE AT WERNER.

 00061   LP       =       3
 00062   HOUSTON  EQUALS  14

 0007             SETLOC  5776

 0008    ONE      OCT     1
 0009             4       7777


 0010             SETLOC  2888D

 0011    WERNER   CAF     ONE
 0012             EXTEND                      NOTICE THAT THIS IS INDEX 5777.
 0013             MP      HUNTSVIL            ZERO TO A, C(HUNTSVIL) TO LP.
 0014     VON     TS      HOUSTON             ZERO COUNT (INITIALLY), ACCUMULATE COUNT (SUBSEQUENTLY).
 0016             CCS     LP                  EXAMINE AND CLEAR BIT 15 (FIRST TIME).
A0017                                        SUBSEQUENTLY, DO BITS 1-14 IN ORDER.
 0018                TC   -1                  IF BIT=0 BUT C(LP) NON-ZERO, LOOK AGAIN.
 0019             TC      BRAUN +1            EXIT WHEN C(LP) EXHAUSTED.
 0020             XCH     HOUSTON             SET UP INCREMENT IF BIT=1.
 0021             AD       ONE                C(LP) CAN BE -0 FIRST TIME ONLY.
 0022    BRAUN    TC      VON
```

```
 95175

R0001
R0003
R0005

R0006

 00061
 00062

 0007

 0008
 0009

 0010

 0011
 0012
 0013
E■■■■■■
 0014
C0015
 0016
A0017
 0018
 0019
 0020
 0021
 0022

   END
```

Attachment 13-4. Yul Assembly Listing

```
 95175  YUL SYSTEM FOR AGC4: NEW PROGRAM TRIVIUM BY IMA NIT-PICKER                    NOV 24, 1963  (MAIN)  PAGE    1

        SECTION 1 AND ONLY OF SAMPLE PROGRAM                                  USER'S OWN PAGE NO.   1

R0001                CANONICAL PROBLEM FOR AGC4: TO PLACE IN REGISTER HOUSTON THE NUMBER OF ONES IN REGISTER HUNTSVIL, USING
R0003   THE LEAST NUMBER OF LOCATIONS AND ASSUMING THAT NO REGISTERS NEED BE SAVED.  THIS SOLUTION, REQUIRING 10 IN-
R0005   STRUCTIONS AND TWO CONSTANTS, IS THE BEST KNOWN TO DATE.

R0006   MISCELLANEOUS DEFINITIONS AND CONSTANTS HERE.  ENTER ROUTINE AT WERNER.

 00061                                  0003  LP       =       3
 00062                                  0014  HOUSTON  EQUALS  14

 0007                                   5776           SETLOC  5776

 0008                           5776    00001 0  ONE    OCT     1
 0009   END OF BLOCK            5777    4 7777 0        4       7777


 0010                                   5510           SETLOC  2888D

 0011                           5510    3 5776 1  WERNER CAF     ONE
 0012                           5511    2 5777 1         EXTEND
 0013                           5512    4 ####           MP      HUNTSVIL      ZERO TO A, C(HUNTSVIL) TO LP.
E####### "HUNTSVIL" IS UNDEFINED
 0014                           5513    5 0014 1    VON  TS      HOUSTON       ZERO COUNT (INITIALLY), ACCUMULATE COUNT
C0015                                                                          (SUBSEQUENTLY).
 0016                           5514    1 0003 0         CCS     LP            EXAMINE AND CLEAR BIT 15 (FIRST TIME).
A0017                                                                          SUBSEQUENTLY, DO BITS 1-14 IN ORDER.
 0018                           5515    0 5514 1           TC    -1            IF BIT=0 BUT C(LP) NON-ZERO, LOOK AGAIN.
 0019                           5516    0 5522 1         TC      BRAUN +1      EXIT WHEN C(LP) EXHAUSTED.
 0020                           5517    3 0014 1         XCH     HOUSTON       SET UP INCREMENT IF BIT=1.
 0021                           5520    6 5776 1         AD       ONE          C(LP) CAN BE -0 FIRST TIME ONLY.
 0022                           5521    0 5513 0  BRAUN  TC      VON


    END OF NEW PROGRAM TRIVIUM BY IMA NIT-PICKER
```

95175

SYMBOL

BRAUN

HOUSTON

HUNTSVIL

LP

ONE

VON

WERNER

Attachment 13-4. Yul Assembly Listing

SYMBOL	DEFINITION	HEALTH OF DEFINITION
BRAUN	5521	NORMALLY DEFINED
HOUSTON	0014	DEFINED BY EQUALS
HUNTSVIL	****	UNDEFINED
LP	0003	DEFINED BY EQUALS
ONE	5776	NORMALLY DEFINED
VON	5513	NORMALLY DEFINED
WERNER	5510	NORMALLY DEFINED

95175

SUMMARY

TOTAL:

Attachment 13-4. Yul Assembly Listing

SUMMARY OF SYMBOL TABLE ANALYSIS

1	UNDEFINED
2	DEFINED BY EQUALS
4	NORMALLY DEFINED

TOTAL: 7	

