BARC-1549

BARC-1549

COMPRESSION AND DECOMPRESSION OF DIGITAL SEISMIC WAVEFORM DATA FOR STORAGE AND COMMUNICATION

by

Y. S. Bhadauria and Vijai Kumar
Seismology Section

1991

B.A.R.C. - 1549

GOVERNMENT OF INDIA
ATOMIC ENERGY COMMISSION

B.A.R.C. - 1549

COMPRESSION AND DECOMPRESSION OF DIGITAL SEISMIC WAVEFORM DATA FOR STORAGE AND COMMUNICATION

by

Y.S. Bhadauria and Vijai Kumar
Seismology Section

BHABHA ATOMIC RESEARCH CENTRE
BOMBAY, INDIA

1991

BIBLIOGRAPHIC DESCRIPTION SHEET FOR TECHNICAL REPORT

(as per IS : 9400 - 1980)

01	Security classification :	Unclassified
02	Distribution :	External
03	Report status :	New
04	Series :	B.A.R.C. External
05	Report type :	Technical Report
06	Report No. :	B.A.R.C.-1549
07	Part No. or Volume No. :	
08	Contract No. :	
10	Title and subtitle :	Compression and decompression of digital seismic waveform data for storage and communication
11	Collation :	34 p., 2 tabs., 8 figs., 1 appendix
13	Project No. :	
20	Personal author(s) :	Y.S. Bhadauria; Vijai Kumar
21	Affiliation of author(s) :	Seismology Section, Bhabha Atomic Research Centre, Bombay
22	Corporate author(s) :	Bhabha Atomic Research Centre, Bombay-400 085
23	Originating unit :	Seismology Section, B.A.R.C., Bombay
24	Sponsor(s) Name :	Department of Atomic Energy
	Type :	Government
30	Date of submission :	February 1991
31	Publication/Issue date :	March 1991

Contd... (ii)

40 Publisher/Distributor : Head, Library and Information Division, Bhabha Atomic Research Centre, Bombay-400 085

42 Form of distribution : Hard copy

50 Language of text : English

51 Language of summary : English

52 No. of references : 6 refs.

53 Gives data on :

60 Abstract : Two different classes of data compression schemes namely physical data compression schemes and logical dat compression schemes are examined for their use in storage an communication of digital seismic waveform data. In physical dat compression schemes, the physical size of the waveform is reduced One, therefore, gets only a broad picture of the original waveform when the data are retrieved and the waveform is reconstituted Correlation between original and decompressed waveforms varie inversely with the data compression ratio. In the logical dat compression schemes, the data are stored in a logically encoded form Storage of unnecessary characters like blank spaces is avoided. D decompression original data are retrieved and compression error i nil. Three algorithms of logical data compression schemes have bee developed and studied. These are : 1) optimum formatting schemes, 2 differential bit reduction scheme, and 3) six bit compression scheme Results of the above three algorithms of logical compression class ar compared with those of physical compression schemes reported i literature. It is found that for all types of data, six bi compression scheme gives the highest value of data compression ratio.

70 Keywords/Descriptors : DATA PROCESSING; ALGORITHMS; INFORMATIO RETRIEVAL; DATA TRANSMISSION; SEISMIC WAVES; FLOW SHEETS; WAVE FORMS NUMERICAL DATA

Additional Descriptors : DATA COMPRESSION; DATA DECOMPRESSION

71 Class No. : INIS Subject Category : F51.00; B31.40

99 Supplementary elements :

COMPRESSION AND DECOMPRESSION OF DIGITAL SEISMIC WAVEFORM DATA FOR STORAGE AND COMMUNICATION

by

Y.S.Bhadauria and Vijai Kumar
Seismology Section

INTRODUCTION

Multichannel digital seismic waveform data has a number of inherent advantages over their analog counterpart, including lower noise, easier data processing and compatibility with modern computer world. For many applications in seismology, preservation of seismic waveform data in a computer compatible medium like magnetic tape or floppy diskette, for each seismic event is very essential. However, taking into account the large amount of seismic data generated everyday at a multichannel seismic array station like Gauribidanur, the preservation of seismic data poses serious problems of storage and increasing cost. Moreover, if this data is to be exchanged with another user by using a commercial communication link , the cost and time factors become prohibitive. Since all the memory devices being of finite size, can store only a limited amount of data, there is constant pressure on the supplier of the digital data to store more number of bits of data on a given amount of storage medium.

Data compression may be usefully applied to store more number of bits of data in a given amount of storage. This will not only be advantagious for preservation purpose but for communication purpose also.

There are many methods of data compression reported in literature, all of them employing the same basic technique of reducing the number of samples required to represent a waveform.

This is done by exploiting one or the other feature of the waveform represented by the data. The data retrieval (decompression) is then achieved by employing some mathematical algorithm which describes this feature. In these schemes which are known as physical data compression schemes, there is always some finite error between the original and decompressed data.

However, there is another class of compression schemes known as logical data compression schemes which are not adequately probed. In these schemes the compression is achieved by reducing the volume of the data file as a whole, with all the samples retained as they are or in some coded form. The simplest method in this class is reformatting. The decompression in this case is achieved by decoding the sample values. Since all the samples are retained in these methods, there is no error introduced in the waveform.

In this report we present three data compression algorithms of this class developed for seismic data communication. A comparison of the two classes of the algorithms is also made by comparing the typical results. All these algorithms were developed one after the other in order to find out the most efficient algorithm for an experiment conducted to study the feasibility of exchanging the seismic data with a number of countries, using a computer to computer communication network. The details of the experiment are reported elsewhere (Vijai Kumar et.al,1990). It was found that the last algorithm namely the Six Bit Compression scheme is most suitable for this purpose. This algorithm can be equally useful for preservation of any ASCII data file.

2. GENERAL CHARACTARISTICS OF A COMPRESSION-DECOMPRESSION SCHEME

The merit of a particular data compression/decompression scheme must be judged by two factors: (1) The amount of compression that is achieved, and (2) The accuracy of the decompressed data.

The amount of compression may be measured by the data compression ratio (DCR) defined as:

$$DCR = \frac{\text{Amount of storage required without data compression}}{\text{Amount of storage required with data compression}}$$

Larger the value of DCR, The more efficient is the compression scheme.

The accuracy of the scheme which applies to the decompression portion of the algorithm may be measured in many ways. One way would be to describe the full time history and spectral charactaristics of the difference between the original and the decompressed waveform data. This difference is referred to as Compression noise (Peng and Iwan, 1990).

Another way could be to give the cross correlation coefficients between the original and decompressed waveform for various values of DCR which can be achieved for a given accuracy (Roy and Murty,1982).

3. TYPICAL SEISMIC WAVEFORM DATA

An examination of the general charactaristics of a seismic waveform will help to gain insight into the necessity of compression of such data either for storage or communication purposes. Fig.1(i) shows a typical seismic waveform of 90 seconds duration recorded at Gauribidanur Seismic Array (GBA) by one of it's channels. Figs.1(ii-iv) show magnified segments (A,B and C) of this waveform for pre-event noise, main seismic signal and post event phase of the record. The charactaristics displayed are typical of most of the seismograms. These figures clearly show, how the general amplitude, frequency and shape of the waveform change with time during the recorded event.

This waveform when digitised at a sampling rate of 20 samples per second, produces a data file of 1800 samples. A typical ASCII data file along with the header information is shown in Fig.2. At GBA there are 20 sensors of this type arranged in an L-shaped array. All these channels produce almost similar outputs of 1800 samples each for one event. On the average there are 10 events recorded per day. The preservation of such a large volume of data is very problematic and expensive.

4. METHODS OF DATA COMPRESSION

As mentioned earlier, for preserving time varying signals sampled at regular intervals, the possible data compression schemes can be broadly classified in following two categories (Ref.4).

1. Physical data compression schemes.

2. Logical data compression schemes.

4.1 PHYSICAL DATA COMPRESSION SCHEMES:

A physical data compression scheme reduces the physical size of the waveform. It includes the schemes which assume something a priori about the nature of time dependent data as a whole and retain only those data samples which deviate from a fixed threshold by some specified amount. Thus, in these schemes only a subset of actual data samples is stored and it is reconstructed (decompressed) by means of some mathematical algorithm. Inherent disadvantage in these schemes is that the original data can never be retrieved. One gets back only a broad picture of the original waveform with a given amount of correlation between the two waveforms. The value of this correlation between original and decompressed waveforms varies inversely with the data compression ratio. Roy and Murty (1982) have reported one such data compression algorithm based on autoregressive modelling of

seismic signals. Recently Peng and Iwan (1990) have also reported similar data compression schemes for accelerograms.

4.2 LOGICAL DATA COMPRESSION SCHEMES

In logical data compression schemes, attempt is made to reduce the number of characters required to represent the value of a particular data sample. All the data samples are stored in a logically encoded form. Emphasis is also laid upon to avoid the storage of unnecessary characters like blank spaces, in the data file. For fast and economical data communication this is very useful. On decompression all the original samples are retrieved and the compression error is nil.

In the following pages we will be discussing logical compression only.

