Operation of a disc data base

C. O. M. Stross

Hawker Siddeley Aviation Ltd., Hatfield, Hertfordshire

A large disc data base has been in use for several years. The paper describes some of the experience gained both of day to day operation and of the support required to maintain integrity and efficiency. Details are given of the file sizes and of some typical run times.

(Received July 1972)

Engineering information files set up on disc by Hawker Siddeley Aviation Ltd. at Hatfield form the data base for a fully integrated production control system. This had its inception 9 years ago in a tape system based on the traditional breakdown file approach. When the new generation of equipment became available 5 years ago it was decided to take advantage of the large exchangeable disc to create an engineering data base. A pilot scheme was operated for a year in parallel with the tape system; this was succeeded by the full scale data base which has been in operation for $3\frac{1}{2}$ years. The new facilities have shown a considerable saving in computer time and improved customer service. For instance the gross parts explosion of a Trident 2E into 40,000 components and assemblies takes just under 2 hours compared with $10\frac{1}{2}$ hours. A retrieval of method information for 200 parts to be loaded on the shop floor took 20 minutes compared with 110 minutes. An enquiry into the work load content of five work centres was reduced to $5\frac{1}{2}$ minutes compared with 125 minutes. The foregoing examples are all for the same machine configuration.

The operation of a data base of this size which is necessarily complex in structure, can afford some problems. The experience gained in meeting and resolving some of these may be of interest.

Hardware and software

The hardware presently in use consists of an IBM System/360 Model 40 computer equipped with 128K bytes of core storage, a nine drive 2319 disc unit, six 60K byte tape drives, two card readers, two line printers, a card punch, a paper tape reader, and a console typewriter. Two Visual Display Units are situated remotely from the computer. **Fig. 1** provides a diagrammatic illustration. The system operates under DOS III in the multi-programming mode. Two batch partitions are used, and a third partition is occupied by a spooling program, GRASP, which performs the function of an output writer. The programming languages in principal use are IBM Basic Assembler, and COBOL.

The data base

The data base consists of a number of discrete but interrelated files, this is illustrated diagrammatically in **Fig. 2.** The number of files in the data base is not fixed and varies with the requirements of the systems. There are more files in the data base than there are disc drives. This is one of the major operational constraints, and has been a major factor in shaping the design of the files and in the selection of the file organisation techniques.

The data base is centred around the Bill Of Materials Processor (BOMP), an implementation package to create and maintain a central information system. Recently the BOMP package has been replaced by the Program Product DBOMP (Data Base Organisation and Maintenance Program), which while accessing the same files as BOMP features a number of improvements. Two techniques of file organisation are used by the package, Control Sequential, and Direct Access, together with a system of interfile links between records using disc address chaining. The Control Sequential access method is similar in most respects to Index Sequential. The major difference lies in the fact that extant records are not moved during file maintenance. Logical sequence in the file is maintained by address pointers (Ref. 1).

The package consists of modular programs to perform the load and maintenance function together with macros and subroutines to facilitate access to the files by the user programs. The four files so linked in the HSA data base are the Product Structure and (Manufacturing) Routing files to the Part Number Master file, and the (Manufacturing) Routing file to the Work Centre Master File. A feature of the BOMP system is that each part is uniquely described by a single record within the Part Number Master File, the part number or key does not appear on the Product Structure File, or the Routing File. These latter files only describe the relationship between one part and another, or between a part and the work centres involved in its manufacture. The reason for this is to save space. BOMP is a file organisation system designed to hold data that is organised in a tree-like structure. Such files can be used to hold the engineering structure of a product starting with the finished part, down through major assemblies, sub-assemblies, detail parts, raw materials, and tools. The Product Structure file describes the level by level breakdown and breakup relationship between parts. The (Manufacturing) Routing file describes the operations required to make or assemble the parts.

Additional files in the data base are either Indexed Sequential or Extension File System (EFS). EFS is a package developed in house, that enables the calling record in a file to be extended logically, to any size required. The package was developed both to cater for volatile files, and by taking advantage of the exchangeable disc unit to keep the size of Part Number Master file records down to reasonable proportions. Unlike the BOMP files, records in an EFS file do not carry address pointers that link back into the calling file. This fact eases some of the problems encountered when the BOMP files have to be reorganised. For example, if one file of a set is re-sequenced then any other file that contains address pointers referring to the re-sequenced file must be amended so that the address pointers remain correct. In most installations, as in HSA's the number of disc drives is a limiting factor. Any file that is linked to a file to be re-sequenced, must be on line during the re-sequencing process. It is advantageous therefore to have files which while they are directly addressable from one file (the calling file) does not contain pointers that link back into the calling file. These files do not require to be on line during any manipulation of the calling file. An additional advantage lies in the time saved through not having to adjust the address pointers in the called file. A full description of EFS is given in Appendix 1.

