21. Brooker, R., and Morris, D. An assembly program for a phrase structure language. *Comput. J. 3* (Oct. 1960).

22. ——, and ——. A general translation program for phrase structure languages. *J. ACM 9* (Jan. 1962), 1.

23. ——. Trees and routines. *Comput. J. 5*, 1 (Apr. 1962).

24. Ledley, R., and Wilson, J. Automatic-programming language translation through syntactic analysis. *Comm. ACM 5* (Mar. 62), 145.

25. Robinson, J. Preliminary codes and rules for the automatic parsing of English. RAND RM 3339, Santa Monica, Dec. 1962.

26. Ginsberg, S., and Rice, H. Two families of languages related to ALGOL. *J. ACM 9* (July 1962), 350.

27. Floyd, R. Syntactic analysis and operator precedence. *J. ACM 10* (July 1963), 316.

28. Gilbert, P., Hosler, J., and Schager, C. Automatic programming techniques. TDR-62-632, Rome Air Dev. Ctr., Rome, N. Y., Dec. 1962.

29. Barnett, M. Syntactic analysis by digital computer. *Comm. ACM 5*, 10 (Oct. 1962), 515.

30. Ross, D. On the algorithmic theory of language. ESL-TM-156, Elect. Systems Lab., MIT, Cambridge, Mass., Nov. 1962.

31. Bastian, L., Foley, J., and Petrick, S. On the implementation and usage of a language for contract bridge bidding. Proc. of the Int. Symp. on Symbolic Languages in Data Processing, Rome, Italy, Apr. 1962, 741–758.

32. Naur, P. (Ed.) Revised report on the algorithmic language ALGOL 60. *Comm. ACM 4* (Jan. 1963), 1.

33. Matthews, H. Discontinuity and asymmetry in phrase structure grammars. *Inform. Contr. 6*, 2 (June 1963), 137-146.

34. Kuno, S., and Oettinger, A. Syntactic structure and ambiguity of English. Proc. 1963 Fall Joint Comput. Conf., Spartan Books, Baltimore, Md., 1963.

35. ——. New techniques for repetitive path elimination. Mathematical Linguistics and Automatic Translation, Sec. XI, Rep. No. NSF-13, Harvard Comput. Lab., Cambridge, Apr. 1964.

36. Abbott, R. Right-to-left parsing using a predictive grammar. Mathematical Linguistics and Automatic Translation, Sec. VIII, Rep. No. NSF-13, Harvard Comput. Lab., Cambridge, Apr. 1964.

37. Kay, M. A general procedure for rewriting strings. Paper, 1964 Ann. Mtg., Assoc. for Machine Translation and Computational Linguistics, U. of Indiana, Bloomington, 1964.

38. Griffiths, T. Turing machine recognizers for general rewriting systems. Proc. IEEE Symp. Switching Circuit Theory and Logical Design, Princeton, Nov. 1964, 47-56.

BLNSYS—A 1401 Operating System with Braille Capabilities

J. B. Landwehr, C. McLaughlin, H. Mueller, M. Lichstein and S. V. Pollack
University of Cincinnati, Cincinnati, Ohio*

BLNSYS is an operating system designed for a 4K 1401 with common optional features and two attached tape drives. Printed output of this system or of executing programs may be in either English or braille. Even though this system was written for a small machine with minimal peripheral equipment, jobs may be batched, so that card handling and lost processing time is at a minimum. This system will perform any or all of the following users specified functions: assemble SPS source decks, post list, produce condensed or uncondensed object decks, execute user's program, list card input to a program, list punched output, provide a storage dump, execute a program submitted for execution as an uncondensed object deck under debugging trace control, card-to-braille conversion, brailled listings of 7040 IBSYS batch output, and update or duplicate the system tape itself. Input-output subroutines are also included in the system.

Introduction

Previous discussions about the suitability of blind people for work in computer programming have pointed out that the production of a readable braille on the conventional high-speed line printer was a pivotal step in the formulation of techniques necessary for the successful training and performance of such personnel [1, 2]. Initial experience with several trainees indicated that the blind programmer equipped with routines which print input, output and/or memory contents in braille is no more dependent on external help than is his sighted colleague. It was also seen that, for training larger groups, it would be desirable to incorporate these braille routines into a generalized operating structure which would allow the processing of a succession of jobs to proceed smoothly and efficiently, with a minimum of intervention. The resulting system (called BLNSYS for Blind Systems and for other obvious reasons) written for the 1401, turned out to be a very effective means for handling general 1401 processing as well as those jobs prepared by blind programmers requesting braille output.

