AAEC/E449

AAEC/E449

3

AUSTRALIAN ATOMIC ENERGY COMMISSION

RESEARCH ESTABLISHMENT

LUCAS HEIGHTS

BIBLIO - A BIBLIOGRAPHIC INDEX SYSTEM

by

R.J. CAWLEY
J.R. BIRD

October 1978

ISBN 0 642 59653 0

AUSTRALIAN ATOMIC ENERGY COMMISSION
RESEARCH ESTABLISHMENT
LUCAS HEIGHTS

BIBLIO - A BIBLIOGRAPHIC INDEX SYSTEM

by

R.J. CAWLEY
J.R. BIRD

ABSTRACT

A general purpose system for recording and searching an index of information, using interactive computer techniques, is described. Normal language nomenclature is used and searches can be made on any character or combination of characters within specified fields of information. The use of the system is illustrated with examples from several applications.

National Library of Australia card number and ISBN 0 642 59653 0

The following descriptors have been selected from the INIS Thesaurus to describe the subject content of this report for information retrieval purposes. For further details please refer to IAEA-INIS-12 (INIS: Manual for Indexing) and IAEA-INIS-13 (INIS: Thesaurus) published in Vienna by the International Atomic Energy Agency.

INFORMATION RETRIEVAL; COMPUTER CODES

CONTENTS

1. INTRODUCTION

It is debatable whether the information explosion of recent times has led to the development of computer-based systems for storing, retrieving and distributing information, or whether the availability of such systems has encouraged the rapid growth in published information. Whichever is the case the emphasis has, quite naturally, been on systems to cope with large collections of abstract or numerical data. Thus most literature-abstracting and profiling systems, such as Information Service in Physics, Electrical and Electronics Engineering, Computer and Control Engineering (INSPEC), International Nuclear Information System (INIS), *etc*., are designed to select, from a large range of published material, those items which are relevant to the interests of an individual or a group. A separate problem arises from the fact that the output of such profiling systems, and the consequent increase in personal reading, may place a considerable strain on a person's ability to file and retrieve within his or her own limited but not necessarily small collection. This report describes a do-it-yourself bibliographic system which has been developed to solve the latter problem.

A personal file must be flexible and versatile enough to cope with developing and changing interests. It should be able to produce information on a reasonable time scale - within minutes for simple queries and within hours for extensive listings or compilations of filed information. It should also overcome the major drawback of conventional filing methods, namely the difficulty in providing adequate cross-references or multiple copies of material which relates to more than one file heading. These requirements can be met by computer equipment which provides for interactive operations.

BIBLIO was first developed as a bibliography of the rapidly expanding uses of 'Prompt Nuclear Analysis' techniques [Bird, Campbell & Cawley 1977]. It has since been applied to such areas as 'Alternative Energy, Australia' and 'Radioisotopes in Environmental Studies'. Some examples from these are given in Section 6 to illustrate its use; the two BIBLIO programs, BIBIN and BIBSORT, are described in Sections 2 to 5.

2. GENERAL DESCRIPTION

BIBLIO is a system for recording and searching an index of information culled from any source. It is designed to accept a flexible, normal-language nomenclature and to provide a quite general search capability. Entries may contain bibliographic information or numerical data, or both. Of course, nothing can be found which is not initially entered into the catalogue, but searches are not restricted to any special set of keywords. Any character or combination of characters within a specified information field can be indexed. Searches can also be made with AND/OR characteristics applying to several such fields. BIBLIO is thus very adaptable to the changing requirements of a personal filing system or the sharing of files by a special interest group. Nevertheless, it is equally suitable for more formalised or more general applications.

BIBLIO was developed to take advantage of the interactive operation possible with the Lucas Heights Dataway system [Ellis 1970, Richardson 1969] and associated communications software [Cawley & Trimble 1977]. Files may be entered, recalled, edited or deleted in a conversational manner. However, these operations could equally well be carried out in the normal computer batch job stream. The latter method is used at present for searches, since these can result in lengthy output operations.