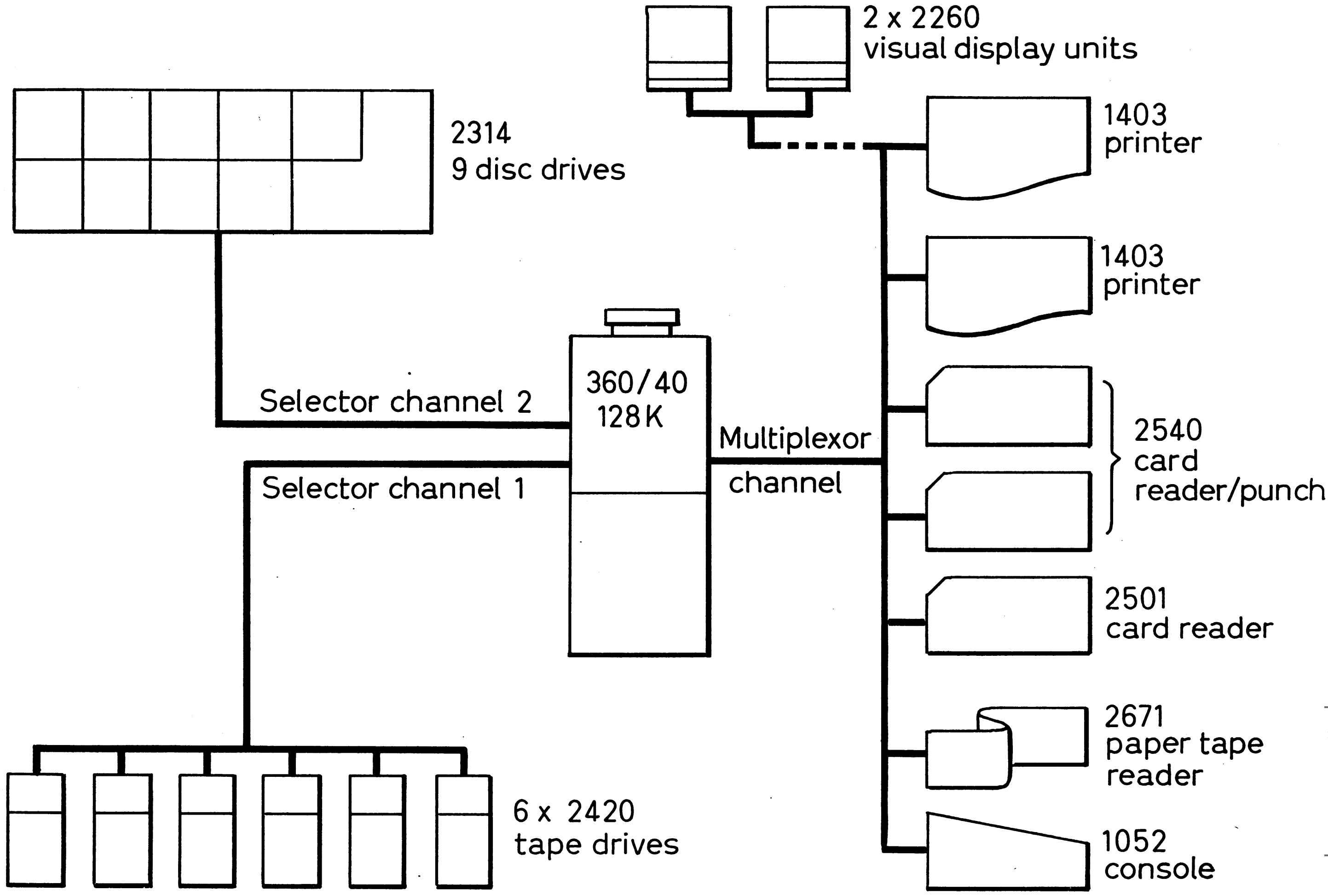
2 x 2260
visual display units
2314
9 disc drives
1403
printer
1403
printer
360/40
128K
Selector channel 2
Selector channel 1
Multiplexor
channel
2540
card
reader/punch
2501
card reader
2671
paper tape
reader
6 x 2420
tape drives
1052
console

Fig. 1.

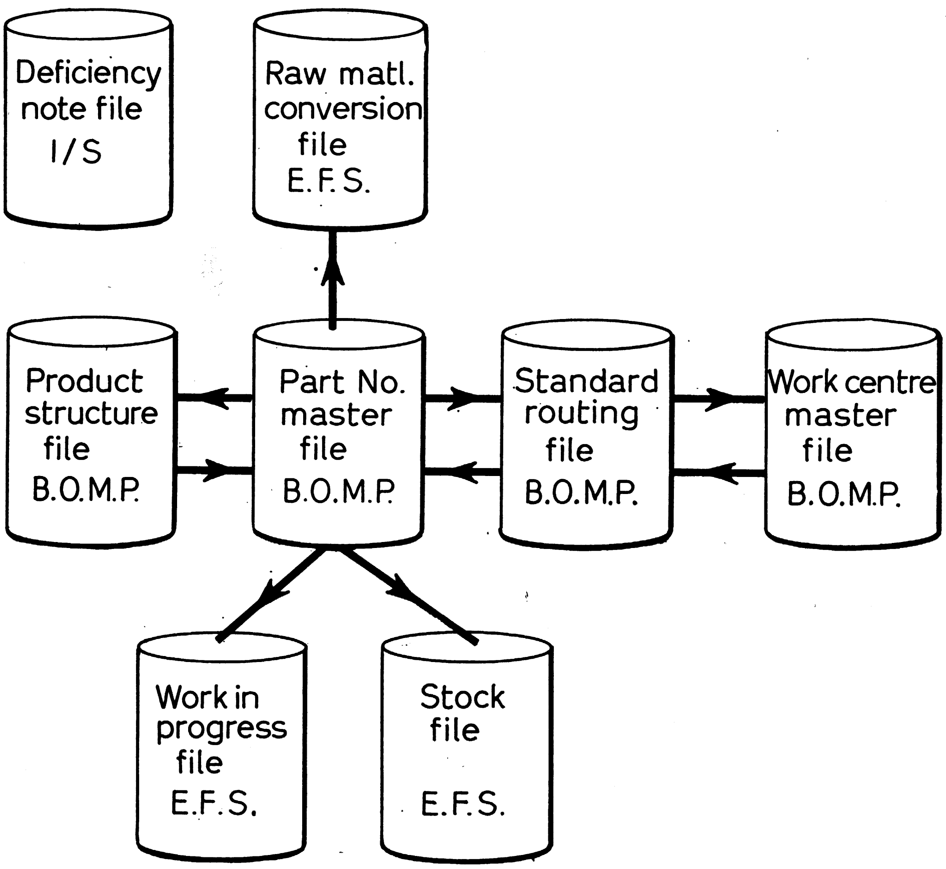
Deficiency
note file
I/S
Raw matl.
conversion
file
E.F.S.
Product
structure
file
B.O.M.P.
Part No.
master
file
B.O.M.P.
Standard
routing
file
B.O.M.P.
Work centre
master
file
B.O.M.P.
Work in
progress
file
E.F.S.
Stock
file
E.F.S.