General Description

In its present form BLNSYS requires a 4K 1401 with two attached tape drives (additional drives may be used but are not necessary). (The attempt was made to define the system for the smallest possible machine configuration so as to give it the widest possible usefulness.) The machine must also be equipped with multiply-divide, extended programming, high-low-equal compare, column binary and sense switches. Overall operation is supervised by a monitor so that successive jobs are handled automatically. Input, output, exit, dump and various diagnostic routines

Work reported here was made possible by NIH Facility Research Support Grant FR 00010 and by a grant from VRA RD-1485-S-65.

* Medical Computing Center.

are also available to the user so that the structure functions as a true operating system.

The design of this system posed several interesting problems due to the severe limitations with respect to memory and external storage devices. The former restrictions precluded the use of a system nucleus which would reside somewhere in memory at all times, thus making an appreciable portion of storage unavailable to executing programs. To overcome this drawback, it was decided to reserve only a small area of upper storage which would contain a communication area as well as a small exit routine, and input-output subroutines when required. The second basic limitation, that of restricting the basic system to two tape drives, meant that the system tape would also have to be used for scratch purposes. Although this is highly undesirable, it was found that the use of special tape routines, included in the system's input-output subroutines, could insure against inadvertently writing over the systems programs. Furthermore, if the executing programs make use of these input-output subroutines, situations are prevented wherein a program continues to read additional input, mistakenly penetrating into the next job. Another restriction posed by a system such as this is the necessity of a loading routine residing in storage when the loading of a program to be executed occurs. Such a routine is necessary when execution follows assembly and/or when input data is to be listed.

As a result of these considerations, it was found that only relatively small areas in upper and lower storage were required for systems purposes. Thus, programs written under BLNSYS for a 4K machine must not be origined in storage below location 333 (normal read, punch and print areas are available for data as under any other circumstances) or above location 3859 (3399 when the input-output subroutines are used).

The upper 100 positions of this reserved area (3900 to 3999) are occupied by a communication system which contains indicators for interrogation by the system monitor. When these indicators are properly set by the monitor, action is instigated which clears the monitor from storage while another system program is operating. The remaining reserved portion of upper memory (3860 to 3899) contains a short exit routine which allows the system program currently being used to call the monitor back into storage where it can continue to reference other system programs for a given job, or, upon completion of a job, start the next job. This general exit routine is also used by the machine operator in the event of a hangup. Since the 1401 stops completely as the result of hangup, execution of the batch is continued by simply transferring to the exit routine, which calls in the monitor, which in turn begins the next job. Since this is the case, it is necessary for programs written under BLNSYS to transfer to the exit routine rather than halting upon completion of execution. The reserved area in lower storage (001 to 332) contains the loading routine, the purpose of which is described above.