BIBLIO consists of two separate programs - BIBIN and BIBSORT. These programs are written in FORTRAN, but all input/output (I/O) operations are performed by ASSEMBLY language routines and none of the FORTRAN run time library routines are referenced. The user modifies the tables which define the information fields, title blocks, abbreviations and maximum field sizes and then compiles and links the two programs, giving them characteristic names. From then on, the two programs are operated independently on a personal or group file.

The BIBIN program is used to input information to the catalogue and communicate with the user at any terminal attached to the Dataway network. This program may also be used to edit and modify existing catalogue information.

The BIBSORT program operates in batch mode and is used to search the catalogue and sort information alphabetically before printing. (The sorting order follows the collating order of the EBCDIC code.)

Both programs keep the catalogue information in a library but an internal text buffer is used for operations involving only one file.

2.1 Text Buffer

The code has an in-core text buffer which is used to build a library file, or to contain a library file image when listing or editing. The text buffer is divided into up to 20 separately addressable fields. The total number of characters which may be held in the text buffer is 5000 minus four times the number of fields. A maximum number of characters is set for each individual field and a warning is issued when the number is exceeded. This can be useful to guard against entering information in a wrong field.

2.2 Library

The catalogue information is stored in a direct access disk library which has a fixed blocksize of 5000 bytes. The first block of the dataset contains three directories. Each directory is a bit map with one bit corresponding to each block of the library dataset. The first directory shows which blocks have been allocated to store information. The two remaining directories indicate that certain catalogue files have been marked with either of the possible special attributes (see Section 2.3).

The remainder of the dataset is used to store catalogue information, one file per block. Thus, catalogue file numbers correspond to the sequential block of the library.

Since the directory size is limited to one third of the first block, and each byte contains eight bits, the maximum number of files which may be stored is (5000/3)*8=13328. The code, however, has a limit of 12 800 files. In practice, the user will want far less files and, as disk space for the entire library must be pre-allocated, the user would set the number of allowable files to a lower value.

The library limitations can be summarised as follows:

Maximum number of files	12 800
Maximum number of fields per file	20
Maximum number of characters per file	5000-4N

where N is the number of fields.

2.3 File Attributes

Files may have up to two attributes. The significance of these attributes is only of meaning to the user but the BIBSORT program can be commanded to exclude any file with a particular attribute from its selection list. The two attributes are selected by the code letters C and X. The present implementation uses these two attributes as follows:

C   means that a copy of the publication is on hand.

X   means that the publication is relatively unimportant.

The character U, used as an argument for the MARK command, causes the nominated file to be unmarked, *i.e.* it is not marked by attributes.

3. HOW TO RUN BIBIN

The BIBIN program is available at the AAEC Research Establishment as a catalogued procedure; it requires 60 K bytes of core storage, runs as a CLASS I interactive job, and is invoked by the job control language statement:

```
//   EXEC   BIBIN,LIB='USR.BIBLIB',
//          SLIB='USR.PROGLIB'
```

The symbolic parameters available and their meanings are as follows:

SLIB=   defines the name of the dataset in which the BIBIN program may be found.

LIB=   defines the name of the library dataset.

DISP=   defines the library dataset disposition status. Default is DISP=OLD, normally used except when creating a new library dataset, when DISP=NEW should be specified.

FILES=   defines the size of the library dataset for the operating system. This should be set to the number of files the user wishes the library to hold. The default value is FILES=500.

REG=   defines the amount of core storage the operating system is to allocate to run the program. The default value is REG=80K.

PRG=   defines the program name to be run. The default is PRG=BIBIN. This may need to be changed if a non-standard version of the program is to be used.

The mechanics of commencing terminal communication with the program are described in Appendix A.