Fig. 2.

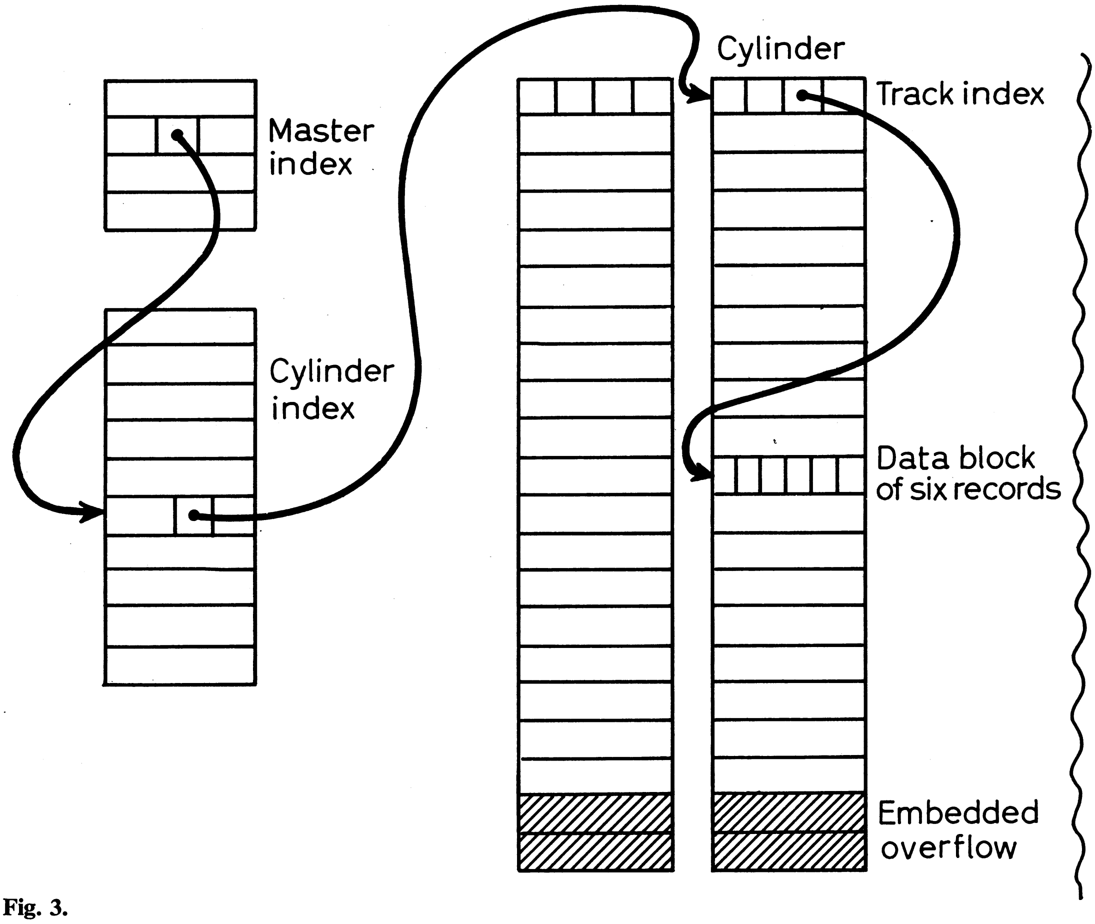

Fig. 3.