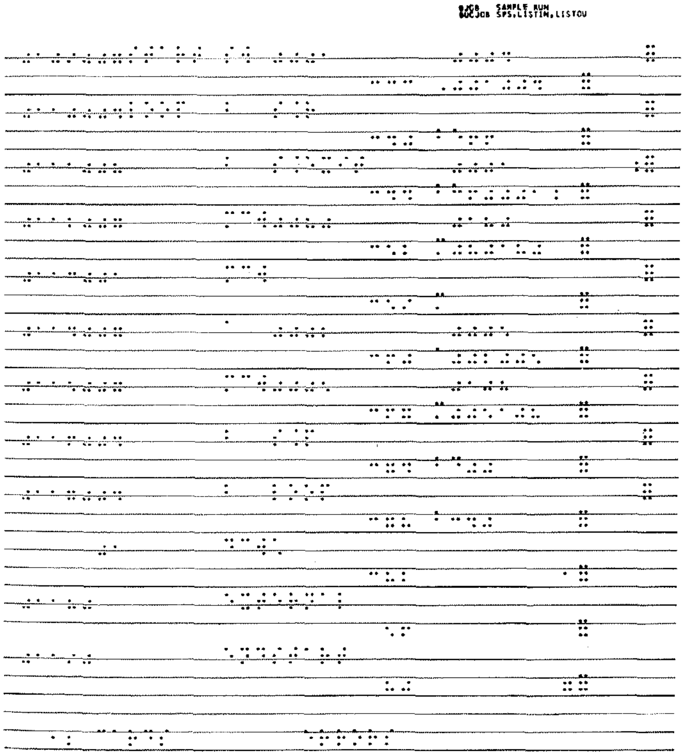
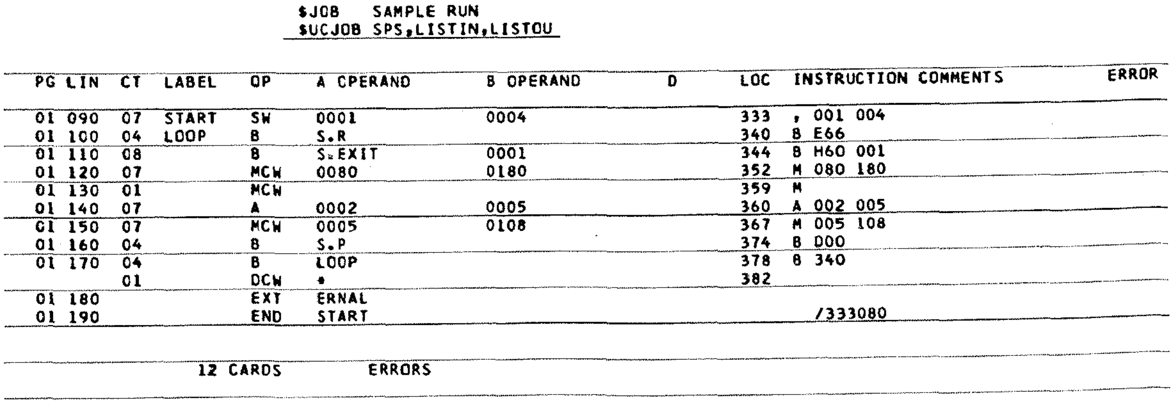


FIG. 1. Post list in braille

System Components

Because of size limitations, BLNSYS was written specifically using SPS as the source language. Users preparing



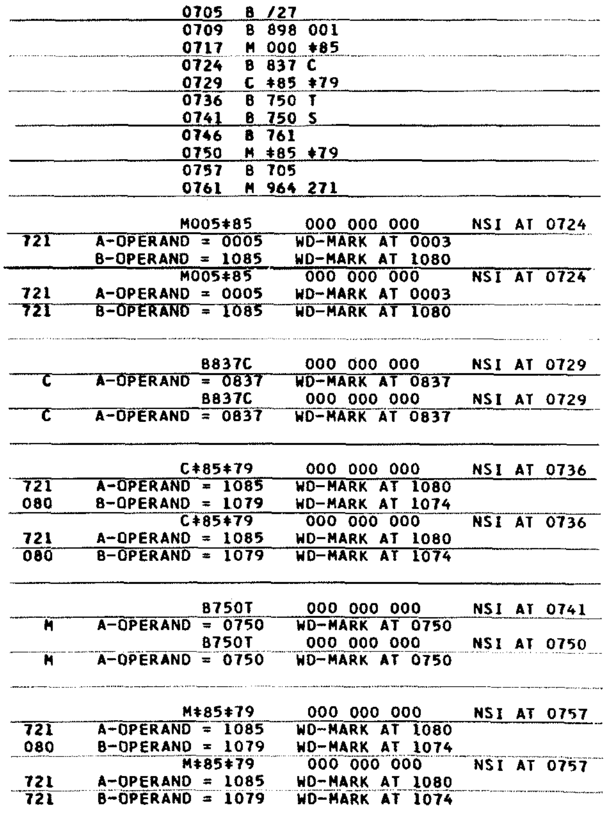
```
                         $JOB    SAMPLE RUN
                         $UCJOB SPS,LISTIN,LISTOU

  PG LIN  CT  LABEL   OP    A CPERAND        B OPERAND          D        LOC  INSTRUCTION COMMENTS          ERROR

  01 090  07  START   SW    0001             0004                        333  , 001 004
  01 100  04  LOOP    B     S.R                                          340  B E66
  01 110  08          B     S.EXIT           0001                        344  B H60 001
  01 120  07          MCW   0080             0180                        352  M 080 180
  01 130  01          MCW                                                359  M
  01 140  07          A     0002             0005                        360  A 002 005
  C1 150  07          MCW   0005             0108                        367  M 005 108
  01 160  04          B     S.P                                          374  B D00
  01 170  04          B     LOOP                                         378  B 340
          01          DCW   *                                            382
  01 180              EXT   ERNAL
  01 190              END   START                                              /333080


                12 CARDS          ERRORS
```

FIG. 2. Post list in English counterpart

programs under this system have the following routines available to them.