3.1 BIBIN Program Control Commands

Each of these keywords instructs the program to perform a particular task. The individual commands, allowed abbreviations and their meanings are now described:

Command	Abbrev.	Example of use	Description
INPUT	IN	IN	signals the code to clear the text buffer and begin accepting data to form a new text buffer. A full description of the input procedure is given in Section 3.2.
SAVE	SA	SA	saves the current contents of the text buffer as a new file. The file number assigned is printed at the terminal.
LIST	L	L	lists the current contents of the text buffer at the terminal.
END	E	E	is the final keyword and closes down the terminal and terminates execution of the program.
SUSPEND	SUSP	SUSP	suspends operation of the program and the terminal, thus freeing the terminal for other uses. The user may reactivate and carry on the program with no loss of the stored information by logging on again at any terminal. A time limit of five minutes is imposed, after which time the program will terminate if the user has not logged on again.
CRMASTER	CRM	CRM	is the keyword used initially to format a new library dataset and directory. This keyword should only be used when creating a new library. If used with an existing library, all files are deleted.
LISTFILE	LF	LF 450	the text buffer is loaded from the nominated (450) library file and the contents are then printed at the terminal.
DELETE	DEL	DEL 99	the nominated (99) library file is deleted.
MARK	M	M 143 X	the nominated (143) library file is marked in the nominated (X) manner in the auxiliary directory.
LISTMAST	LM	LM	the entire library contents are listed on the line printer file by file (unsorted).
EDIT	ED	ED 49	the text buffer is loaded from the nominated (49) library file and then the edit subroutine is entered. The edit operations are described in Section 3.3.
FIND	F	F AU /ABEL/	searches the library checking the specified field (Authors) for the specified string (ABEL) - see Section 3.4.
HELP	H	H	a list of keywords and other helpful information is printed at the terminal

3.2 BIBIN Input Procedure

After receiving the INPUT command, the program will prompt the user for input to the text buffer by printing the name of the first field. The user should type the information to be placed in this field and terminate the line with a carriage return. The program will prompt again for this field by printing

```
CONT:
```

If a further line of information is to be input, the user types it and again terminates with a carriage return. When all lines of information for this field have been entered, the user types an immediate carriage return to the CONT: prompt; the program then prompts for the next field. This process continues until all fields have been entered. A field may be skipped by typing an immediate carriage return to the initial prompt for the field, and all succeeding fields may be skipped by typing a single space followed by a carriage return to the initial prompt for a field. Excessive spaces are removed from the input text so that a maximum of one space may appear between words. The fields for the standard BIBIN program are described in Section 6.1.

When all fields have been entered (or skipped), the program prompts

```
SAVE ?
```

The text buffer is saved as a new library file unless the user responds NO to this prompt.

The following example illustrates this process. (Underlined words here and in other examples are those printed by the program.):

```
OPTION:   IN
CODE:   BLO 72
CONT:
AUTHORS:   BLOGGS,J.J.,JONES,R.P.
CONT:
REFERENCES:   JOURNAL OF RUBBISH
CONT:
ABSTRACT NOS:
SAVE ?   YES
SAVED AS PSEUDO FILE 123
```

The user responded with a space and carriage return to the ABSTRACT NOS prompt thus skipping all subsequent fields.

3.3 BIBIN Edit Procedure

When the EDIT command is received, the program loads the text buffer from the nominated library file and then enters 'edit mode'. In this mode the information in any field may be altered or replaced. The subcommands available in this mode are:

Command	Abbrev.	Example of use	Description
EDIT	ED	ED AU	selects the author field for editing.
REPLACE	R	R CODE	selects the code field to be replaced.
SAVE	SA	SA	replaces the original library file with the edited version currently in the text buffer and then leaves edit mode.
EXIT	EX	EX	leaves edit mode.

EDIT subcommand

After receiving the EDIT subcommand, the program selects the nominated field for alteration and prompts the user by printing CHANGE. The user then types in the old and new character strings delimited by some character which is not present in either string. The program replaces the old string by the new string and prints the new contents of the selected field. The CHANGE promp is printed again and the user may request further changes. When no further changes to this field are required, the user types an immediate carriage return to the change prompt. The program will then prompt another EDIT subcommand. The following example illustrates this routine:

```
OPTION:   EDIT 10
EDIT OPTION:   ED AU
CHANGE   /SMITH/SMYTH/
SMYTH,J.P., BLOGGS,K.,JONES,J.N.
CHANGE
EDIT OPTION:
```

In this case the author field is selected for alteration and the word SMITH is changed to SMYTH. The character / is the delimiter and serves only to identify the beginning and end of the two strings. Any character not present in either string may be used for this purpose (excluding the space character).