5. COMPRESSION ALGORITHMS

Three compression algorithms were developed for seismic data exchange using communication links. For all these algorithms input is an ASCII data file along with header and footer information. A typical data file is shown in Fig.2. The algorithms are listed here in the increasing order of their complexity and correspond to the order in which they were developed one after the other to get larger values of data compression ratio. Thus six bit compression scheme which was finally used for the data communication purpose, gives largest value of compression ratio. The algorithms are:

1.Optimum formatting Schemes

2.Differential Bit Reduction Scheme (DBR)

3.Six Bit Compression Scheme (SBC)

5.1 OPTIMUM FORMATTING SCHEMES

5.1.1 Fixed field integer format

In the conventional sense it may not appear to be a compression algorithm but it is the most intutive way of attempting to develop a logical compression scheme. A closer look at the data file shown in Fig.2 reveals that the values of the data samples are varying right from one digit number to six digit number including minus sign, in a random manner. This file is written in 10I8 format and occupies 15335 bytes for storage. As seen from Fig.2, in this data file there are many blank spaces appearing between adjacent samples which increase the volume of this file. In order to reduce the storage occupied by this file by reformatting we require a minimum field width of seven characters (six for sample value and one space for separation between adjacent samples). Hence the optimum format for this data file is 11I7. Fig.3 shows the same file written in optimum fixed integer format of 11I7. This file occupies 13350 bytes of storage. As seen from Fig.3 no more reduction in storage can be achieved by conventional reformatting methods.

5.1.2 Variable field integer format (INTV)

In this algorithm we attempt to further reduce the storage occupied by the file by eliminating the unnecessary blank spaces. As seen from Fig.3 there are many spaces present between the samples which can not be eliminated by conventional reformatting methods. The reason for the existence of large number of spaces is that, in the fixed field integer format each data sample is allotted a fixed field width of seven characters irrespective of its value.

In the Variable Field Integer (INTV) formatting method we pack the data samples as closely as possible, by eliminating all but one blank space between the adjacent samples. This is done by

designing an integer data format with dynamic field width, that is the width of the format field varies as per the number of characters in a data sample plus one space for blank character so as to maintain the clarity of reading the sample values. No attempt is made to change the original sample values. Fig.4 shows the same data file written in variable field integer format. As seen in the figure the samples are packed as multiples of 80 characters (width of the terminal screen) with just one blank space between them. This file occupies 9060 bytes of storage as compared to the optimally formatted file of Fig.3 which occupies 13350 bytes. The compression ratio achieved is 1.47 in this case.

Description of the algorithm

As explained above, in this algorithm we vary the field width according to the number of characters in the given sample value. The data file written in any format is read and converted to 1018 format. The program counts the number of space characters in one field of width equal to 8 characters. Then whole of the data array is shifted to left by an amount which is one less than the existing number of blank spaces. This way the separation between two adjacent samples is reduced to one space. A counter is maintained to indicate the number of unfilled characters in a line of 80 columns. At the end of the line if the number of characters in the next sample is more than the number of unfilled characters of the line, the space is left blank and the next sample is written from the first column of the next line. This avoids the splitting of the characters of a particular sample between two consequtive lines.

Although there is no need for decompression yet the original data can be read from this file in any desired format. This algorithm was written on a PC using FOR77 in which such a format with dynamic field width does not exist. This simple method can be very useful for writing more number of ASCII data files on a floppy diskette either for storing purpose or for

sending the floppy itself.

5.2 DIFFERENTIAL BIT REDUCTION SCHEME

By writing the data file in variable integer format as in Fig.4, using the variable field width method discussed above, we have seen that no more reduction can be achieved untill we reduce the number of characters required to represent the value of a data sample. One way to achieve this can be to store the first data sample as it is and store only the difference between the next data sample and the previous sample. For example after storing the first sample value take the difference between second and first sample values and store it. Similarly store only the difference between third and second, fourth and third sample and so on, instead of the sample values themselves. This way, since the difference between two sample values will always be less than the samples it self, the resulting number will be of lesser number of digits. This is referred to as first difference. The same subroutine may be called again to get the second difference. The resulting data samples when written using variable integer format discussed in first method occupy lesser number of bytes. Fig.5. shows the same data file with second difference data written in variable integer format. This file occupies 8915 number of bytes. So compression ratio with respect to optimally formatted data file shown in Fig.3 is 1.50. This shows that there is not much of the gain in achieving higher DCR in comparison with variable integer method discussed earlier. The reason for this is discussed below.

In the case of a seismic waveform data, taking the first or second difference, in fact 'whitens' the background seismic noise and reduces its excursions. This means that by differencing the data, the sample values are made to vary between negative and positive numbers with an average value around zero. On this dat application of difference operation will introduce the bias now in opposite direction that is now the number of characters in

sample values will start increasing in negative direction resulting in no compression in number of characters required to represent a sample value. Thus this algorithm is highly dependent on the bias introduced in sample values at the time of digitisation. If all the samples of data file were positive numbers the DCR achieved will be more.

The original data can be retrieved by reversing the process of compression. That is now the differenced data samples are added to their previous sample value. The first data sample which is not differenced, is added to the second sample as many times as it was differenced, to get the original second sample. Then the second original sample so obtained is treated as the first original sample and added to next differenced sample. The process continues till all the differenced samples are converted to original samples.

5.3 SIX BIT COMPRESSION SCHEME

In the two compression schemes discussed so far we have seen that the compressed data samples were decimal numbers like 0,1,2,3...etc. and we were able to read them in the conventional form. We have also seen that the compression ratios achieved are also not sufficiently large. This is because, in these algorithms all the time we were interested in reducing the number of redundant characters like spaces.

In the six bit compression scheme (Muirhead 1990) the waveform data in numeric form is converted into a set of printable ASCII characters by encoding each data samples into different combinations of characters chosen from a look up table containing only 64 characters '+','-','0' to '9', 'A' to 'Z', and 'a' to 'z'. The constituent characters of the combination will depend upon the value of the data sample.

In this scheme only six bits of a byte are utilised. These

six bits can represent sample values from decimal 0 to decimal 2^6 = 64. If we allot one character for each decimal value in this range we require 64 characters. Thus sample values from decimal 0 to 64 can be represented by one six bit byte which can be encoded into a character chosen from the look up table of 64 characters, according to the sample value. Similarly for encoding sample values from 0 to 2^12 = 4096 we need two bytes and two characters and so on. However, for the sake of decoding, provision has to be made in each byte to indicate that subsequent byte is a part of the same sample value. Thus one out of the six bits of a byte is reserved as continuation bit, which if set indicates that subsequent byte (encoded as a character) is a part of the same sample and has to be considered while decoding. Moreover to represent a negative number one bit in the first byte only has to be used as a sign bit, which if set will indicate that the given number is negative. Thus there remain only four bits in the first byte and five bits in subsequent bytes which will represent the magnitude of samples.

Now let us see how the compression is achieved in this scheme. Consider the first byte which can represent values from 0 to 16. To represent values from 0 to 9 in ordinary way we require one character which is also required by this scheme. But for values between 10 to 16 ordinarily we require two characters but in this scheme we require only one. If all these numbers were negative we would have needed two charac rs for first block of 0 to 9 and three characters for second block of 10 to 16, but using this algorithm we need only one character for all these values. Thus a compression of two characters per sample.

Similar argument applies for two byte data, where from 17 to 99 there is no compression but from 100 to 512 there is a saving of one character for each sample. Thus this algorithm also works on the same principle of reducing the number of characters required to represent a sample value.

Description of the compression algorithm

Fig.6 shows the flow chart of the six bit compression algorithm. As discussed above the algorithm first converts the given sample value into sign and magnitude form. If the sample value is negative, the fifth bit (second MSB) of the first data byte is set to 1 by adding 2^4 = 16 to sign bit variable S which is otherwise 0. Similarly, for all the sample values except those which occupy one byte, the most significant bit (sixth bit) is set to 1 as continuation bit by maintaining the value of variable C = 32.

After setting the sign and continuation bits, the number of bytes required by a sample value is determined. This is done by successively comparing the sample value with the maximum decimal numbers that can be represented by given number of bytes. Thus if given value is more than 2^4 = 16, it occupies more than one byte, if it is more than 2^(4+5 = 9) = 512, it occupies more than two bytes and if the given sample value is more than 2^(4+5+5 = 14) = 16,384 it occupies more than three bytes. Like this we continue to compare upto 2^29 that is six bytes. Now each byte is encoded into a character chosen from the look up table. Thus in our case we can get a character string of maximum length of six characters for maximum sample value equal to 2^29. It is to be noted that negative numbers are also taken care off in the same representation. Thus the range of sample values that can be encoded by this scheme is from -2^29 to +2^29 which can be easily extended.

Now, in order to determine, which among the 64 characters to be placed to represent the sample value, we start with highest byte. The sample value is integer divided by a factor so chosen that the quotient value lies between 0 and 2^4 so that when we add continuation bit value (C = 32) and sign bit value (S = 1 or 17 for negative numbers), the sum does not exceed 64, the number of characters in the look up table. Thus six byte data is divided

by 2^25, five byte by 2^20, four byte by 2^15, three byte by 2^10, and two byte data is divided by 2^5. This quotient is added to S and C. This way the complete information of a byte including sign and continuation bit is encoded in just one character.