1. *Brailler.* The properties and technique by which a high-speed printer can be made to emboss regular printing paper with braille symbols is described in Sterling et al. [1]. The BLNSYS routine performs a character-for-character conversion of alphameric symbols into their braille equivalent, using grade one braille. As such, it may be called directly by the user who wishes to produce a braille listing of some input cards, or it may be called automatically by the system to produce a program listing, core dump, etc. The brailler is basic to the system in that it is called in automatically when printed output is to be produced. A sense switch (sense switch F is used) must be turned on to prevent translation into braille and retain the original English printout.

2. *Assembly Package.* This set of programs takes a source deck and, depending on the options specified by the user, produces a post list in English or braille and a condensed or uncondensed object deck. Although the use of a certain portion of upper memory is restricted as described above, the assembler will produce object decks for programs occupying as many as 16,000 positions of memory. It should be noted here that although the object deck produced with this assembler is basically the same as that produced by the standard IBM SPS Assembly Program, system considerations have necessitated some changes in the post list and condenser programs so that object decks produced via BLNSYS cannot be post listed



```
                    0705  B /27
                    0709  B 898 001
                    0717  M 000 ‡85
                    0724  B 837 C
                    0729  C ‡85 ‡79
                    0736  B 750 T
                    0741  B 750 S
                    0746  B 761
                    0750  M ‡85 ‡79
                    0757  B 705
                    0761  M 964 271

                    M005‡85      000 000 000      NSI AT 0724
 721     A-OPERAND = 0005     WD-MARK AT 0003
         B-OPERAND = 1085     WD-MARK AT 1080
                    M005‡85      000 000 000      NSI AT 0724
 721     A-OPERAND = 0005     WD-MARK AT 0003
 721     B-OPERAND = 1085     WD-MARK AT 1080


                      B837C      000 000 000      NSI AT 0729
   C     A-OPERAND = 0837     WD-MARK AT 0837
                      B837C      000 000 000      NSI AT 0729
   C     A-OPERAND = 0837     WD-MARK AT 0837


                    C‡85‡79      000 000 000      NSI AT 0736
 721     A-OPERAND = 1085     WD-MARK AT 1080
 080     B-OPERAND = 1079     WD-MARK AT 1074
                    C‡85‡79      000 000 000      NSI AT 0736
 721     A-OPERAND = 1085     WD-MARK AT 1080
 080     B-OPERAND = 1079     WD-MARK AT 1074


                      B750T      000 000 000      NSI AT 0741
   M     A-OPERAND = 0750     WD-MARK AT 0750
                      B750T      000 000 000      NSI AT 0750
   M     A-OPERAND = 0750     WD-MARK AT 0750


                    M‡85‡79      000 000 000      NSI AT 0757
 721     A-OPERAND = 1085     WD-MARK AT 1080
 080     B-OPERAND = 1079     WD-MARK AT 1074
                    M‡85‡79      000 000 000      NSI AT 0757
 721     A-OPERAND = 1085     WD-MARK AT 1080
 721     B-OPERAND = 1079     WD-MARK AT 1074
```