REPLACE subcommand

After receiving this subcommand, the program prompts the user for input by printing the name of the selected field. The user then types in new information for this field as if the INPUT command had been given. Only this field is requested and the information typed in completely replaces any previous information. Example:

```
EDIT OPTION:   R BEAM
BEAM:   2.5 MEV PULSED
CONT:
EDIT OPTION:
```

3.4 BIBIN FIND Command

The FIND command is used to locate files with particular information. The contents of the located files are printed at the terminal. The command takes the general form:

```
FIND FIELD STRING OPERATOR FIELD STRING
```

where the term FIELD refers to one of the addressable fields discussed in Section 2.1, the term OPERATOR refers to one of the logical operators "AND", "OR" and "NOT", which are typed as such, and the term STRING refers to a character string enclosed in delimiter characters. The delimiter character may be any non-blank character not present in the string. As many recurrences of the portion

```
OPERATOR FIELD STRING
```

may appear as will fit on one input line. Example:

```
OPTION:  F AU /BROWN/ OR AU /SMITH/
```

This will find and list all files in which the author field contains the character strings BROWN or SMITH. Note that the / character denotes the beginning and end of the strings. If an excessive number of files is found, then boring printout can be interrupted by typing the question mark (?) character at the terminal. Printout will cease after printing the entire contents of the current file.

The execution of this command may take a few minutes, depending on the number of allocated files in the library.

4. HOW TO RUN BIBSORT

The BIBSORT program is available at the AAEC Research Establishment as a catalogued procedure, requires 120 K bytes of core storage, and is invoked by the job control language statement:

```
//     EXEC    BIBSORT,LIB='USR.BIBLIB',
//                SLIB='USR.PROGLIB'
```

The symbolic parameters available and their meanings are as follows:

SLIB= defines the name of the dataset in which the BIBSORT program may be found.

LIB= defines the name of the library dataset.

FILES= allocates disk space for the sort and should equal the maximum number of files expected in any one sort. The default value is FILE=500.

SBF= defines the number of I/O buffers to be allocated to each of the sort/merge datasets. The default value is SBF=2 and, should this be increased, more core storage area will need to be allocated to run the program.

REG= defines the amount of core storage the operating system is to allocate to run the program. The default value is REG=120K.

PRG= defines the name of the program to be run. The default is PRG=BIBSORT. This parameter should only be specified if a non-standard version of the pro gram is to be run.

The BIBSORT program runs as a batch job and takes its commands from the SYSIN dataset. Commands are typed in free format, one command per card. Cards which have the character * in column 1 appear on the output listing as comments and are ignored by the program. The output will appear on the BIBPRINT dataset unless the AUXOUT command is given, in which case the output will appear on the AUXOUT dataset in an abbreviated form.

4.1 BIBSORT Program Control Commands

Some of these commands are passive and some active. The passive commands set flags within the program to control actions taken on receipt of the active commands. The active commands instruct the program to perform a particular task. The individual commands are as follows:

Command	Abbrev.	Example of use	Description
FIELDS	F	F C AU	selects fields to be used to sort files in order of significance (passive).
PRINT	PR	PR AU REAC	selects fields to be printed on output (passive).
PAGE	PG	PG 72	selects output page width in columns (passive).
MASK	M	M X	selects the attribute mask (passive).
AUXOUT	AUXOUT	AUXOUT	switches the output unit to the auxiliary output file (passive).
SELECT	SE	SE AU /RUBIN/	selects files whose author field contains the character string RUBIN and places their file numbers in the selection list (active).
ADD	A	A REF /AMC.SO/	adds to the selection list any files whose reference field contains the character string AMC.SO (active).
AND	AND	AND AU /CAMP/	performs a logical "AND" of the selection list with those files whose author field contains the character string CAMP, and retains in the list only those files which contain the character string CAMP (active).
NOT	NOT	NOT AU /RUS/	removes from the selection list any file whose author field contains the character string RUS (active).
SORT	SO	SO	sorts the files in the selection list according to the field order set up by the FIELDS command and then prints out information from the sorted files according to the options set up by the PRINT command (active).