Now to choose the next character to be placed corresponding to second highest byte it is necessary to remove higher order bits of the sample value, already represented by previous byte. This is done by bitwise ANDing the given sample value with the maximum integer number that can be represented by one less number of bytes, than the previous byte. As we know that by bit wise AN-Ding an integer N1, with another fixed integer N2 so chosen that it has all 1's in its binary representation (e.g. N2 = 15, 31, 63, 511 etc.), all the higher order bits of N1 are truncated. For example if we choose N2 = decimal 31 (i.e binary 11111), the value of the function N1.IAND.N2 will vary between minimum 0 and maximum 31 irrespective of the maximum value upto which N1 can go. In this way, after encoding the first MSB byte into a charac-ter the sample value is truncated of higher order bits so that it now lies within the range of next lower byte. The process of en-coding is repeated to get second MSB byte character and again it is truncated. The encoding and truncation continues till all the bytes of the sample are encoded into characters.

Table-1 shows the ASCII character representation of some decimal numbers obtained by this algorithm. As seen from the table, for one byte data only one character is required for each value. Decimal 0 is represented as '+', decimal 1 as '-' and decimal 2 as '1'. For negative numbers the sequence starts from 'F' for -1, 'G' for -2 and so on.

In this way when all the samples are encoded into ASCII characters the output buffer is made multiple of 80 characters with blanks. The typical output file is shown in Fig.7 This file occupies 5009 bytes. The data compression ratio with respect to optimally formatted data file of Fig.3 is 2.66.

5.3.1 DECOMPRESSION ALGORITHM

In order to retrieve the original data from the six bit compressed data file the corresponding decompression algorithm was also developed. The process of decompression is reverse of the compression. Here we decode the ASCII characters into decimal numbers. Appendix A (Ref.3) shows the standard ASCII character set. As seen, every character in the set has a unique decimal value and the machine refers to binary representation of this decimal value, whenever a character is read from or written into the memory.

We utilise these decimal values of ASCII characters in the decompression algorithm. Like for compression, here also we make a look up table, but of 128 values because there are 128 ASCII characters out of which only 64 will be appearing in the six bit compressed data file. In this table, we write the serial number of the ASCII characters used in the compression algorithm, at the places which correspond to the decimal value of these characters. For instance character '+' represents decimal 0 in compression and it has got a decimal value of 43 in ASCII set, so 43rd entry of the decompression lookup table is 0. Similarly character '-' represents decimal 1 and has ASCII value 45 so 45th entry of the table is 1. This way we continue to write in the look up table and the last entry is 63 at the position 122 which correspond to character 'z'. Remaining positions of the lookup table corresponding to other ASCII characters which are not used at the time of compression are set to 0.

Description of the decompression algorithm

Fig.8 shows the flow diagram of the decompression algorithm. The decompression of the six bit compressed data starts with reading the first character from the data file. If this character is a CARRIAGE RETURN or a LINE FEED character it is ignored. As mentioned earlier when we read a character, in

fact we get a decimal number corresponding to its ASCII value and from the look up table we find out the decimal number represented by this character. From this number we find out the sign and continuation bits by bit wise ANDing it with fixed minimum numbers of five and six bits (i.e.16 and 32) respectively. These bits are then stripped off, again by bitwise ANDing and we get the magnitude of the value represented by this character.

If the continuation bit is not 0 this means that next character is also the part of the same value so it is read and the magnitude represented by this is computed as explained above and is added to the earlier computed value. The process of decoding the characters continues untill the continuation bit becomes 0. The value obtained after decoding all the characters of a sample, is the magnitude of the original sample, to which minus sign can be added if the sign bit is not 0.

In this way all the characters of the six bit compressed data file can be converted back to original decimal numbers which can be read in any format.

6. RESULTS AND DISCUSSION

Table-2 summarises the values of the data compression ratios (DCR) achieved by the three algorithms for different types of ASCII data files. All these values of DCR are calculated with respect to the original data file written in optimum format. The field width of this optimum format is equal to the maximum number of characters in the sample plus one. It is seen from the table that for all types of data the largest value of data compression ratio (DCR) is obtained by using six bit compression scheme and its typical value is 2.8. It is also to be noted that in these schemes the value of achievable DCR is highly dependent on the nature of sample values in original data file.

The Data compression ratio for Differential Bit Reduction

(DBR) depends on the average value of the data and may some time result in amplification with respect to Integer Variable Field Width method (S.No.2 of table-2). However, this method, followed by six bit compression scheme is very effective to compress the file having all positive sample values (S.No.1 of table-2).

The DCR in Six Bit Compression and INTV method is dependent upon the variation in number of characters required to represent minimum and maximum values of samples. The DCR achieved by six bit compression method is highest for all types of data.

7. COMPARISON OF LOGICAL AND PHYSICAL COMPRESSION SCHEMES

As mentioned earlier these algorithms belong to a particular class of compression schemes called as logical compression schemes. In order to study the merits and demerits of these compression schemes, it is interesting to compare the results of these algorithms with another class of compression schemes, known as physical compression schemes.

One physical data compression algorithm based on Autoregressive (AR) modelling the stationary time series for one step ahead prediction has been developed by Roy and Murty (1982). It is reported that the values of data compression ratio DCR and the crosscorrelation coefficients, C between the original and reconstructed waveforms are related to each other for different orders J of AR models along with other parameters of compression scheme. For a typical data it is seen that for DCR = 2.8 the value of C is 0.997.

Another set of physical data compression schemes has been reported by Peng and Iwan (1990). Although these algorithm are reported for compressing the accelerogram data but can be used for seismograms also. These algorithms employ storing of only the peak values of the accelerogram to achieve compression and reconstruction is done by different interpolation methods. It is

reported that in most of the cases the DCR is 2.3 for typical maximum compression error.

In both the physical compression schemes there is a tradeoff between the DCR and the error introduced during reconstruction. The DCR increases as the demand for accuracy decreases. One major inherent drawback of physical compression schemes is that once the data is stored at a certain value of DCR and accuracy, the original data can never be retrieved. If at any instant it is found that the accuracy of the decompressed data is not enough, there is no way to increase the accuracy. Moreover, the need for accuracy of the retrieved data may vary from user to user, the data manager at the data centre, whose main interest is to save the memory space, may not be able to maintain the tradeoff between accuracy and DCR so that the data is suitable to all the users.

Secondly the physical compression schemes require that the data to be compressed should satisfy certain conditions. For example, the method discussed by Roy and Murty requires that the time series comprising of the data to be compressed should be stationary and should satisfy the conditions necessary for AR modelling. That is the typical DCR values reported hold good if the waveform data to be compressed, consists of either pre-event/ post-event seismic noise or only the seismic signal. This method if used as it is to compress the waveform shown in Fig.1(i) may give lesser DCR values. However, piecewise modelling of three portions separately may give comparable results but computation involved will increase three fold.

On the other hand in logical compression schemes discussed here, although the DCR achieved is lower than the maximum that can be achieved by physical compression schemes but the error between original and reconstructed waveform is nil. The original data can be retrieved any time. Moreover, since in these schemes no assumption is made about the nature of the data so it is more

general in the sense that any ASCII data file can be compressed by these schemes.

The physically compressed data can be further compressed by the logical compression schemes discussed here and much higher than any one of them alone, DCR values can be obtained at the expense of same accuracy factor. But this should be done by the user himself and not by the supplier of the data. Especially for communication purpose the data should be compressed logically only unless there is an agreement between the user and the supplier regarding the desired accuracy of the data.

Thus, it is desirable that at the data centre the original data should be preserved in logically compressed form and communicated to the user. The user, if wishes can store it in any form of compression either physical or logical or both.

8. CONCLUSION

Two different classes of data compression schemes are examined. The results of three algorithms of logical compression class are compared with those of physical compression schemes reported in literature. It is found that the six bit compression scheme offers the value of data compression ratio which is comparable with that offered by many physical compression methods at reasonable accuracy. It is recommended that primary preservation of the data should be done in logically compressed form.

9. ACKNOWLEDGEMENTS

We are highly grateful to Dr. F. Roy of Seismology section BARC, for having many useful discussions. Thanks are due to Dr. G.S.Murty for his keen interest in this work. Our thanks are due to Shri R.N. Bharthur, Shri A.G.V. Prasad and Shri E.Unnikrishnan of Seismic Array Station, Gauribidanur for providing the digital data of the events.

REFERENCES

1. Roy,F. and Murty,G.S.(1982). Application of FPE based AR model for signal detection and digital data compression. Ann.Inst. Statist.math. 34(1982), part B. pp 181-188

2. Peng,Chia-Yen and Iwan,W.D.(1990). Some observations on data compression for digital strong motion accelerograms. Bull. Seismol.Soc.Am. Vol.80 No.2 pp 252-266.

3. ND FORTRAN reference manual,ND-60.145-06.

4. Engineering Note on Data compression by M/S Kinemetrics Systems U.S.A.

5. Vijai Kumar, Bharthur,R.N., Bhadauria,Y.S., Prasad,A.G.V., Unnikrishnan,E. and Subbaramu,K.R.(1990). Seismic data exchange using computer to computer communication network. Proc. National Symp. on Recent Advances in Seismology and their Applications, Gauribidanur, July 17-19.

