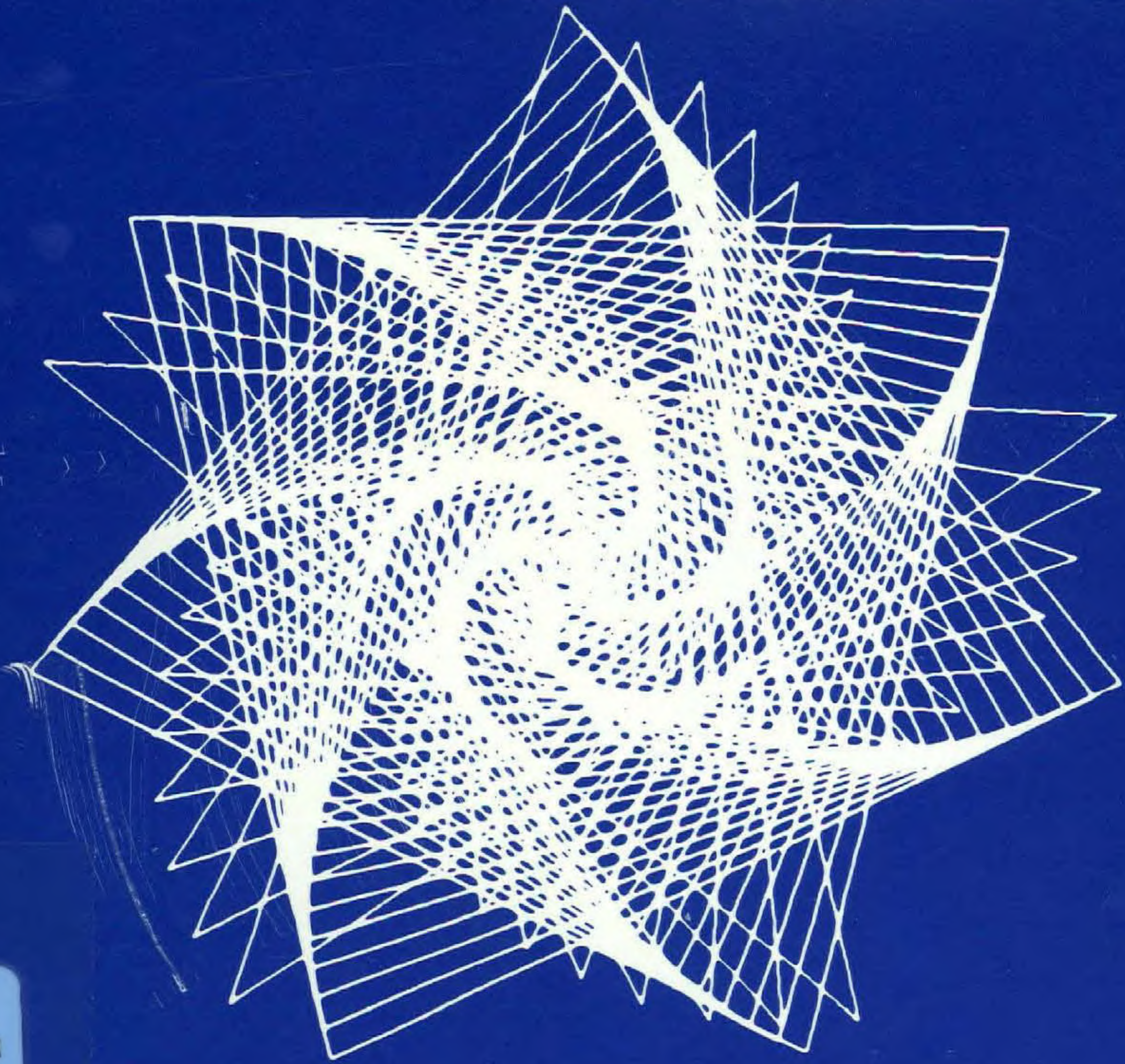
A Study

For The

Department of Communications (DOC)

New Teletext Services (NTS): Report #3

RECORD FORMAT RECOMMENDATIONS



orpak corporation

A Study

For The

Department of Communications (DOC)

New Teletext Services (NTS): Report #3

RECORD FORMAT RECOMMENDATIONS

==

NORPAK Corporation

A Study

For The

Department of Communications (DOC)