4.2 BIBSORT Field Selection

The commands SELECT, ADD, AND, and NOT take the general form:

COMMAND SELECTOR ARGUMENT

The BIBSORT program has two lists of keywords for matching the selector, namely the 'primary' and 'secondary' lists. The primary list contains special purpose keywords. A particular action is taken if a match is made to a keyword in the primary list. After receiving the command, the primary list is scanned first and the secondary list is only scanned if no match is found. The current implementation has the following keywords in the primary list:

Keyword	Abbrev.	Description
ALL	A	all files in the library are placed in the selection list. The argument, if present, is ignored.
REACTIONS	REAC	the reaction of each file is tested for the criterion given in the argument (see below) and files which satisfy this criterion are placed in the selection list.
STREACTIONS	STREAC	this has the same result as if the REACTIONS keyword were matched in the secondary list.

The secondary list contains the field identifiers described in Section 6.1. When the selector is matched by a keyword in the secondary list, the argument field of the command takes the following form:

/STRING/

where the / character is a delimiter and serves only to identify the beginning and end of the STRING. (Any character not present in the STRING may be used for this purpose.) The field selected is searched in each library file and, if the character string STRING is found, then that file is placed in the selection list.

If the REACTIONS keyword is selected in the primary list, then a special subroutine is entered. This subroutine searches the REACTIONS field of all files for a match to the special criteria shown in the argument field of the command. If these criteria are satisfied, then the file is placed in the selection list. The argument field here takes a special form as follows:

```
A,B,C(D,E)
```

where A, B, and C are 'OR' conditions and (D,E) is an 'AND' condition. Any number of OR conditions may be specified, but only one AND condition. If no OR conditions are specified, the AND condition becomes the sole criterion for selection. If no AND condition is specified, then satisfying any of the OR conditions is sufficient for selection. For example, A, B, and C may be element symbols and D and E may be symbols which describe a reaction type. The strings A, B, C and (D, E) each have a maximum length of eight characters.

5. PROGRAM MODIFICATION

Both programs are table driven in the area of field definition and I/O paths. A user who wishes to modify the field list thus needs only to alter the appropriate tables in each program and reassemble the programs. The tables are held in COMMON areas set up by a single BLOCK DATA subprogram in each program.

Users who wish to have special purpose select keywords and actions will need to modify the MAINP subroutine of the BIBSORT program accordingly, and provide any special subroutines required.

6. EXAMPLES

6.1 Prompt Nuclear Analysis Bibliography

This first application uses the 15 information fields listed in the following table. The user addresses these fields by name or abbreviated name:

Field	Abbrev.	Usage
CODE	C	code for this file.
AUTHORS	AU	publication, authors and institutions from whence they come.
REFERENCES	REFS	publication references.
ABSN	ABSN	abstract numbers.
ABSTRACT	ABS	short description of publication contents, *e.g.* title or abstract.
REACTIONS	REAC	description of reactions covered by the publication.
DESC	D	description of type of experiment.

continued

Field	Abbrev.	Usage
SAMPLE	S	description of target samples.
BEAM	B	description of incident beam.
RAD	RAD	description of product radiation detected.
METHOD	M	description of experimental method.
RESULTS	RES	significant experimental results.
REF.DATA	RD	reference data measured.
APPLIC	AP	field of application.
SRES	SR	cross reference to other publications describing the same results.