For retrieval and maintenance there are two points of entry to the data base, first via the Part Number Master file, and secondly via the Work Centre Master file. Both these files are Control Sequential in organisation and have a multi-level indexing structure. (**Fig. 3**). The data content of each of the records may be thought of in two parts; the first is descriptive of the item or work centre, i.e. price, manufacturer, or machine type; the second part consists of direct disc addresses of records either in the same file as the calling record, or in other files. These two files can therefore be thought of as acting as indexes to records in other files. None of the other files in this data base (with one exception) carry the part number, or work centre number; it follows that these files have no use or meaning in isolation. An example of this is that the information relating to stock holdings, and consumption which is held on the Stock file, is only accessible via the Part Number Master file, as the part number is not repeated on the Stock file record. While the disadvantages of this idea are obvious, there is a considerable saving when the part number is 28 bytes in length. A further advantage is that there can never be a record on the Stock file that is not also described on the Part Number Master file, so none of the linked files in the data base can be logically out of step with one another. Using the figures given in the Table of File Data (**Table 1**) the savings realised by omitting the key are as follows:

1. For the Structure file $\frac{28}{69 + 28} = 29\%$, 245 cyls.

2. For the Standard Routing file $\frac{28}{66 + 28} = 30\%$, 194 cyls.

3. For the Work in Progress file $\frac{28}{300 + 28} = 8{\cdot}5\%$, 19 cyls.

4. For the Stock file $\frac{28}{158 + 28} = 15\%$, 30 cyls.

5. For the Raw Material Conversion file $\frac{28}{84 + 28} = 25\%$, 5 cyls.

Total = 493 cyls. (or $2\frac{1}{2}$ drives)

$$= \frac{493}{1665 + 493}$$

$$= 23\%.$$

There are other fields it might be thought necessary to propagate through the files such as the descriptive keyword, of eight bytes. As a further example, if the Work Centre description had to be included for every shop in the route, the increased file size at 100 bytes per record would be six packs.

Within the Part Number Master file there is a considerable

Table 1 Table of file data

FILE DESCRIPTION	RECORD LENGTH BYTES	NUMBER OF RECORDS	DISC PACKS REQUIRED
Part Number Master file	204	239,651	3
Product Structure file	69	384,739	$1\frac{1}{2}$
Standard Routing file	66	780,863	$2\frac{1}{4}$
Work Centre Master file	3,452	950	30 cyls.
Work in Progress file	300	50,867	1
Stock file	158	127,000	170 cyls.
Raw Material Conversion file	84	10,000	15 cyls.
Deficiency Note file	130	15,000	41 cyls.

NOTE: A disc pack contains 200 cylinders.

number of 'handed' parts. This description encompasses parts with a high degree of similarity as well as left and right hand parts. There may be more than two parts in a set. The sets of handed parts are linked into a ring (**Fig. 4**) within the Part Number Master File by disc addresses. Only one of these parts is linked to a full list of manufacturing operations on the Routing file. The remaining records in the set are linked to an abbreviated list of Routing file records that represent the differences between themselves and the original record. A similar ring of disc addresses links records that are alternative parts; there is no priority sequence within the ring. This technique has reduced the amount of space required for the Routing file by 25%.

Maintaining the integrity of the data base

The status of each BOMP and EFS file is defined by a file control record. The file control record is read into core during the file opening sequence, and is written back to disc only when the files are closed. The file control record in core is amended whenever a record is added to or deleted from the file. It follows therefore, that whenever a maintenance program suffers an uncontrolled termination the file control record on disc will not agree with the file control record in core. In this case, while the data on the file remains accessible to retrieval programs, further maintenance is not possible.

The problem of providing adequate security for a data base of this size is highlighted by the fact that there are 10 million disc addresses linking these files. Furthermore it can be seen that a single failure is likely to be catastrophic in nature, rather than the slow deterioration that would be expected with a tape based system.

A file security technique has been developed to deal economically and safely with these problems. The first of the two major areas covered is that of overt program errors, operating errors, and hardware failures. What is required in this case is the ability to return rapidly to the position as it was before the failure occurred. This is achieved by high speed snapshot dumping of all the linked files, and the provision of an equally fast restore capability; this is referred to as physical dumping and is carried out by volume, rather than by file. The only programs initially available from IBM either required a dedicated machine, or special tailoring for each disc pack, in which case the program was only optimised for one file layout. A program was developed by the software section which copies an entire disc pack regardless of content, either to another disc or to tape; other than assignment of input and output devices, no control parameters are required. The program incorporates extensive functional checks and is specifically designed to operate in a multi-programming environment. Typically it takes 12 minutes to copy one disc pack to $1\frac{1}{4}$ reels of tape. A data base of 10 volumes requires just over two elapsed hours of dumping time. There are four sets of dump tapes. Although a physical dump is taken every week, a complex cycling system ensures that 6 to 8 weeks of security is available. It is necessary to keep sufficient input data to ensure that recovery is possible from the oldest dump held. In over 3 years of operation it has not been found necessary to fall back on the other forms of security that will be described. A failure between weekly dumps necessitates returning to the most recent weekly dump. It also requires the execution of all the ensuing file maintenance runs, to bring the files up to date. It takes as long to restore the physical dumps as it does to take them, i.e. just over two elapsed hours. Depending on what time in the update cycle the failure occurred, it takes up to 10 elapsed hours to bring the files up to date.

Covert program failures could produce latent defects in the file that do not come to light until all the physical dump cycle tapes contain the defective information. Since the essence of physical dumping is speed, no checking is performed. To give security against this form of failure the files are periodically unloaded onto tape. This is a more lengthy process but has two major advantages. First, most errors in the file are revealed. Secondly, the program attempts to recover the maximum of information, in spite of any file corruption that may exist. This unloading process is known as logical dumping. A logical dump is taken every month to 6 weeks. Recovery from this dumping technique requires that the files are reloaded. About 12 hours are required to carry out a logical dump, and about 200 hours would be required to reload the files.