New Teletext Services (NTS): Report #3

RECORD FORMAT RECOMMENDATIONS

Submitted In Partial Fulfillment Of Contract

DSS File # 1ER.36001-4-1979

Authored by:

Robert J. Fitzgerald, Manager, Hardware Systems

Document No.	DOC35084
Date of Issue	850323
Revision	FIRST RELEASE

NORPAK Corporation

10 Hearst Way
Kanata, Ontario
Canada K2L 2P4

==

TABLE OF CONTENTS

1. INTRODUCTION....................1
1.1. Background....................1
1.2. Record Format Recommendations....................1

2. NABTS PROTOCOL EXTENSION REQUIREMENTS SUMMARY....................2

3. FORMAT RECOMMENDATIONS....................3
3.1. Seven and Eight Bit Data....................3
3.2. Error Protection....................4
3.3. Service Address....................5
3.4. User Address....................5
3.5. Clear and Cipher Text....................7
3.6. Key Distribution....................8
3.7. Session Type....................9
3.8. Frame Identification....................9
3.9. Request Channel....................10

4. SUMMARY....................12
4.1. Framing Code....................12
4.2. Error Protection....................12
4.3. Data Group Header....................12
4.4. Record Header....................12
4.5. Header Extension Field....................13
4.6. Key Format....................13

1. INTRODUCTION

1.1. Background

NORPAK Corporation has been contracted by the Department of Communications to perform a study in which new teletext-based services are examined, and to make specific recommendations for any extensions that maye required of NABTS.

The only recommendations for incorporation into NABTS will be those which can be shown to be manifestly essential to either the functionality, performance, cost or some combination of these to implement a viable service.

The complete study and recommendations are be contained in four documents which address the following topics (grouped according to the contents of each document):

* analysis of applications and derivation of their technical requirements consolidated for all of the applications examined;

* quantification of the major technical requirement groups and consideration of alternative design approaches with particular emphasis on their implications with respect to the viability of the services (applications) and possible extensions to the basic NABTS specification;

* specific recommendations for extension to the current NABTS definition; and

* an examination of the possible design approach alternatives to the implementation of the new services (using the recommended extensions to NABTS).

This document, the "Record Format Recommendations" Report, represents the third component of an overall study to address the analysis of several new teletext-based services (applications).

1.2. Record Format Recommendations

The "Technical Requirements Analysis" report identified a list of system requirements which must be met in order to support the services discussed in that report. There are implications for the NABTS protocol associated with some of these requirements.

The purpose of this report is to define a set of NABTS record formats which allow the various components of a teletext system to support the services defined in the Technical Requirement Definition.

2. NABTS PROTOCOL EXTENSION REQUIREMENTS SUMMARY

The Technical Requirement Definition stated a number of requirements in system terms. This section will restate those requirements in terms of the functions which must be included in the protocol in order to support the system level requirements.

The technical requirements which must be supported in the NABTS protocol, and for which there is currently no specific provision, include:

* the ability to differentiate between 7 bit and 8 bit data;
* a means of specifying the type and level of error correction which is in use;
* a provision for a service address identification;
* a provision for a user address identification;
* a means of determining whether the data contained in any record is encrypted or not;
* a means for transmitting keys for access to the specific services to individual users;
* a means for identifying the type of session to which a record belongs - in order to support non-roman alphabets and telesoftware;
* in order to support the video "film strip" (VFS) function, there is a requirement for a rapidly detectable frame number; and
* in order to support the hybrid interactive/teletext services, there is a need for a "cross-channel transmit macro" function which enables page linkages to be defined through one communication channel and executed through another.

3. FORMAT RECOMMENDATIONS

In this section we will examine the alternatives for the coding of the functional capabilities mentioned in the previous Section. Each section will state any fundamental assumptions which are made in arriving at a decision. We will examine each of the possible alternatives with reference to its ability to perform the required function, the complexity of its implementation, and compliance with good practices of data communication protocol design such as the ISO reference model for open system interconnection.

3.1. Seven and Eight Bit Data

The identification to the decoder of the presence of either 7 or 8 bit data is fundamentally a link layer function, in that it defines the size of the data word and restricts the available forward error correction options. The selection of the 7 or 8-bit data format is often viewed as a session layer function. In the establishment of a session between two interacting entities, one of the parameters which can be set up for the link layer is the word size. Thus it is only viewed as a session layer function to the extent that two communicating entities may choose to agree on the use of a data format for a given session.