The first five fields are typical of those used in any abstracting system, but the remaining fields are used to file information on the methods and results reported in relevant references. A typical file is given below:

```
ABE 72B
ABEL,F.,AMSEL,G.,BRUNEAUX,C.,COHEN,C.,MAUREL,B.,RIGO,O.S.,ROUSSEL,J.
INT.CONF. ON MODERN TRENDS IN ACTIVATION ANALYSIS; SACLAY, CONF-
721010(1972);J.RADIONAL.CHEM.16(1973)587
NSA19438,INIS 174640;CA 30/5948
NUCLEAR MICROANALYSIS USING MEV CARBON ION BACKSCATTERING;USEFULNESS
AND APPLICATIONS.
SI,FE,SB,YB,AU,HG,PB(4HE,BS) & (12C,BS)
```

The format used in each field is entirely up to the user, but it must be standardised if searches are to be successful. However, new symbolism can be introduced, as the need arises, to distinguish features of new publications which had not been important in previous entries. Likewise, temporary information can be included, such as a symbol to identify entries which may require further information. A search on this symbol then produces a list of such entries. The symbol is removed when the entry is complete. The following examples illustrate the results of various types of searches:

<u>The ^{14}N(d,p)^{15}N reaction</u>

```
SELECT REAC 14N(D,P)
PRINT C
SORT
```

```
AMS 68A *, AMS 69B *, AMS 69C *,  AMT 75, BER 72 *, CHA 75,
COO75, CRO 70A, GOL 74A, JON 72 *, OLI 72 *, OLI 75A, OLI 75B,
OLI 76B, PEI 66A *, PEI 67A *, PEI 68A *, PIE 76, QUA 73,
RIG 76, SUN 74A, SUN 74B, SUN 75, SUN 76, THO 74A, WEB 72 *,
YAT 73.
```

This list is an index of the publications which report on the use of the ^{14}N(d,p)^{15}N reaction. The inclusion of additional fields following "C" in the PRINT statement would have produced a listing of filed information from these publications.

Author profile

```
FIELDS AU
SELECT AU "COHEN,B.L."
PRINT AU REFS
SORT

COHEN,B.L.,FINK,C.L.,DEGNAN,J.H.
J.APPL.PHYS.43(1972)19

COHEN,B.L.,MOYER,R.A.
ANAL.CHEM.43(1971)123

SHABASON,L.,COHEN,B.L.
ANAL.CHEM.45(1973)284
```

In this way, a list of publications is obtained for which "Cohen,B.L." was author or joint author. The same information can be obtained using BIBIN with the option: F AU "COHEN.B.L."

Samples studied

```
SELECT REAC 27AL(P,G)
AND B "EP=0.99"
PRINT C S
MASK X
SORT

AMS 62 *
ANODISED AL,TA

AMS 71B *
ANODISED AL ON TA

BUT 69B
POLISHED METALS AU,PT
```

```
BUT 70 *
POLISHED METALS CR,W,MO,AG,NI,ZN,CD,ZR,NB,SN,AU,CU
DEC 72 *

DUN 72 *
AL IMPLANTED IN SIO2

DUN 73
AL2O3;AL IMPLANTED IN SIC,SIO2

DUN 74
AL2O3;AL IMPLANTED IN SIC,SIO2

DZH 71 *
AL2O3 + SIO2

LUC 75
AL IMPLANTED IN SIC

MAN 74
ALLOYS
```

This search lists the code number and details of samples studied in all work on the ^{27}Al(p,γ)^{28}Si reaction with an incident proton energy near the strong resonance at 0.992 MeV, excepting those publications which do not contain significant information.

<u>Exploratory search</u>

```
SELECT REAC 14N(P,G)
AND REFS "KARLSRUHE,1975"
PRINT C B
SORT

LAN 76E
EP=1.061

SIM 76
EP=1.061
```

In this case, the search could have arisen from the information "I think somebody reported an example of the use of a resonance in the ^{14}N(p,γ)^{15}N reaction, at the Karlsruhe Conference on Ion Beam Analysis".