```
95175

MEMORY 1

0000
0060  TO
2000
5510
5522  TO
5776
03,6000
15,6000  TO
21,6000
```

Attachment 13-4. Yul Assembly Listing

MEMORY TYPE & AVAILABILITY DISPLAY

0000	TO	0057	SPECIAL OR NONEXISTENT MEMORY
0060	TO	1777	AVAILABLE ERASABLE MEMORY
2000	TO	5507	AVAILABLE FIXED MEMORY
5510	TO	5521	RESERVED FIXED MEMORY
5522	TO	5775	AVAILABLE FIXED MEMORY
5776	TO	5777	RESERVED FIXED MEMORY
03,6000	TO	14,7777	AVAILABLE FIXED MEMORY
15,6000	TO	20,7777	SPECIAL OR NONEXISTENT MEMORY
21,6000	TO	34,7777	AVAILABLE FIXED MEMORY

95175

OCCUPIE

5510

Attachment 13-4. Yul Assembly Listing

OCCUPIED LOCATIONS			PAGE	OCCUPIED LOCATIONS			PAGE
5510	TO	5521	1	5776	TO	5777	1

95175

PARAGRAF

5400

Attachment 13-4. Yul Assembly Listing

95175 YUL SYSTEM FOR AGC4: NEW PROGRAM TRIVIUM BY IMA NIT-PICKER NOV 24, 1963 PAGE 6

PARAGRAPHS GENERATED FOR THIS PROGRAM: ADDRESS LIMITS AND THE MANUFACTURING LOCATION CODE ARE SHOWN FOR EACH.

5400 TO 5777 PARAGRAPH # 013 STICK # R2A SENSE LINE SET 3 (WIRES 49-64)

```
95175

OCTAL L

ALL VAL

5400
5410
5420
5430

5440
5450
5460
5470

5500
5510
5520
5530

5540
5550
5560
5570

5600
5610
5620
5630

5640
5650
5660
5670

5700
5710
5720
5730

5740
5750
5760
5770

THE ASSE
```

Attachment 13-4. Yul Assembly Listing

OCTAL LISTING OF PARAGRAPH # 013, WITH PARITY BIT IN BINARY AT THE RIGHT OF EACH WORD; "@" DENOTES UNUSED FIXED MEMORY.

ALL VALID WORDS ARE BASIC INSTRUCTIONS EXCEPT THOSE MARKED "I" (INTERPRETIVE OPERATOR WORDS) OR "C" (CONSTANTS).

5400	@	@	@	@	@	@	@	@
5410	@	@	@	@	@	@	@	@
5420	@	@	@	@	@	@	@	@
5430	@	@	@	@	@	@	@	@
5440	@	@	@	@	@	@	@	@
5450	@	@	@	@	@	@	@	@
5460	@	@	@	@	@	@	@	@
5470	@	@	@	@	@	@	@	@
5500	@	@	@	@	@	@	@	@
5510	35776 1	25777 1	BAD WORD	50014 1	10003 0	05514 1	05522 1	30014 1
5520	65776 1	05513 0	@	@	@	@	@	@
5530	@	@	@	@	@	@	@	@
5540	@	@	@	@	@	@	@	@
5550	@	@	@	@	@	@	@	@
5560	@	@	@	@	@	@	@	@
5570	@	@	@	@	@	@	@	@
5600	@	@	@	@	@	@	@	@
5610	@	@	@	@	@	@	@	@
5620	@	@	@	@	@	@	@	@
5630	@	@	@	@	@	@	@	@
5640	@	@	@	@	@	@	@	@
5650	@	@	@	@	@	@	@	@
5660	@	@	@	@	@	@	@	@
5670	@	@	@	@	@	@	@	@
5700	@	@	@	@	@	@	@	@
5710	@	@	@	@	@	@	@	@
5720	@	@	@	@	@	@	@	@
5730	@	@	@	@	@	@	@	@
5740	@	@	@	@	@	@	@	@
5750	@	@	@	@	@	@	@	@
5760	@	@	@	@	@	@	@	@
5770	@	@	@	@	@	@	C: 00001 0	47777 0

THE ASSEMBLY WAS BAD. NO BINARY RECORDS FROM IT WERE WRITTEN ON YULPROGS. 1 LINES CUSSED BETWEEN PAGES 1 AND 1.

```
7  ASSEMBLY FOR AGC4:
NEW PROGRAM TRIVIUM BY IMA NIT-PICKER
7  (ASSEMBLER  CHECKOUT INCOMPLETE)
7  BEGIN ASSEMBLY
7  END YUL  PASS 1
7  END YUL  PASS 2
7  ↓↓↓ BAD ASSEMBLY
```

Attachment 13-5. Printout of Console Typewriter