One of the interesting locations which could have been considered for the selection of 7-bit or 8-bit data word would have been the data Group Type (GT). There are two reasons, however, why this is not a fortuitous choice:

* The first is that the selection of the word size limits the available error correction options; however, by the time GT is detected by a receiver, the error correction level has been set by bits 6 and 8 of the Packet Structure (PS) byte.

* Secondly, data bridges which function at the network layer do not examine the GT or subsequent bytes; they base their decision on how to error correct a packet on the prefix alone.

For the above reasons it is desirable to locate the word size selection in the packet prefix. There are two ways in which this can be done:

* the first, is to assign a specific Byte Sync code for use with 8-bit data; and

* the second, is to make use of an otherwise unused bit in the Packet Structure (PS) byte.

Bit 4 of PS is supposed to indicate whether or not a packet is full of useful data. However its raison d'etre is almost purely historical. All known existing encoders always set this bit to Ø, and FSS decoders treat the bit as a "don't care" situation (since its use has never been clearly defined). For this reason, this bit could be used as an indicator of 7-bit or 8-bit mode. This would be consistent with the presence in PS of bits 6 and 8, which specify the size of the error correction suffix.

From the point of view of elegance of the protocol, PS is the most appropriate location for the 7 or 8 bit selection. However since this bit is ignored by FSS decoders, this does not provide a method of ensuring that the FSS decoder does not attempt to access and apply inappropriate error correction to 8-bit data. For this reason the only possible method of indicating that a packet contains 8-bit data is to assign a specific Byte Sync code to it. The recommended Byte Sync code is 1ØØØØ1ØØ.

3.2. Error Protection

All of the facilities for the support of the Product/C/Bundle code for 7-bit data are already included in the NABTS, in the form of bits 6 and 8 of PS. The combinations

(b8,b6) = ØØ and Ø1

refer to the cases were there is no code, or a Product code, respectively. The case

(b8,b6) = 1Ø

has been set aside for a 2 byte code. It is recommended that the C code be adopted for this case. The combination

(b8,b6) = 11

has been set aside for a check packet. It is recommended that the Double Bundle code be adopted for this check packet.

For 8-bit data the codes could take the following meanings:

(b8,b6) = ØØ:	no suffix
(b8,b6) = Ø1:	packet level code such as Reed-Solomon or NHK majority logic decodable code.
(b8,b6) = 1Ø:	reserved for future standardization
(b8,b6) = 11:	check packet for a "Bundle" code for 8-bit data

This is the most appropriate location for the bits selecting the error correction code, since they are already defined there for the 7-bit data format.

3.3. Service Address

The NABTS presently defines the equivalent of a service address. This is known as the Packet Address and it is represented by bytes P1, P2 and P3 of the packet prefix. This defines 4096 possible service addresses which should be more then ample for any of the services contemplated in the Services and Requirements report. There is therefore no need for any new protocol information to support the concept of a service address.

We note that this and the discussion of the previous section imply that the Hamming code used to protect prefix bytes should be maintained when using the 8 bit data format. This enhances the symmetry of the protocol and maximizes the commonality between the 7-bit and 8-bit services. It also has considerable practical value since receivers and data bridges which include provision for demultiplexing data based on the Hamming-decoded packet address can be adapted more easily to the use of 8- bit data. This principle will be extended to the Hamming encoding of the bytes in the data group and record headers. Thus the entire existing data group header (GT = Ø) and record header (RT = Ø,1,2 and 3) can be used without modification with both 7 bit and 8 bit data.

3.4. User Address

The NABTS protocol as defined includes a field in the record header known as the record address which can, without any loss of generality, be used as a user address. Although this has been traditionally viewed as a "page address" there is nothing preventing the use of this field as a user address. Thus user Ø would access page Ø, user 1 would access page 1, and so on.

There are however, two compelling reasons for studying the use of an alternate method of user addressing:

* First, the great flexibility provided by the variable-length record header defined for use with GT = Ø implies a significant processing overhead and a restriction on data throughput.

* Secondly, there is considerable information in both the data group and record headers which is not required for services other than broadcast videotex. This extra information represents a loss of channel capacity which, to a service provider deriving revenue from the service, translates directly into a loss of revenue.

For this reason we should examine the possibility of using shortened and simplified data group and record headers.