The production control for the factory is entirely dependent on the functioning of the data base. There is no form of manual backup in the shape of written records, the data is held entirely in magnetic form. The above mentioned drawback in the physical dumping technique makes it vital that an alternative and absolutely reliable method of restoring is available. While reloading the files at other than specially selected times would cause serious disruption of normal work, total loss of the data would be catastrophic in a way that need not be detailed. Against the loss of all production control data accumulated over 6 years, 200 hours computer time is a small price to pay.

There is a special investigative program which is used to check the integrity of the address chains in the whole or a part of the file. Rarely it is used to prove the integrity of the whole of one or more of the files. More generally the program is used to explore limited areas of the file where a problem is suspected.

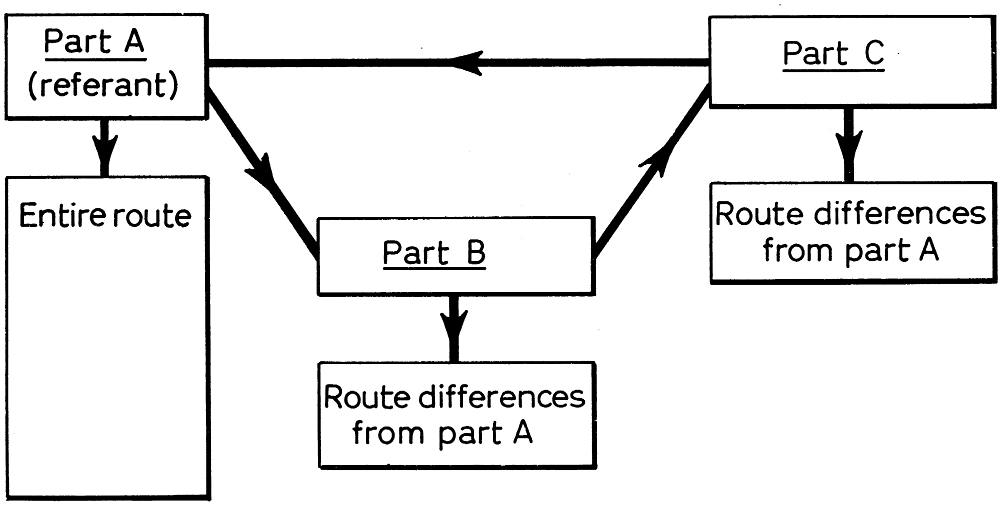

Fig. 4.

This program together with the facility to display and enter data into areas of disc directly are the principal tools used to recover from a fault situation by patching the files, rather than going back to a dump, and doing the updates again. Each file failure is individually documented. The documentation consists of a description of the fault symptoms; this is followed by an analysis of the nature of the fault and the possible causes. Finally the action taken to clear the fault is detailed. Included with the documentation is a set of before and after disc prints. It goes without saying that this sort of patching can only be done if there exists a very clear knowledge of the inter-actions of the files involved; it is a very skilled procedure. The indications of a failure vary from one problem to another and can come from operators, programmers, or users. Each incident must be separately considered. The approach to a problem is rigorous, it cannot possibly be otherwise in view of the sheer complexity of the files. In every case this type of fault rectification is carried out by two authorised people acting as a check on each other. There exists within the Company a software section, and the investigating team draws at least one of its members from this section. It appears that there is developing, in installations with large data bases, a role for a specialist 'software engineer', in maintaining the serviceability of the files.

The data base in daily operation

Production work (as against testing and development) on the computer occupies an average of 115 hours per week. Of this time 63% is taken up with Production Control applications most of which require some part of the data base to be on-line. The remaining time is devoted to Financial applications (32%), and Product Support (5%). The day-to-day operations involving the data base can conveniently be divided into three categories. Before doing this it is worth noting that the computer department acts as a bureau to the Production Control department which has its own data preparation service. The Production Control Department is entirely responsible for the data provided for maintenance, enquiries, etc. The first of the three categories of operation is maintenance. It is necessary to define maintenance, as a program which is capable of adding records to or deleting records from the file. This is potentially the most hazardous operation since updating of the file control record is involved. For the four BOMP files all maintenance is carried out by IBM supplied programs. These are tailored to the needs of the individual user and can contain large segments of user supplied coding. As the EFS files were developed in house, they are maintained by user written programs. Their maintenance involves the update of the file countaining the calling record. Update may be defined as changing the information content of records, without the addition or deletion of records, and forms the second category of operations. Updating can take two forms. If the data written to the file is independant of that already on the file (non-cumulative), the process is less hazardous than the case where cumulative records are being written. The hazard, with cumulative records, is this—if the update is performed twice over (accidentally or deliberately) the wrong accumulative total will result. On the other hand on a non-accumulative file, all fields will still be correct. It follows therefore that an abandoned update run on a non-accumulative file may be restarted from any point before the failure. The third category of operations only requires the retrieval of information from the data base. This operation poses no hazard to the file, other than that which might arise from hardware failure leading to physical damage to the disc.