FIG. 3. Sample printout generated by trace program

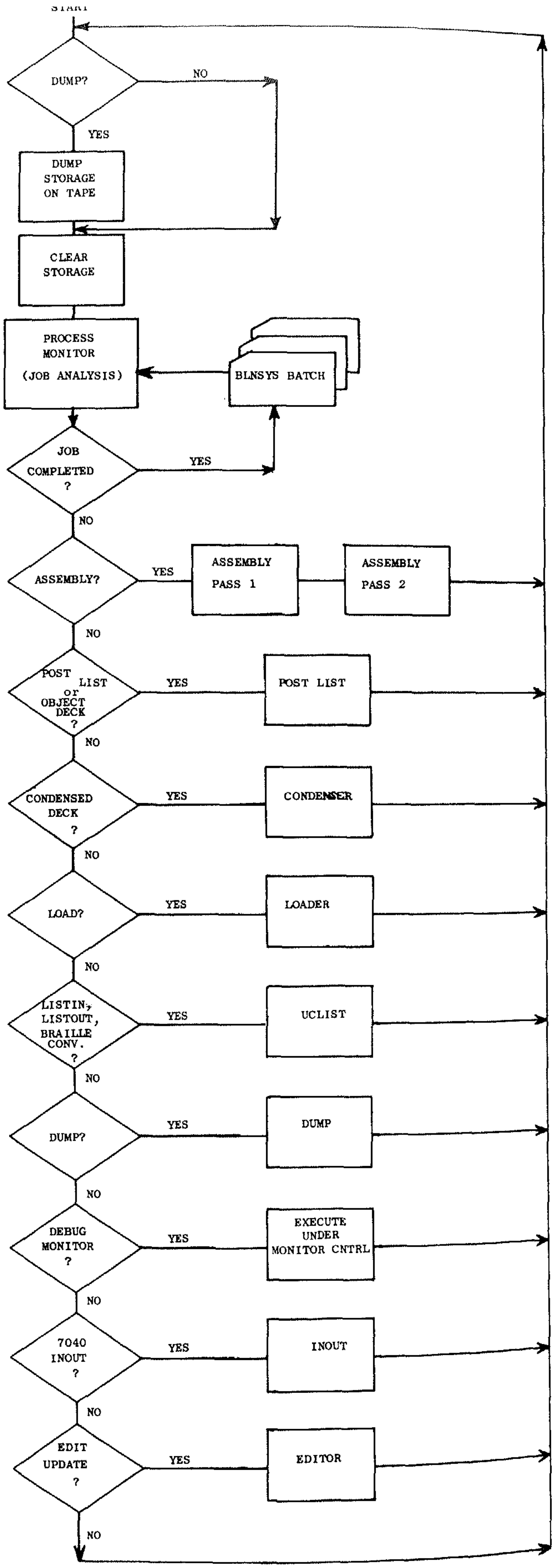


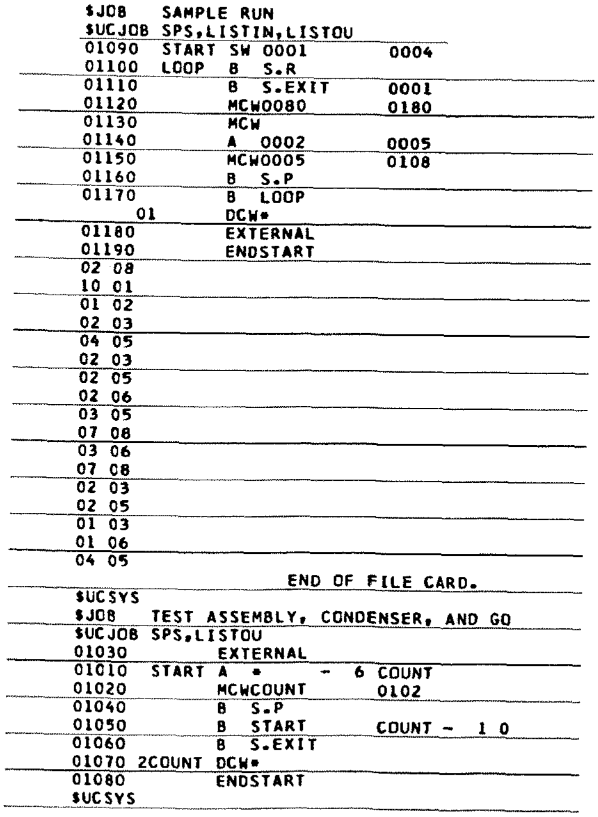
FIG. 4. Flow of processing under BLNSYS control

or condensed with IBM programs. Figure 1 shows a portion of a BLNSYS post list in braille and Figure 2 shows its counterpart in English.

3. *Debugging Monitor (Trace Routine).* This type of program, described in detail in [3], is included in the system to provide a valuable debugging aid. When called by the user, the debugging monitor will assume control of execution and step through the processing sequence one instruction at a time, printing the contents of the appropriate registers before and after execution of each instruction. Capability is included for specifying portions of a program over which this mode of operation is to extend. Figure 3 shows a sample printout generated by this trace program.