6. Muirhead,K.(1990). Data compression algorithms, Personal communication.

TABLE1

SIX BIT COMPRESSION CODE FOR SOME DECIMAL NUMBERS

S.NO.	DECIMAL VALUE	SBC CODE	REMARK
1	0	-	
2	1	-	
3	-1	F	
4	2	0	
5	-2	G	
6	3	1	
:	:	:	
7	10	8	One character data
8	-10	O	
:	:	:	
9	15	D	
10	-15	T	
11	16	UE	
12	-16	kE	
13	17	UF	
14	-17	kF	Two character data
:	:	:	
15	31	UT	
16	-31	kT	
17	32	V+	
18	-32	l+	
:	:	:	
19	63	VT	
20	-63	lT	
21	64	W+	
22	-64	m+	
:	:	:	
23	95	WT	
24	-95	mT	
25	96	X+	
26	-96	n+	
:	:	: :	
27	511	JT	
28	-511	zT	
29	512	Uk+	
30	-512	kk+	
:	:	:	
31	543	UkT	
32	-543	kkT	
33	544	Ul+	
34	-544	kl+	Three character data
:	:	: :	
35	1023	UzT	
36	-1023	kzT	
37	1024	VU+	
38	-1024	lU+	
:	:	: :	
39	16383	VzT	
40	-16383	lzT	
41	16384	UkU+	Four character data
42	-16384	lkU+	
:	:	:	

TABLE-2

DATA COMPRESSION RATIOS OBTAINED BY USING LOGICAL COMPRESSION METHODS FOR VARIOUS TYPES OF ASCII DATA FILES. ALL THESE VALUES ARE COMPUTED WITH RESPECT TO OPTIMALLY FORMATTED RESPECTIVE DATA FILES OF EACH TYPE.

S.NO.	Type of variation in number of digits of sample values.	Data compression ratios			
		INTV	DBR	SBC	DBR+SBC
1	All 3 digit positive numbers	1.06	1.17	1.38	2.20
2	1 to 4 digits both positive and negative	1.48	1.26	2.80	2.55
3	1 to 5 digits both positive and negative	1.26	1.31	2.23	2.33
4	1 to 6 digits both positive and negative	1.47	1.50	2.66	2.67

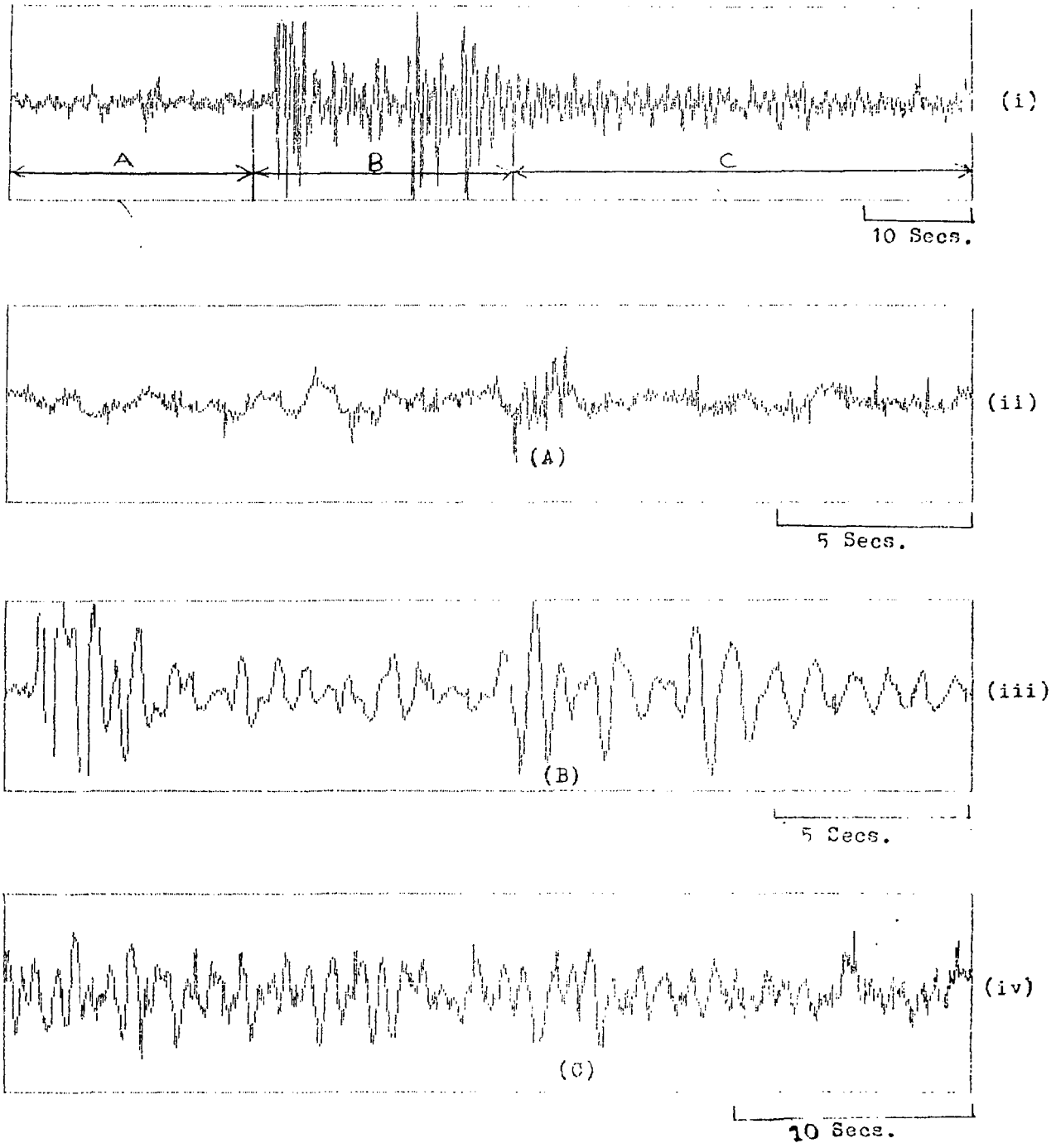

FIG. 1 (i) A TYPICAL SEISMIC WAVEFORM OF 90 SECONDS. MAGNIFIED VIEWS OF PRE-EVENT NOISE (A), SEISMIC EVENT SIGNAL (B) AND POST-EVENT NOISE (C) PORTIONS ARE GIVEN IN TRACE (ii), (iii) AND (iv) RESPECTIVELY.

```
XW01
WID1  1990333 12 10 59 000      1800 GBA      BEAM      SZ  20.0000000 WMKII INT8 0
 0.026054   1.0000    13.6150    77.5900    685.0000     0.0000    57.00     6.40   -1.0
R1 R2 R3 R4 R5 R6 R7 R8 B1 B2 B3 B4 B5 B6 B7 B8 B9 B10
DAT1
     420      321      638      590      293      545      316      316      271      620
      50      913        4      527      220     -259      128     -283      278     -208
    -135     -129     -582     -248     -516      182       64      239       -6      747
     -98      452      325      146      678       15      352      189     -487     -384
    -589     -594     -719     -566     -747     -774     -508     -793     -643      -19
   -1105     -240     -658     -318      103     -348       27     -328     -246     -465
     -44     -281       23      165     -260      268      114      149      692      416
     420      812      233      505      440      243      486     -101      384     -277
    -365       32     -410      264      591     -792     -171      550     -343     -205
    -233     -115     -530      -69       20      -35     -522     -498     -396     -333
    -242       15      141     -202      -34      121     -521     -282      -48    -1735
    -703     -411     -521    -1049     -521     -913     -539     -656     -662     -471
     433     -287     -307      164       11      272      352      578      133      123
     265      147      346      454       59      239      588     -289     -326     -387
    -702     -424     -643     -409     -801     -433     -463     -122     -175      295
     555      822      875      894     1814      844     1019     1032      753      656
     764      516      641      839      -58     -112     -604     -418     -551     -578

              (30 Lines skipped)

     100     -262     -104      845      223      683      139      816      260      111
    -115      387      151      206      999      861      157     -345      478      -20
     214     1069     -570     -357      115     1581     3438     6839    10616    12649
    8291    -3758   -21081   -34477   -36256   -26210    -7714     8549    18615    20106
   16668    11754     8210     5864     5725     7536     8062     7369     3205    -2399
   -9683   -16106   -18862   -16935   -10178    -1594     7201    11518    13386    12395
    8461     4288      263    -3174    -5089    -4395    -2779       94     2247     3863
    3019      570    -3385    -7775    -8443    -7116    -3209      151     3084     4951
    7153     8346     6960     4409      429    -2045    -3829    -4381    -3306    -2372
   -1303    -2247    -2121    -3064    -2683    -3031    -1083     -102     1663     2726
    3008     3531     2288     1634      -98      899      387      869     2257     1623
    2407     -406     -665    -2196    -2040    -1841    -2148    -2106    -1712    -1294
   -1033      121     -312      323      288      227      -62      586      582     -363
   -1264    -2204    -2118     -894     1146     3119     4313     4225     3163     1015
   -2127    -3473    -4421    -3799    -3021    -1933     -790    -1128     -501    -1279
    -983    -1196       98     1116     1879     3257     4114     3853     2743     1994

              (110 Lines skipped)

    -371      -28       84      217      726     -811      494      287       17     -213
     -66      480      -43      601     -147     -402     -646     -343     -939    -1346
   -1836     -827    -1115    -1180     -861      -28     -150     -164       65      482
    -271     -689      106    -1194     -302     -500      -55      -75      245      424
    1192      860     -235       98     -613     -376     -524     -715      -86     -760
     385      109     1123      598      394     1141     1066     1108     2653      739
     834     1203      941     1103      832      606      803      682      924     1504
CHK1      -76921
STOP
```