6.2 Alternative Energy in Australia

Whereas the previous section illustrated a formal bibliography intended for use by others working in the field and which is, therefore, concerned with recognised publications, this example is of a personal collection of information derived from many sources, and hence uses a different version of the BIBLIO system. The first five bibliographic fields are retained for use when convenient but, for many entries, they are used simply to code unpublished information. An example of the teletype record of the entry of one file is given below:

```
CODE:   MCC 76
CONT:
AUTHORS: MCCANN,D.J.,SADLER,H.D.W.(ENERGY RES.CENTRE,SYDNEY
         UNI.)
CONT:
CONT:
REFS:   SEARCH 7(1976)17
CONT:
ABST#S:
ABSTRACT:   PHOTOBIOLOGICAL ENERGY IN AUSTRALIA.
CONT:
PROCESS:   PS(L,BM);BB(BM,ALC);BC(BL,CH4);PYR(BM,OIL)
CONT:
DESCRIPTION:  THEORETICAL ESTIMATES OF YIELD,COST,EFFICIENCY,
               ENERGY
CONT:
CONT:  INPUT
CONT:
MATERIAL:   PLANTS
CONT:
INPUT:  WOOD,BAGASSE,ORGANIC WASTES,CEREAL STRAW,CASSAVA,KENAF
CONT:
OUTPUT:  ALCOHOL,METHANE,PYROLYTIC OIL
CONT:
METHOD:  FERMENTATION,PYROLYSIS
CONT:
RESULTS:  CEREAL STRAW $8.40/T; CASSAVA $28-34/T; WOOD
          CHIPS $19/T;
CONT:
CONT:  KENAF $18-36/T; SUGAR CANE $18-23/T; ALCOHOL $8-20/GJ;
CONT:  METHANE $4.2-3.5/GJ; PYROLYTIC OIL $3.3-3.5/GJ
CONT:  CHEAPEST PS FUEL TWICE PRE-TAX COST OF FOSSIL FUEL
CONT:
```

```
SAVE ?   YES
SAVED AS PSEUDO FILE 51
```

7. CONCLUSIONS

The bibliographic system described here can be used at the AAEC Research Establishment from a teletype or visual display unit (VDU) terminal (further information is given in Appendix A). Users wishing to use the system at other computer centres will need to have the terminal communications section of the program modified to use the IBM360 operating system's TSO (Time Sharing Option) terminal input/output facilities.

Although the entry of file records can be time consuming, it is no more so than the preparation of cards or other filing operations. BIBLIO is particularly valuable when a number of people, who are interested in a particular topic, share the literature search and entry of records in order to be able to locate specific information selected or sorted in a variety of ways. The search system used in BIBLIO is especially versatile, making it possible to locate information using simple (and minimal) clues such as part of an author's name, a specific journal and year, or any word or symbol used in the relevant entries.

8. REFERENCES

Bird, J.R., Campbell, B.L. & Cawley, R.J. [1978] - AAEC/E443.

Cawley, R.J. & Trimble, G.D. [1977] - AAEC/E425.

Ellis, P.J. [1970] - AAEC/E206.

Richardson, D.J. [1969] - *Aust. Computer J.* 1 (5) 273.

Richardson, D.J. [1971] - *Aust. Computer J.* 3 (2) 55.

APPENDIX A

USE OF BIBIN AT THE AAEC RESEARCH ESTABLISHMENT LUCAS HEIGHTS

BIBIN is run as a CLASS I (Interactive) job at the AAEC Research Establishment. CLASS I jobs are automatically held by the operating system at the time they are submitted. When the user is ready to run the job, the following steps should be taken:

1. Start a HASP INTERNAL CONSOLE from the terminal
2. Issue the $RUN command to the console and give the job name or number in reply to the JOB prompt. The job will then be released to start if a CLASS I initiator is available, otherwise a message will be printed to indicate the reason for not starting the job.
3. The CONSOLE will prompt with an equals (=) sign after releasing the job. The user should respond by typing an immediate carriage return. The terminal will then remain idle until the job is ready to communicate with the user. At that stage the message:

   ```
   START BY JOB XXX
   ```

   will be printed at the terminal.
4. The user then issues the $RUN command again, this time replying to the JOB prompt with an immediate carriage return. The program will begin communicating with the user and prompt for a command by printing OPTION: at the terminal. At that stage the user must type in a command.

It should be noted that the above procedure is only applicable at Lucas Heights. Users wishing to use the BIBIN program at other computer centres will need to have the terminal communications section of the program modified to use the IBM360 operating system's TSO (Time Sharing Option) terminal input/output facilities.