4. *Input-Output Subroutines.* These subroutines are available to the programmer writing under BLNSYS and may be used to handle read, write, punch, control carriage and all tape instructions. Although these are uneconomical in terms of storage usage (upper limit for users' programs is 3399), they do provide assurance that the integrity of the system will be maintained during the operation of all programs.

5. *Housekeeping Programs.* In addition to a loader and monitor, BLNSYS contains a dump program which may be called by the user at object time resulting in a storage dump when execution of users program is completed, an exit routine which may be called by a program or by the operator during execution, and an edit routine for updating, modifying, expanding or rewriting the systems tape itself. A simplified flowchart of the processing of jobs under BLNSYS control is shown in Figure 4.



```
$JOB     SAMPLE RUN
$UCJOB   SPS,LISTIN,LISTOU
01090    START SW 0001          0004
01100    LOOP  B  S.R
01110          B  S.EXIT        0001
01120          MCW0080          0180
01130          MCW
01140          A  0002          0005
01150          MCW0005          0108
01160          B  S.P
01170          B  LOOP
     01        DCW*
01180          EXTERNAL
01190          ENDSTART
02 08
10 01
01 02
02 03
04 05
02 03
02 05
02 06
03 05
07 08
03 06
07 08
02 03
02 05
01 03
01 06
04 05
                    END OF FILE CARD.
$UCSYS
$JOB     TEST ASSEMBLY, CONDENSER, AND GO
$UCJOB   SPS,LISTOU
01030          EXTERNAL
01010    START A  *      -  6 COUNT
01020          MCWCOUNT       0102
01040          B  S.P
01050          B  START       COUNT -  1 0
01060          B  S.EXIT
01070 2COUNT   DCW*
01080          ENDSTART
$UCSYS
```

FIG. 5. Sample small batch setup

Preparation of BLNSYS Batches

As is the case with IBSYS, the IBM operating system after which BLNSYS is patterned, cards containing instructions to the system are characterized by a dollar sign in column 1. Various options are available to the user, who may specify them in terms of instructions to the system. These may be summarized as follows:

1. Assemble, post list, produce condensed or uncondensed object deck, and execute. Any of these options may, of course, be bypassed in a given job.
2. If a program is submitted for execution, input to that program may be listed on the printer. A similar request can be made for output which under normal circumstances would be punched only.
3. The user may request a dump at any given point.
4. Any program submitted for execution as an uncondensed object deck can automatically be processed via the single instruction trace program either in its entirety or over a range of instructions specified by the user.
5. Since this installation also includes an IBM 7040, it was found expedient to add routines which produce a braille printout of IBSYS output. Thus it is possible for the blind programmer writing in FORTRAN, COBOL, or MAP to receive a braille printout of his listings, diagnostics, and/or output.

A typical setup for a small batch is shown in Figure 5.

Experience with this system has shown it to be a sizable convenience for both blind and sighted programmers. In a number of instances assembly and listing procedures required less time than with programs previously used for those purposes.

Similar systems for other computers may be developed for institutions employing the blind. In most cases, such system work can be done by the blind programmer himself, or in cases where the work becomes too complex, through the ACM Committee on Professional Activities of the Blind. Further information on the BLNSYS system and its use may be obtained from the authors or through the ACM Committee on Professional Activities of the Blind, Medical Computing Center, College of Medicine, Eden and Bethesda Avenues, Cincinnati, Ohio.

Acknowledgments. The valuable help of Dr. Theodor D. Sterling, Mr. Robert Walton and Mrs. Diane M. Stuebing in the preparation of this system is gratefully acknowledged.

RECEIVED JANUARY, 1965

REFERENCES

1. STERLING, T. D., LICHSTEIN, M., SCARPINO, F., STUEBING, D., AND STUEBING, W. Professional computer work for the blind. *Comm. ACM 7*, 4 (Apr. 1964), 228.
2. ——, ——, ——, ——, AND ——. Computer work for the blind. *J. Rehabil. 30*, 6 (Nov.-Dec. 1964).
3. ——, AND POLLACK, S. V. MEDCOMP, part I: statistical systems. Med. Comput. Ctr., Coll. of Medicine, U. of Cincinnati, Cincinnati, Ohio.