FIG.2 A TYPICAL SEISMIC WAVEFORM DATA FILE OBTAINED BY DIGITISING THE WAVEFORM OF FIG.1(i). ALL THE 1800 SAMPLES ARE WRITTEN IN FIXED FIELD INTEGER FORMAT (10I8). TOTAL NUMBER OF LINES REQUIRED WAS 187 INCLUDING HEADER LINES. THIS FILE OCCUPIES 15334 BYTES.

```
XW01
WID1  1990333 12 10 59 000      1800 GBA      BEAM      SZ  20.0000000 WMK11 INT7 0
 0.026054   1.0000    13.6150    77.5900    685.0000     0.0000    57.00     6.40   -1.0
R1 R2 R3 R4 R5 R6 R7 R8 B1 B2 B3 B4 B5 B6 B7 B8 B9 B10
DAT1
     420     321     638     590     293     545     316     316     271     620      50
     913       4     527     220    -259     128    -283     278    -208    -135    -129
    -582    -248    -516     182      64     239      -6     747     -98     452     325
     146     678      15     352     189    -487    -384    -589    -594    -719    -566
    -747    -774    -508    -793    -643     -19   -1105    -240    -658    -318     103
    -348      27    -328    -246    -465     -44    -281      23     165    -260     268
     114     149     692     416     420     812     233     505     440     243     486
    -101     384    -277    -365      32    -410     264     591    -792    -171     550
    -343    -205    -233    -115    -530     -69      20     -35    -522    -498    -396
    -333    -242      15     141    -202     -34     121    -521    -282     -48   -1735
    -703    -411    -521   -1049    -521    -913    -539    -656    -662    -471     433
    -287    -307     164      11     272     352     578     133     123     265     147
     346     454      59     239     588    -289    -326    -387    -702    -424    -643
    -409    -801    -433    -463    -122    -175     295     555     822     875     894
    1814     844    1019    1032     753     656     764     516     641     839     -58
    -112    -604    -418    -551    -578    -420    -649   -1969    -707    -268   -1039
     -87    -277    -588    -591    -302    -445   -1062    -506   -1170    -906    -780

            (26 Lines skipped)

     845     223     683     139     816     260     111    -115     387     151     206
     999     861     157    -345     478     -20     214    1069    -570    -357     115
    1581    3438    6839   10616   12649    8291   -3758  -21081  -34477  -36256  -26210
   -7714    8549   18615   20106   16668   11754    8210    5864    5725    7536    8062
    7369    3205   -2399   -9683  -16106  -18862  -16935  -10178   -1594    7201   11518
   13386   12395    8461    4288     263   -3174   -5089   -4395   -2779      94    2247
    3863    3019     570   -3385   -7775   -8443   -7116   -3209     151    3084    4951
    7153    8346    6960    4409     429   -2045   -3829   -4381   -3306   -2372   -1303
   -2247   -2121   -3064   -2683   -3031   -1083    -102    1663    2726    3008    3531
    2288    1634     -98     899     387     869    2257    1623    2407    -406    -665
   -2196   -2040   -1841   -2148   -2106   -1712   -1294   -1033     121    -312     323
     288     227     -62     586     582    -363   -1264   -2204   -2118    -894    1146
    3119    4313    4225    3163    1015   -2127   -3473   -4421   -3799   -3021   -1933
    -790   -1128    -501   -1279    -983   -1196      98    1116    1879    3257    4114
    3853    2743    1994     374    -651   -1123   -1781   -2143   -1810    -943    -319
     828    2579    2596    2706    3044    2709     540    -161    -877   -1396   -1038

            (96 Lines skipped)

     448     228    -245    -379    -454    -354    -597     637     -93    -144     270
     670     371     256    -403    -178    -180    -242     243     149    -157    -365
   -1074    -573    -722    -371     -28      64     217     726    -811     494     287
      17    -213     -66     480     -43     601    -147    -402    -646    -343    -939
   -1346   -1836    -827   -1115   -1180    -861     -28    -150    -164      65     482
    -271    -689     106   -1194    -302    -500     -55     -75     245     424    1192
     860    -235      98    -613    -376    -524    -715    -86     -760     385     109
    1123     598     394    1141    1066    1108    2653     739     834    1203     941
    1103     832     606     803     682     924    1504
CHK1      -76921
STOP
```

FIG.3 DATA FILE OF FIG.2 REFORMATTED OPTIMALLY USING FIXED FIELD INTEGER FORMAT (11I7). VOLUME OF THE FILE IS 13345 BYTES.

```
XWO1
WID1  1990333 12 10 59 000      1800 GBA      BEAM      SZ  20.0000000 WMKII INTV 0
 0.026054  1.0000   13.6150   77.5900   685.0000    0.0000   57.00    6.40  -1.0
R1 R2 R3 R4 R5 R6 R7 R8 B1 B2 B3 B4 B5 B6 B7 B8 B9 B10
DAT1
420 321 638 590 293 545 316 316 271 620 50 913 4 527 220 -259 128 -283 278 -208
-135 -129 -582 -248 -516 182 64 239 -6 747 -98 452 325 146 678 15 352 189 -487
-384 -589 -594 -719 -566 -747 -774 -508 -793 -643 -19 -1105 -240 -658 -318 103
-348 27 -328 -246 -465 -44 -281 23 165 -260 268 114 149 692 416 420 812 233 505
440 243 486 -101 384 -277 -365 32 -410 264 591 -792 -171 550 -343 -205 -233 -115
 -530 -69 20 -35 -522 -498 -396 -333 -242 15 141 -202 -34 121 -521 -282 -48
-1735 -703 -411 -521 -1049 -521 -913 -539 -656 -662 -471 433 -287 -307 164 11
272 352 578 133 123 265 147 346 454 59 239 588 -289 -326 -387 -702 -424 -643
-409 -801 -433 -463 -122 -175 295 555 822 875 894 1814 844 1019 1032 753 656 764
 516 641 839 -58 -112 -604 -418 -551 -578 -420 -649 -1969 -707 -268 -1039 -87
-277 -588 -591 -302 -445 -1062 -506 -1170 -906 -780 -131 -137 621 172 543 378
190 595 -90 255 140 102 156 -473 148 -285 143 360 66 264 783  510 -206 -238 -460
 -36 -523 681 -508 -183 -97 201 240 140 509 69 390 -148 211 -336 511 286 341 654
 -99 533 369 352 116 442 532 381 830 899 644 171 528 1357 193 462 -291 -99 -737
-171 -874 -546 -2921 -337 -398 -1076 231 1025 695 -1415 -809 -502 1338 -1233
-361 -123 291 -1448 1547 339 384 620 2291 1273 313 -862 619 2805 335 365 844 260
 644 32 -145 -376 102 -137 -650 -950 -364 -63 -693 -422 -105 -276 -260 438 -444
452 -24 29 -43 -253 143 -81 -302 -388 -491 -208 -237 349 -159 409 203 187 286 54
 693 47 540 236 380 254 304 249 2 611 21 306 3 596 -12 359 21 17 731 -198 326
-222 616 -4 -271 1390 -761 -546 -141 -467 -452 -220 -752 61 -665 42 -657 -315 79
 -284 339 -645 415 -508 384 126 383 207 629 64 96 453 -132 194 -403 -82 79 -348
-329 -748 -383 -473 -420 -91 -472 -395 -205 -712 -334 -490 458 -152 -980 -341
-904 -361 -253 -269 489 -38 -1120 151 224 481 742 838 913 331 640 634 790 988
397 741 160 886 670 141 846 -382 723 -34 -83 265 -217 -81 94 24 -348 178 198
-419 -128 1379 -366 54 161 -787 215 -209 -107 296 1 -129 47 -78 -412 223 -461
-241 -36 362 -204 212 63 -379 -463 -719 1283 -545 -398 -251 -370 -371 -375 130
-555 -180 -305 -385 -358 100 -262 -104 845 223 683 139 816 260 111 -115 387 151
206 999 861 157 -345 478 -20 214 1069 -570 -357 115 1581 3438 6839 10616 12649
8291 -3758 -21081 -34477 -36256 -26210 -7714 8549 18615 20106 16668 11754 8210
5864 5725 7536 8062 7369 3205 -2399 -9683 -16106 -16862 -16935 -10178 -1594 7201

              (66 Lines skipped)

-1219 -662 -356 -508 16 -549 -110 -411 -707 257 273 786 837 1219 455 744 522
-332 -537 -610 -880 -1132 -1259 -1080 -833 -806 -347 -77 494 932 1032 1357 1678
978 1064 912 395 -134 -69 -106 180 -591 45 -932 -592 -824 -510 -55 168 1180 309
234 -480 -572 -727 340 -1241 -626 -357 -476 -24 -155 -279 -444 -68 -983 -491
-581 -1017 -363 -582 402 220 818 1099 852 -113 469 802 59 -228 -280 -658 -431 92
 224 -325 680 177 517 614 703 616 220 -242 -628 -506 -1082 -771 -642 -268 -81
279 362 320 475 270 667 144 -142 -412 -936 -447 -1192 -516 -1444 -1038 -776 9
-374 -633 -260 -65 -162 -621 -375 -356 -327 360 -718 -61 -29 -369 -203 -338 -118
 907 861 836 2022 1280 1777 1374 1906 1030 887 871 3067 1390 448 228 -245 -379
-454 -354 -597 637 -93 -144 270 670 371 256 -403 -178 -180 -242 243 149 -157
-365 -1074 -573 -722 -371 -28 64 217 726 -811 494 287 17 -213 -66 480 -43 601
-147 -402 -646 -343 -939 -1346 -1836 -827 -1115 -1180 -861 -28 -150 -164 65 482
-271 -689 106 -1194 -302 -500 -55 -75 245 424 1192 860 -235 98 -613 -376 -524
-715 -86 -760 385 109 1123 598 394 1141 1066 1108 2653 739 834 1203 941 1103 832
 606 803 682 924 1504
CHK1      -76921
STOP
```