Given that the intent here is to augment (if necessary) the NABTS rather than replace the existing structures, it is possible to design a shortened data header as

(GT=1),GC,S1,S2,F1,F2.

In this sequence, each of the bytes has the same meaning as defined in NABTS Article 4.2.1. Note that the "data group repetition" and "data group network routing" bytes have been eliminated since they do not have any clearly defined use. We note also that the "data group continuity byte", which is not useful in existing NABTS services, is retained here because it will be used as the principal means of sequencing records.

In keeping with the fundamental principles of the NABTS, the data group header is followed immediately by a record header which consists of the Record Type (RT) byte and then an optional address. The following values can be defined:

* RT = Ø identifies broadcast data with no user addressing with the format

  RT=Ø, DATA.

* RT=1 defines the user-addressed service with the format:

  RT=1, user address (A1-A9), DATA.

As we shall see later, other values of RT can be defined for transmission of data decryption keys, frame identification codes and so on.

The above defined record format for GT = 1 is an efficient and easily decodable header for use in a record oriented service. Such services are frequently known as virtual connection services. Most packet-oriented communications protocols provide a lower level, known as a "datagram" service, which simply guarantees that packets of data are delivered as sent. It would be useful to include such a feature in the NABTS protocol although there is no certainty that any of the services discussed in the "Applications and Requirements" report require this class of service.

A service which consists of delivery of individual packets of data, without any information as to how these packets may be linked together, can be identified in a number of ways:

* For example a specific value of group type (say GT = 8) could be assigned to indicate that the following packet is not part of a record.

* Alternatively, B4 = 1 in PS could signal this.

* Thirdly, certain illegal combinations of packet structure bits could serve this purpose. For example the combination

$$b2 = b6 = b8 = 1$$

identifies a synchronizing packet which contains only check bytes, which clearly has no meaning. This combination could be used to identify a packet which is not part of a record.

No specific recommendation is made here, since there is no identifiable requirement for this service. These options are mentioned for future reference only.

3.5. Clear and Cipher Text

There is a need to identify to the receiver which records are transmitted in clear form and which are ciphered. This prevents spurious and unpredictable behavior when a ciphered record is inadvertantly processed without decryption.

Let us examine the case GT=Ø first. What is needed is a flag in the record header which, if set to 1, causes FSS decoders to reject the record. One possible location for this bit is in the Y21 byte, since NABTS Article 8.5.2.7.1. specifies that FSS receivers must ignore any records containing YØ2 and subsequent bytes. However, this location has the following disadvantages:

* three full bytes of overhead are added to the record header in order to gain 1 useful bit;

* it lengthens and entrenches the classification sequence whose recursive nature makes it complex and relatively slow to process; and

* it does not appear logical to define a new Y byte when Y11, Y12, and bit 2 of Y13 are not presently defined - these latter bits can not be used for this purpose since the FSS receiver will simply ignore them, rather than reject a record which contains them.

The alternative location for this flag would be in one of the bits of the Record Type (RT). We note that values Ø though 3 have been defined for RT. If bit b8 = 1 is used to indicate that the data in the record is encrypted, this defines four new record types (8, 9, 1Ø and 11) which the the FSS receiver must reject, as stated in NABTS Article 8.5.2.2.6.

The recommended method of distinguishing between clear and cipher text in data groups having GT = Ø is through the Record Type (RT). Values Ø through 3 are the existing record types containing clear text. Values 8 through 11 are the encrypted versions of the records defined for RT = Ø through 3.

For data groups GT = 1 the same method of distinguishing clear from cipher text can be used. Thus when bit 8 of RT is 1, the record contains encrypted data.

3.6. Key Distribution

As seen in the previous report, there is a need for a record format for transmitting keys to individual users. The most efficient method of implementing this is by a single user-addressed record which contains all keys to be changed in that user's terminal. We therefore need a method of indicating that a record contains keys and a technique for delimiting the various keys which may be contained in the record.

Since the keys are in general inherently composed of 8-bit data bytes, there will be no attempt here to define a record format for transmitting keys as 7-bit data.

The most straightforward way to indicate that a record contains keys is to assign a new record type to it. We can assign GT = 1 and RT = 2 to this purpose, except that we note that it is clearly not sensible to transmit keys without encryption. Therefore only the encrypted version of this Record Type is meaningful, and record type 10 will be defined to contain decryption keys. The format of this record is as follows:

RT = 10, User Address (A1-A9), KEYS.