An interesting situation is possible if two programs involving the data base are being run concurrently, and one of these programs is performing maintenance on the data base. If the maintenance program terminates normally before the other program terminates, and if this other program closes the file, it will follow that the file control record written back to disc will be the one obtaining at the time maintenance was started, rather than the updated one. There will be no indication of a problem until the next maintenance of the file is attempted. Problems of this sort are easily avoided, as soon as they are recognised. The problems are recognised by program failure, or from diagnostic messages from the various maintenance programs. In each case the problem must be assessed on its merits and a suitable solution devised. The solution can involve instructions to operators, amendments to programming standards, or modifications to software. It may involve a combination of these. It is the function of the software section to analyse the problems and devise the solutions.

Visual Display Units (IBM 2260) have been in use for over a year now on a trial and development basis. During this period they have been made available for two half to 1 hour slots twice a day. With the addition to the hardware complement of another three disc drives, and another four Visual Display Units the system will be made available for 8 hours a day, at the end of April. The Visual Display Units are situated in the Production Control offices to give independant enquiry facilities on the data base. They are operated in the local mode, supported by a locally developed terminal operating system. As yet there are no update facilities available to data base users (though other applications have been developed allowing update of other files). A major part of the effort in providing update facilities to data base users will go towards ensuring adequate security for the files.

At the time that the files were loaded, it was decided to allocate 20% of the prime data area on the Part Number Master file to overflow records. All additions are placed in the overflow area. When the overflow area associated with a particular seek area is filled, further additions are made starting at the end of the file, and working forwards. Unfortunately when this happens to any great extent the working efficiency of the file is impaired, and the time required for retrieval and maintenance increases. When first loaded, the records in a Control Sequential File are in logical sequence. When additions are made the logical sequence within the file is preserved by address links, but logically contiguous records do not remain physically contiguous. The records are arranged in blocks of six, each block is accessed from an entry in the lowest level index. It is possible, when additions have been made, for one of these blocks to encompass a hundred or more records after a year or so of operation. The records within one of these logical blocks are both in the embedded (same seek area) overflow and the external overflow, and are scattered amongst many physical blocks. In the worst case, i.e. a region that has received a large number of contiguous additions, it is possible for access times to increase by a factor of 50. After the first 18 months of operation it became clear that the addition of a new mark of aircraft had noticeably slowed those operations involving the retrieval by key of a record from the Part Number Master File from areas where numerous additions had been made. Similarly affected were additions, deletions and updates in these areas. Accordingly it was decided that the files should be reorganised to restore operating efficiency. The overall effect of this was to reduce maintenance times to half the pre-reorganisation figure. The same symptoms were recognised after a further 18 months of operation, and the same benefits derived from reorganisation.

Reorganising the data base

Since the data base was created, it has been reorganised twice. The method employed the second time was quite different from that first employed, and can be said to reflect the march of progress.

A standard reorganisation package for use with BOMP is supplied by IBM. Because the BOMP files occupy seven drives, and because all the files must be on line when this form of reorganisation is used, it would have involved gradually over-writing the old Part Number Master file with the new one during the course of the operation. The reorganisation program it was estimated, would run for about 90 hours. It is not possible to check-point programs that update disc files in place, in the normal manner. It must be done by interrupting the update periodically, and then dumping the files. It was not found practicable to do this with the reorganisation program supplied.

An easy alternative to this method was found. The logical dumping programs provide the file data in a form suitable for immediate re-loading. Before the final logical dumps were taken, the files were purged of all 'dead wood'. In this way some 20% of the records on the Part Number Master file were removed. A record was classified as 'dead' if there were no linkages in existence to any of the other files. This implied that the part was not in stock, on shop, or used by any assembly. The re-loading was commenced with the Part Number Master file, this took 3½ hours. The disc area required for the Product Structure file and the Standard Routing file was pre-formatted, and the Work Centre Master file re-loaded. The Product Structure file was re-created in 45 hours, the Standard Routing file in 90 hours. Both of these files were re-created using the normal maintenance programs, which are easily interrupted. The times given include the time spent taking intermediate file security dumps. The time interval between dumps was calculated using the rule of thumb that the time spent on security dumps should not exceed 20% of the running time spent in re-creation. The time required for this project was considerable, and so it was carried out when the factory was closed for the annual works holiday. The whole operation was carefully planned. A 24-hour rota covering the entire fortnight was prepared. This ensured that a Systems Analyst, Programmer and Software specialist were available at the other end of a telephone. In the event there were no failures of any sort, neither hardware, nor software, despite the heavy demands made on the reliability of both. It was at one time thought that the difficulties inherent in a project such as this would not render the attempt worthwhile. Our experience has shown the contrary to be the case. Comparison of run times before and after the re-load have shown some run times to be cut to a third of their former length. It will be noted that this exercise took approximately 60 hours less than the 200 quoted previously. The difference is in part due to the fact that none of the EFS files required re-loading because they hold no point back addresses. This eliminated the Stock, Work in Progress, and Raw Material Conversion files from the exercise. By far the larger part of the difference is due to the fact that a logical dump will be anything up to 6 weeks old and would normally need a great deal of maintenance to bring it up to date.