FIG.4 DATA FILE OF FIG.2 COMPRESSED BY VARIABLE INTEGER FORMAT METHOD (INTV). THIS FILE OCCUPIES 9060 BYTES OF STORAGE.

```
XW01
WID1  1990333 12 10 59 000      1600 .BA      [illegible]                    6.40  -1.0
 0.026054  1.0000   13.6150   77.5500   [illegible]
R1 R2 R3 R4 R5 R6 R7 R8 B1 B2 B3 B4 B5 B6 B7 B8 B9 B10
DAT1
420 -519 416 -365 -249 549 -481 229 -48 394 -611 [illegible]  -172 866
-798 972 -1047 559 -67 -459 787 -802 986 [illegible]  -677 -52
711 -1195 1000 -500 -513 779 -308 20[illegible]  -1710
1951 -1283 758 81 -872 826 -730 437 -501 [illegible]  189 508
 -819 280 388 -971 851 -337 -132 940 -6[illegible]  -347
-1710 2004 100 -1614 1031 -166 144 -5 [illegible]  166
-131 -469 511 -13 -797 881 -5 -1921 2[illegible]  491 111
197 713 -1624 700 491 -624 414 -161 1[illegible]  575 169
-1226 840 -24 -254 593 -497 453 -626 7[illegible]  301
-1890 1145 -162 -292 182 205 -356 373 73[illegible]  -367
-1091 2582 -823 -1210 1723 -1142 -121 [illegible]  -135 523
-655 764 -1207 820 -535 -25 594 -1080 [illegible]  881 -211
-511 492 321 -1812 1597 -336 -100 646 [illegible]  -138
469 -809 761 -859 897 -906 1394 -1072 [illegible]  -68 -238
-241 600 -380 -324 -218 830 472 -1903 [illegible]  -2703
4959 -2645 -617 1985 -513 -1124 -1780 [illegible]  -2153
4734 -4203 1253 191 1435 -2689 58 [illegible]  -998
435 -54 709 -717 -274 913 886 -285 [illegible]  529
 -125 -138 606 -620 3 135 -17 386 [illegible]  571
-1285 1139 -797 448 -270 176 -105 [illegible]  -709 334
 718 -1643 1453 -1072 1386 -1458 [illegible]  1545
-1539 1433 -1406 1041 52 -757 986 [illegible]  -987
597 325 -942 911 -923 918 -160 [illegible]  -897
885 -534 1104 -1558 -218 1487 [illegible]  -1198 184
 4 -185 -21 -657 991 -315 162 [illegible]  333
-1862 708 397 -830 618 39 -245 [illegible]  -1055
 1950 -1426 526 301 -698 165 306 [illegible]  -565
-293 358 -172 2258 -3830 1975 [illegible]  -829
520 791 -1571 1082 -1004 1221 [illegible]  1327

            (64 Lines skipped)

 -1288 1485 -710 498 -234 195 [illegible]  -838
1296 -1367 1021 -507 101 -400 [illegible]  497
 -462 331 -1146 1053 -511 -632 [illegible]  -133
 -338 225 -4 -1021 786 -238 [illegible]  546
 141 -232 789 -1883 796 -639 [illegible]  -41
541 -1291 1407 -582 -346 1090 [illegible]  -1076
 456 235 -326 605 296 -391 [illegible]  -608
 887 -182 245 -187 173 -277 [illegible]  1421
-1604 1334 -144 523 -1168 [illegible]  -625
 -372 506 -301 355 805 [illegible]
-3873 735 722 -253 339 [illegible]  -827
 -60 547 -579 -212 98 [illegible]  40
377 399 -1069 1167 -1392 [illegible]  1198
 243 188 -1170 335 1213 [illegible]  1428
-1044 948 -385 -43 820 [illegible]  5159
2009 274 -631 424 -433 [illegible]
CHK1      -76921
STOP
```

FIG.5 DATA FILE OF FIG. [illegible]
METHOD (DBR). THIS [illegible]

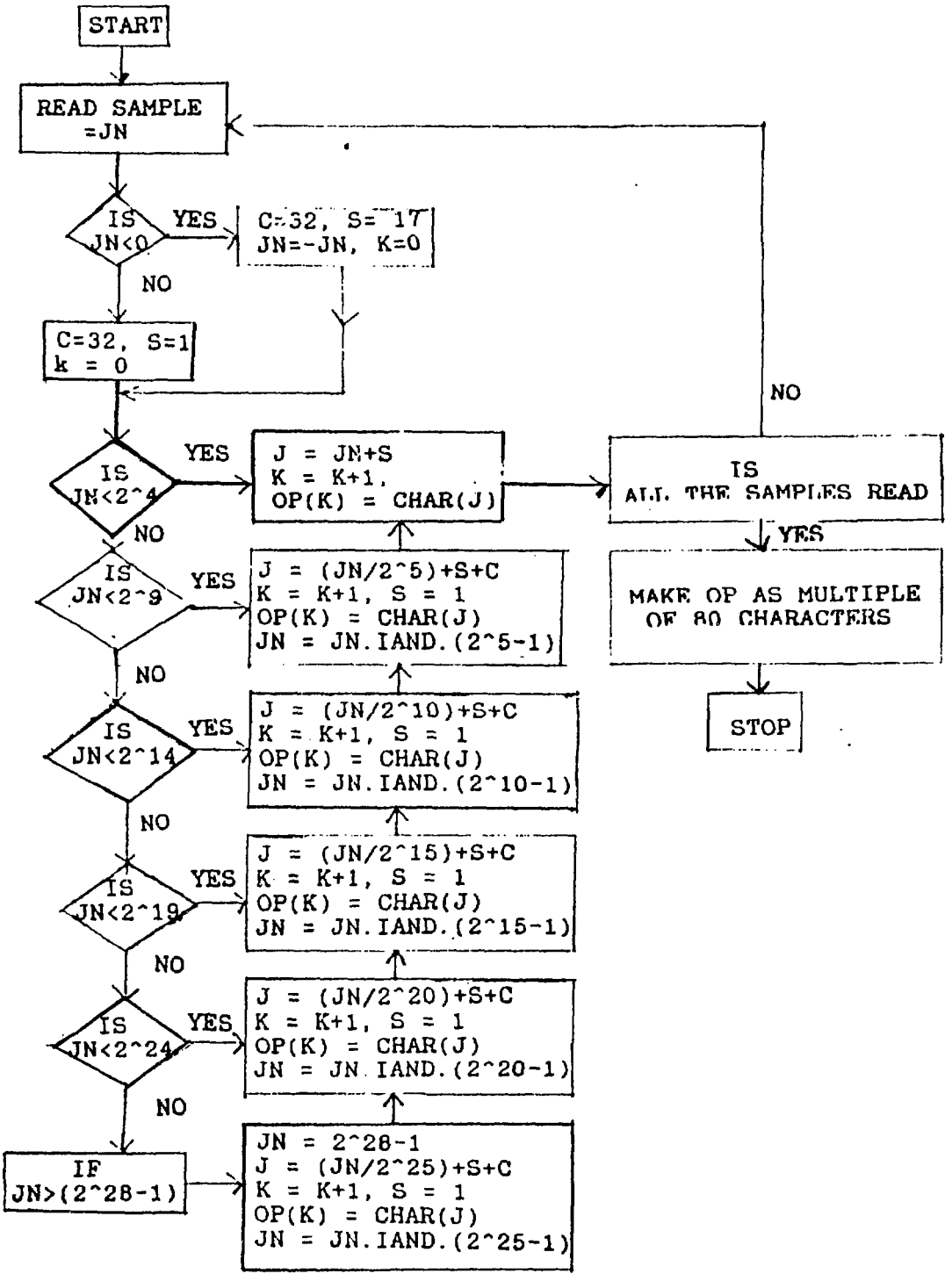

FIG.6 FLOW CHART OF SIX BIT COMPRESSION ALGORITHM

```
XWO1
WID1  1990333 12 10 59 000      1800 GBA      BEAM      SZ  20.0000000 WMKII CMP6 0
 0.026054  1.0000   13.6150   77.5900   685.0000    0.0000   57.00    6.40  -1.0
R1 R2 R3 R4 R5 R6 R7 R8 B1 B2 B3 B4 B5 B6 B7 B8 B9 B10
DAT1
h2e-UnSUmCd3U1-dQdQcDUnAVGUwF2UkDaQs1Y+sPcKqEo5o-km4rMkk2ZKW+bDKUr9nOi2e3YGUp4Df
+ZRz5w+kmBkmGkqDk1Kkr9ks4zQksNko1kH1WFrEkoGtSX5uQUPu6rKyF1AsNULZ3s2cAXGYJUpIh+h2
UtAb7jNhMbHj4n3g+sJvBV+wOc6UmDksMp9U14uLqBr7nHkkGm3UI11kk8zGwAuBrGDYBq810XNkk7sO
1E1q5kpTwPkk71UNkk7kwFkkPkoEkoKyLhFsTtHZ29cEf+Um0Y3XPc7YHeOi4VPbDUmAt-u4w1kpSx6k
o1wNkt-xFyDnOpDd5U19UtKUv9UvSVsKUuAUzPVU6UrFUoEUrQUk2Uo-Uu51OnEkmQx0k15km0x2ko71
xFkq1sA1UDmLsJkmAkmDtCxR1V4zO1YGkw8ksAo1o7UnBZAUkTfPZSUmHmObTYAX4YQyNYIsRYDf6W0c
6UsDzSqCrCyAl2kk9Up7zQpLn-a7bEYAjRW3g4oIaHuEjTcSeJUoCn1UkJfFf+XIhOUkIfRUtSUw1Uo2
Z9UkEVeBa-iCt1n1kr-p9kv8klOmv7uFwC1VIb5VU-UpL1g5kt7zKVdO1aFv7nPd11h6Vk9eHg+UnAWb
HVbNdNkuSUn9WrJeDfBUuAc2Uo2V+oFvMX4o7ko8kxKvAlTkpJx4n7sIs2hKxQi2kMUR19rRYDmFtCw2
z9qErBeRoTgNa9ZPcSVKUpJVDUkQbAfQbSdEbNOUn1UJdG1UmIQf5UJUFUqPq4e4qSUn6IsDVfCkrNk1
0oByHy2qQkrEVRkoNV8koFtPWDsQeHko3gTzQg+XSfTaDUnJW+X+i3o2a0wHmGWDuQu7krAvTyNx2mPy
Mw9qBkq6uCz8i8oMkyIuJkw6v7rRsBj7141X+YLb+j-Ur4Uu4UwFe9Uo+UnOUsKUyQgBUr3Z+UvKUoSY
BUuCvSUqHl0mHc7qNmFWSUMuQZGa4x1o+Vf1vCVKZ-ksHaLqFn9d6-o-VDmCwQaTyBrFl2f8qAaIVTvP
yDkqDVc1kl-wCrPvGvHvLY0kl9pItFw-v4X2s4n6UuBaTUp9Y9UtEc2XDnHg1YLaCUz5uURYRuNiSkIa
KVVBklOv3XHVlBXfCapLefMgf7cX1npCkomNlVpBlXh+ktn0r10cf3UmZLUno8UkcQfj8cUGZr6ZmRbf
EbvSba7XY3meTtiHzr8kmhCkk15ty011ObV-fbShW8gX9ccBYa+c5nX4oz-od9mqPWSWa5XsLWy9U1On
dNrmTsbPqyAnY7YLXUAYuLazFcYOatEYdNhBlzRnrJocRnb8me21cLma5mW7mzMmnPmyL1VPn4VnTWp4
Wy+Xi9WbEVn0n0Uw1g1Uv3WaFVmLWf5wKkoNmYI1zM1tFmX2mVOlpElcC1U7XNtMe1d+b11SUm8Um4v9
1bEmYQmW4kvSVXOXVDYaNYY-XWPUzLmWDngFoe3nqLmyB1wBksK1X6zJ1bTkyL1ZAX0VWQVuLXZNYUGX
sBWpLVy8fKko91X11rJmWT1sGkxDtTUtQWkHW12WoGWz2WoJUkQp-kvB1fI1UCktGk12VSkGUoFUrAe0
bAh51pDmUGmm+1cPUy3Vd2VzEVZJu0oD1h9yBkzRkyM1f6mg5mvQnWJnp5nhFmrF1pGoQU12VhHVtNVy
4VyGXVCXv9YZEYkGXhEVbMkzDmfDmyKnkGmh0mfF1cEkqPVs4WUDXqAVxSVVHVjLbAWcOVjIVZ3sA1XJ
1fT1g1kp5rFn-y1i3hPUsNUpLU19nSTnTxSzOaN1pOmcKmg71tAmYO1f1kz9q4zLt5kmDkoHkq6km3ZC
VY4WgNXjAYsNYi4YiAXq611YMUw9mmJmnTpjNpm-tpCsb-rU6nnLUv8ZY1dWRf1TfsJeX5bUEWwC1fLp
dCrpSsgOrW6oW0kwHVbOWrAXaCXiGWnOVY4ksGlo3nf2nWF1vBkmEUvOVWFUsBUuIdOUvGVq4WYDWhKV
g0fD1dTof3puCsa2ruQqdHnrGuMWs9Yb6ZV4YoCXhEXZTWnDWrPWoFVpCUqPkrG1jRlzGmsTnhTmrFnU

            (13 Lines skipped)

1fL1oSmgCmZ91tC1aPkl2X1gLt+a+1Mk1QoLm8m7jR1NdCi0hMUw7eOeQkv91UNkoQkxHdOUn1Vu0Vy0
VkGVjR7a4yOkp2oLknFfLoDvNx3wGs7UoDhGVa3Ux+VeSUxQbNi9koCkrFkoT1sS1z-mp7mhLmVJkyMz
AbNg-Uw4UwBV1NVnCVq-Vo1Uu7VVGkn1u81dTloEla21WCkn1sAhK-uIVMeBx+fPWUDh5VaBVULVfQVk
-VhAVe6gFUo6x6kv11k01yQmmDmd41wQ1WSwEiPVYOVa6Vq9UuOUtTpFkwQkxG1x8mV6ma1mXBmZJ1i3
krFuKU11Z8VkDUyAVY7UuOVgIUkAUy1UxFd4Un-AYILdLfIVX8Uv1VbEVmHUmIU1KPtIkvBwTko2kp9U
LktKkkNVJoRfOWCW1b9nFrEtLkt3vLkoKuQq3vOwNkrD1ZF1d41eD1151X1kyTz1p1f0Y2kGj4U1-Wc5
ViSVdSVnJVe2VWBUkDkMmOz-dIkpFkm4knRkx2xT13uQwBrIs9iGUoOVUHVZNVe8UxPVYEj2c0rJksTk
mKwN1ZCkkQbP8UqEVZLVbKVaJUw-UuCXScMZFkqSk1J1tL1zOmkFmkSmhPmhLmbHmaK1yC1gR1Z5knQU
TUHUrOVUDUuGUoLVVTUvFWW5cTV9pGwDkkRtB-Um8b8Us0UvOVeCVYNUw6UoPaTkq01gSqOVPUzKe7Vb
FVf7VoMVi7WY1VqPVmJVb7aDvE1pTmVNmo1m1SmqC1r4kzSkyEy3oFjAZHUoBUnEcFuAtIuOkpI1U4xJ
1VPko4koBYDkuIyBko01XCkp6W4qMbKVWIUnJVYTUx2VcBUu7b-W5kzAk1Q1g1ksTkvEyDsMX0Uk+ZQU
16dGUqMUs4Uo2UvRUn-Up91Rj9u2nKx1kn91a1koKv2zQUEk13nCwPkq1c-cFUsGUu3Va1i5Ur6Uk8uA
kkNkn0kvE1XA1b91VMku-kt4uPmBjCUx2VU6VeBVoCUyGVV6UwEg9o4m3n8ZIkmDVBkx2kmEktMzS1LZ
6VYQdJb8z+k1QkqLe1laNknGv3yQkMoPsLxQm2kyLz9km3kzNv9km4gGaQUtGVW9UuInFiJUt0VPr2sM
koGxDWQb+u3Up6ZFUk3Un4UpTUn6aQrGknIz01VOks1ko0sAmFcLf8e+iPcCUoPYEoCwQkx6xT1Z6kk2
1h21UCks67vKknNs2m-p0knBvLv2u5f6kqC1RkRvFq9uGnKUw9UuRUu2Vz4Vc+VrFVeSVvGVU4UvLUv5
WzPVfCi+b2rJvPy4v0kmJUnRmRoEcCUoSfHc+wHpGpIrGbHYJoRvB1VGk1RkqGvHkQW+aNUqKkt9jCcT
UFqJm0j+19UmNoHwGko4uLkx91e01tAktP1WP1YQkuRkQoKp2W-j0sDkpFX81Z8tCzI1Lm9bJh6VZ6Uu
Qr9X0kn3vMkkAkq9mKkrMg-XBVX1UmKg8VXJVV8VWIWmRUr1Uu0VZHUxBVWDUu+UmSUt1Up8UwQVj+
CHK1      -76921
STOP
```

FIG.7 DATA FILE OF FIG.2 COMPRESSED BY SIX BIT COMPRESSION (SBC) METHOD. THIS FILE OCCUPIES 5009 BYTES OF STORAGE.