It may seem that the 90 hours quoted earlier as the time required to reorganise the files using the IBM package is excessive. Hawker Siddeley were amongst the largest users of BOMP; for users with substantially smaller files, the run times were acceptable. However, subsequent to this first reorganisation IBM introduced the Program Product DBOMP (Data Base Organisation and Maintenance Program). This includes a new method for the reorganisation of files, which in addition to cutting the time required to 40 hours also splits the process up into some six separate runs, thus posing no file security problems. The reorganisation of the Part Number Master File itself took 10½ hours. The subsequent runs were occupied in constructing, sorting, and using files of old and new disc addresses to update the Product Structure and Standard Routing Files. This dramatic reduction in run times, not only enabled us to reorganise the files over one week-end, but will in the future permit us to reorganise the files as often as required, instead of being confined to the annual works closed period.

IBM Manual References

E20-0114 Bill of Material Processor—A Maintenance and Retrieval System
This manual describes, conceptually, the organisation and use of a central product structure information system, giving examples of the variety of data to be maintained and reports to be extracted from this data.

H20-0197 IBM System/360 Bill of Material Processor—Application Description
This manual explains the actual file organisation techniques employed in the S/360 Bill of Material Processor Package. The user should have a thorough understanding of these techniques before beginning to define a system tailored to his specific requirements.

H20-0246 System/360 Bill of Material Processor Version 2, Programmer's Manual
This manual includes (1) a brief discussion of system definition, (2) a checklist of things to be done before modifying the file organisation and maintenance programs, (3) descriptions of all I/O files, (4) explanation and operating instructions for the sample problem, (5) descriptions of all the file organisation and maintenance programs and instructions for their modification and (6) descriptions and logic diagrams for selected user retrieval programs.

SH20-0829 S/360 Data Base Organisation and Maintenance Processor Program Description Manual
This manual provides an overview of the system, specifies the functions, options, and features included in the system, and provides information necessary for implementation. The manual is data processing oriented with a variety of application examples. The primary audience includes systems analysts and programmers.

Appendix 1: Extension file system

Application description

IBM System 360 DOS does not support non-sequential variable length blocked records on disk, neither does it support variation in record length without reloading of the file under any system of file organisation. The Extension File System fulfils these requirements by enabling the user to extend any record indefinitely at any time. The user has to provide four bytes in the record to be extended, and an area on disk to hold the extension records.

Applications that have benefitted from this technique include an advanced Production Information and Control System using the IBM Bill of Materials Processor package, and a Spare Parts catalogue. In the former the extension records provide information on Stock, Work in Process, Modification States, Customer Applicability, i.e. the information required in an engineering data base. The descriptive information in the Spare Parts catalogue is variable in length and this is carried on extension records.

Deleted records are made immediately available for additions. This is useful for files with a transient information content, e.g. Work in Process. Extension file areas that prove too small may be simply increased; no re-load of existing information being required. BOMP master or chain files, or Indexed Sequential files that call extension records do not need access to them during the course of reorganisation, an advantage when the number of disk drives available is a limiting factor. Extension files are not affected by the restriction to four of the number of associated BOMP files.

Package contents

The package as supplied requires no tailoring by the user, will support both 2311 and 2314 units either separately or simultaneously and is complete in itself. It contains a formatting program, and I/O module, and a set of macros. If the I/O module is to support more than eight extension files on line at once, a statement in it must be amended.

The formatting program is used to set up the area allocated to an extension file initially or to extend it subsequently. The I/O module is self-adaptive, self-relocating, and handles all the user's I/O requests on the extension file. The macros are supplied to facilitate the user's programming and provide linkage to and from programs written in Basic Assembler and COBOL.

Systems and configuration

The package is written in IBM Basic Assembler language to run under DOS. The I/O module occupies about 3,000 bytes of core; separate I/O areas must be provided by the user in his program and will be additional to this.

The formatting program uses Direct Access Method. The I/O module uses Physical Input Output Control System.

File description

An extension file consists of a number of blocks occupying one or more extents. These extents need not be contiguous, or on the same volume. The first 36 bytes of the first block in the file are occupied by the File Control Record (FCR). The FCR contains a number of parameters which describe the file and condition the I/O module at execution time; it is created by the file formatting program. The first four bytes of the FCR contain the address of the first record in the availability chain; they are known as the availability chain anchor. The first four bytes of each record are reserved for the chain link address. When the file is formatted each chain link address points to the next record in the availability chain of unoccupied records. The chain link address of the last record in the file contains the characters 'END'. In the file that is in use the availability chain contains all those records that are unused and thus available for additions.

When an addition is made to the file the record to be added is written onto the first record in the availability chain, i.e. the record pointed to by the address in the availability chain anchor. The availability chain anchor is then updated to contain what was the second record in the availability chain and has become the first.

When a deletion is made, the record to be deleted is set to binary zeros. The address of the next record in the availability chain is moved from the availability chain anchor in the FCR in core to the chain link address field of the deleted record.

The deleted record is written back to the file and the address of this record is placed in the availability chain anchor field. The deleted record will have been pointed to by an address in another record and could itself have carried the address of another extension record in the chain link address field. These pointers will be automatically reconciled by the I/O module. **Fig. 1** and **Fig. 2** show a chain of extension records before and after a deletion has been carried out, on the second extension record.