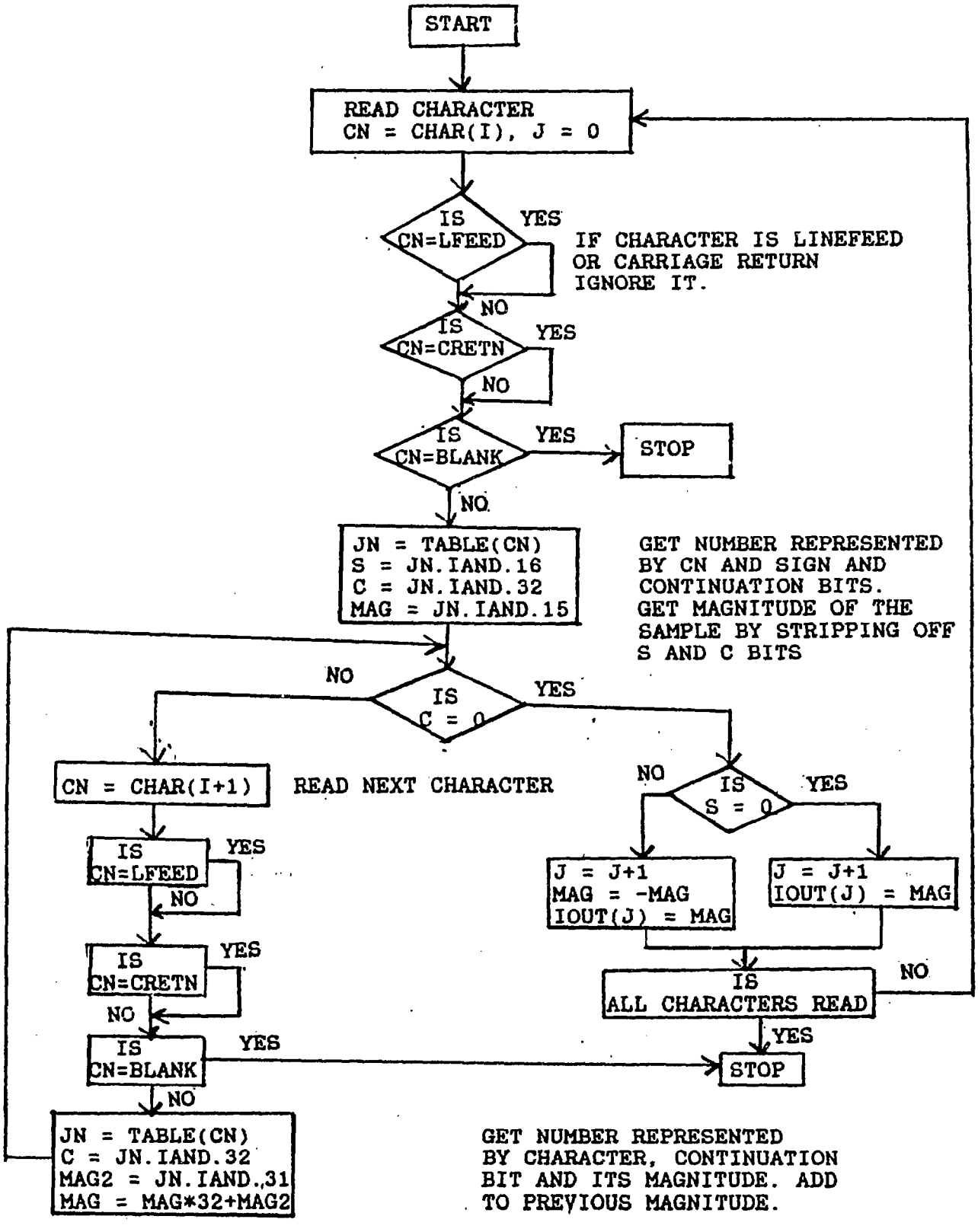

FIG.8 FLOW CHART FOR DECOMPRESSING SIX BIT COMPRESSED DATA

APPENDIX A

ASCII CHARACTER SET

Octal Value: Left Byte	Octal Value: Right Byte	Decimal Value	ASC Abbreviation:	Comments:
000000	0	0	NUL	Null
000400	1	1	SOH	Start of heading
001000	2	2	STX	Start of text
001400	3	3	ETX	End of text
002000	4	4	EOT	End of transmission
002400	5	5	ENQ	Enquiry
003000	6	6	ACK	Acknowledge
003400	7	7	BEL	Bell
004000	10	8	BS	Backspace
004400	11	9	HT	Horizontal tabulation
005000	12	10	LF	Line feed
005400	13	11	VT	Vertical tabulation
006000	14	12	FF	Form feed
006400	15	13	CR	Carriage return
007000	16	14	SO	Shift out
007400	17	15	SI	Shift in
010000	20	16	DLE	Data link escape
010400	21	17	DC1	Device control 1
011000	22	18	DC2	Device control 2
011400	23	19	DC3	Device control 3
012000	24	20	DC4	Device control 4
012400	25	21	NAK	Negative acknowledge
013000	26	22	SYN	Synchronous idle
013400	27	23	ETB	End of transmission block
014000	30	24	CAN	Cancel
014400	31	25	EM	End of medium
015000	32	26	SUB	Substitute
015400	33	27	ESC	Escape
016000	34	28	FS	File separator
016400	35	29	GS	Group separator
017000	36	30	RS	Record separator
017400	37	31	US	Unit separator
020000	40	32	SP	Space
020400	41	33	!	Exclamation marks
021000	42	34	"	Quotation marks
021400	43	35	#	Number sign
022000	44	36	$	Dollar sign
022400	45	37	%	Percent sign
023000	46	38	&	Ampersand
024000	47	39	'	Apostophe
024400	50	40	(	Opening parenthesis

Octal Value: Left Byte	Octal Value: Right Byte	Decimal Value	ASC Abbreviation:	Comments:
024400	51	41	)	Closing parenthesis
025000	52	42	*	Asterisk
025400	53	43	+	Plus
026000	54	44	,	Comma
026400	55	45	—	Hyphen (Minus)
027000	56	46	.	Period (Decimal)
027400	57	47	/	Slant
030000	60	48	0	Zero
030400	61	49	1	One
031000	62	50	2	Two
031400	63	51	3	Three
032000	64	52	4	Four
032400	65	53	5	Five
033000	66	54	6	Six
033400	67	55	7	Seven
034000	70	56	8	Eight
034400	71	57	9	Nine
035000	72	58	:	Colon
035400	73	59	;	Semi-colon
036000	74	60	<	Less than
036400	75	61	=	Equals
037000	76	62	>	Greater than
037400	77	63	?	Question mark
040000	100	64	@	Commercial at
040400	101	65	A	Uppercase A
041000	102	66	B	Uppercase B
041400	103	67	C	Uppercase C
042000	104	68	D	Uppercase D
042400	105	69	E	Uppercase E
043000	106	70	F	Uppercase F
043400	107	71	G	Uppercase G
044000	110	72	H	Uppercase H
044400	111	73	I	Uppercase I
045000	112	74	J	Uppercase J
045400	113	75	K	Uppercase K
046000	114	76	L	Uppercase L
046400	115	77	M	Uppercase M
047000	116	78	N	Uppercase N
047400	117	79	O	Uppercase O
050000	120	80	P	Uppercase P
050400	121	81	Q	Uppercase Q
051000	122	82	R	Uppercase R
051400	123	83	S	Uppercase S
052000	124	84	T	Uppercase T
052400	125	85	U	Uppercase U
053000	126	86	V	Uppercase V

Octal Value: Left Byte	Octal Value: Right Byte	Decimal Value	ASC Abbreviation:	Comments:
053400	127	87	W	Uppercase W
054000	130	88	X	Uppercase X
054400	131	89	Y	Uppercase Y
055000	132	90	Z	Uppercase Z
055400	133	91	[	Opening bracket
056000	134	92	\	Reversing slant
056400	135	93	]	Closing bracket
057000	136	94	^ or ↑	Circumflex, up-arrow
057400	137	95	_, UND, BKR	Underscore, back arrow
060000	140	96	`, GRA	Grave accent
060400	141	97	a, LCA	Lowercase a
061000	142	98	b, LCB	Lowercase b
061400	143	99	c, LCC	Lowercase c
062000	144	100	d, LCD	Lowercase d
062400	145	101	e, LCE	Lowercase e
063000	146	102	f, LCF	Lowercase f
063400	147	103	g, LCG	Lowercase g
064000	150	104	h, LCH	Lowercase h
064400	151	105	i, LCI	Lowercase i
065000	152	106	j, LCJ	Lowercase j
065400	153	107	k, LCK	Lowercase k
066000	154	108	l, LCL	Lowercase l
066400	155	109	m, LCM	Lowercase m
067000	156	110	n, LCN	Lowercase n
067400	157	111	o, LCO	Lowercase o
070000	160	112	p, LCP	Lowercase p
070400	161	113	q, LCQ	Lowercase q
071000	162	114	r, LCR	Lowercase r
071400	163	115	s, LCS	Lowercase s
072000	164	116	t, LCT	Lowercase t
072400	165	117	u, LCU	Lowercase u
073000	166	118	v, LCV	Lowercase v
073400	167	119	w, LCW	Lowercase w
074000	170	120	x, LCX	Lowercase x
074400	171	121	y, LCY	Lowercase y
075000	172	122	z, LCZ	Lowercase z
075400	173	123	{, LBR	Opening (left) brace
076000	174	124	|, VLN	Vertical line
076400	175	125	}, RBR	Closing (right) brace
077000	176	126	~, TIL	Tilde
077400	177	127	DEL	Delete, rubout

Published by : M. R. Balakrishnan Head, Library & Information Services Division
Bhabha Atomic Research Centre Bombay 400 085