Retrieval and maintenance concepts

The I/O module supplied with the package is catalogued into the relocatable library after assembly and then included with the user's program at link edit time. The module handles all the user's I/O requests on the extension file, and maintains the

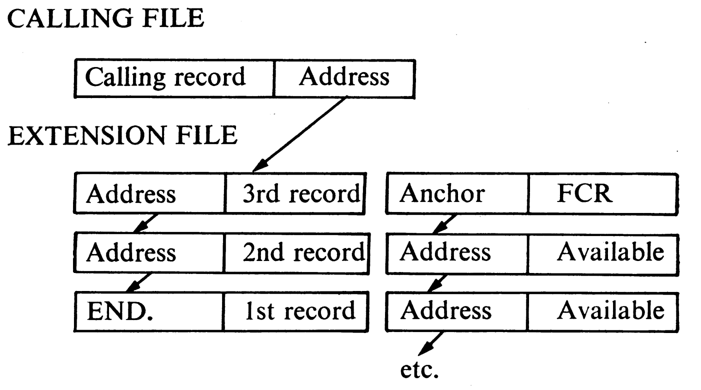

Fig. 1. File before deletion is made

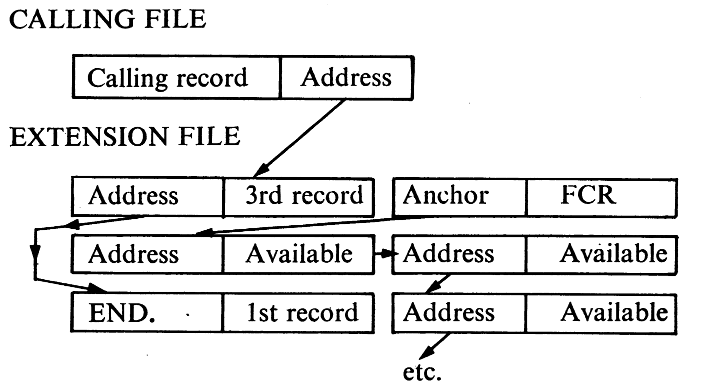

Fig. 2. File after deletion is made

necessary chains and anchors on addition and deletion. Up to eight files can be accommodated by the unmodified module at any one moment.

The interface between the user and the I/O module is provided by the standard work area prefix. The user communicates his I/O requests by placing a mnemonic code (the Process Indicator) in the standard work area prefix and then passes control to the I/O module by using one of the special macros provided with the package. (CA£LL or EFCALL).

The user can opt to have the module return control when the I/O has been completed, or alternatively when the request has been queued but not completed. This will allow processing to continue, overlapped with one or more I/O operations. If the request cannot be completed, the I/O module accesses two bytes in the work area prefix and sets the bits of these to indicate the reason for the failure. The user's program should test the error bytes after every I/O request to make sure that it was successful.

Disk addresses

For conservation of disk file space all disk file addresses which are to be stored in disk file records are first compressed from 9 bytes to 4 bytes as shown in **Fig. 3.** For subsequent use these compressed addresses are first expanded back to the 9 byte format. Routines to perform both expansion and compression are included in the I/O module.

Disk labels

The extension file system requires the use of IBM standard disk labels both for the formatting run(s), and for any file processing. The use of these labels will safeguard the files against accidental destruction and save unnecessary reconstruction efforts. It is necessary to supply the sequence number in all extent cards.

Expanded Disk Address

	Module	Cell		Cylinder		Head		Record	Record within block
	M	B	B	C	C	H	H	R	R′
Byte	0	1	2	3	4	5	6	7	8
Range	0–15	0	0	0	0–202	0	0–19	0–62	0–127
Bit configuration	00001111	00000000	00000000	00000000	11111111	00000000	00011111	00111111	01111111

$R \times BF + R' = 255$ max.

M	C	H	R
0	1	2	3
00001111	11111111	00011111	11111111

Compressed Disk Address

Fig. 3. Disk addresses

Programming considerations

The user is required to supply one work area per file, and at least one I/O area, when accessing extension files. The address of the I/O area is passed to the I/O module at file open time, and thus a particular file or files will be associated with an I/O area. It is possible to overlap the retrieval of records from files that are served by different I/O areas. It is not permissible to overlap the retrievals from files sharing a common I/O area. The I/O module checks to see if the required block is already in core, and higher operating speeds can be achieved by supplying a separate I/O area for each file. If core space is the ruling factor, all the files being accessed may have the same I/O area. The retrieval of an extension file record may then be overlapped with any I/O operation not involving the extension file.

The extension file system is characterised by great flexibility in operation. A balance can therefore be struck between the requirements of core space and speed of execution, without any modification to the software supplied.

IFIP WG 2.1

IFIP WG 2.1 (the ALGOL committee) has set up a small sub-committee to consider methods by which its responsibilities towards ALGOL 60 may be discharged. The convener of the sub-committee is Tony Hoare, and his address is:

Department of Computer Science,
Queen's University,
Belfast BT7 1NN,
Northern Ireland

As a preliminary to the work of the sub-committee, we hereby request submissions from users and lovers of ALGOL 60, stating views on how their interests may best be served by continued activity on the part of the committee. All suggestions and contributions on any topic (from burial to resurrection) will be welcome, and should be sent to the above address before May 31 1973.

If sufficient interest is evoked, contributors may be invited to a meeting for further discussion.