Let us assume that each key consists of 8 bytes. Each entry in the sequence of keys can then consist of the following 10 byte sequence:

SM, SL, 8 key bytes.

The composition of the SL byte is as follows:

$$SL = 16\ P2 + P3,$$

where P2 and P3 are the two least significant digits of the packet address. SM contains P1 in its least 4 significant bits, and P1 is the most significant digit of the packet address. When the most significant bit of SM = 1, there are additional keys in the sequence. If the most significant bit of SM = 0, this is the last key in the sequence.

There is a vast number of ways in which a format for transmitting keys could be defined. The only particular virtues of the format recommended here are that it is relatively simple to process and uses a chaining technique which is consistent with that used elsewhere in the NABTS.

3.7. Session Type

The purpose of including a Session Type in the protocol would be to allow a receiving terminal to distinguish between different types of applications and, in particular, to reject applications which it is not capable of processing. This includes, but is not limited to, the two following applications, for which there is an immediate requirement here:

* software programs; and

* non-roman alphabets such as Chinese, Japanese or Arabic.

It would be both difficult and of doubtful value to define a session level filter for each of the possible alphabets or each of the possible software languages, operating systems or target computers. It is assumed that this detailed differentiation is performed by presentation level coding. The purpose of the session type is to identify a general application such that the receiving terminal knows whether it has any possibility of processing it.

The proposed method of providing this application differentiation is again through the use of the Record Type (RT). For both Group Types Ø and 1, Record Types Ø through 3 have been defined. We can define RT = 4 to identify messages containing characters of a non-roman alphabet. We can similarly identify Record Type 5 to contain telesoftware programs. This leaves record types 6 and 7 for future definition.

The alternative method of coding this function, such that new services such as software or non-European languages would not be accepted by an FSS decoder, involves the use of Y2 bytes as discussed in Article 3.5. For the same reasons as were given in that section, this coding method is not recommended here. The use of the record type is both simpler and more efficient.

3.8. Frame Identification

For support of the video "film strip" (VFS) function we need a method of transmitting a packet which contains a code identifying the contents of the visible portion of the video field.

We could simply define another group type which contains a frame identification which, for consistency with other addressing, could consist of 9 digits. However, we note that this produces a packet which is largely empty, since there is no data transmitted with a frame identification. The data is, in a sense, the video signal in the visible portion of the frame. Transmitting one full packet containing a frame identification along with each field of video therefore is not an effective use of the channel capacity.

We can obtain a significant increase in efficiency by transmitting a frame identification packet at the beginning of each video sequence and including in that packet the number of TV frames contained in the video sequence.

Since the frame identification is a fixed-format single-packet record the S1, S2, F1 and F2 fields of the data group header have no function in this application. The frame identification packet should therefore be identified by a new data group type and the format for a frame identification packet is as follows:

(GT=2),Frame Identification (A1-A9),N1,N2

In this sequence, N1 and N2 contain the 4 most-significant and 4 least-significant bits, respectively, of an 8-bit number representing the number of frames (following the present one) which belong to the sequence identified by the frame identification code. This allows approximately 8.5 seconds of continuous video to be transmitted before it becomes necessary to transmit another frame identification packet.

This coding is up to 256 times as efficient as transmitting a packet with each frame. The improvement is a function of the average video sequence duration.

3.9. Request Channel

Hybrid videotex/teletext services, such as the catalogue shopping service supported by the video "film strip" (VFS) function, require a means of requesting, through an interactive channel, that content be delivered through the video channel.

Although such requests could be issued by the user, it would clearly be more desirable to have the request formulated for him/her by the terminal, in response to a simple action (such as pressing the "more" key on the keypad).

One desirable method of creating this linkage would be to define a new kind of transmit macro which uses the alternate request channel. However, there does not appear to be any way of coding this so that FSS decoders will ignore it.

The only plausible method is to define a new Header Extension Field for "reverse channel request". Referring to NABTS Article 5.2.8.2, this field could have

EI (b6,b4,b2) = 1Ø1.

After reception by a suitably-equipped decoder of a record containing this Header Extension Field, the invocation of the "More" function by the user causes the decoder to transmit the contents of the field through the request channel, and begin a search for the record in the teletext signal.

This method is quite general, in that it allows linked access to both traditional NABTS records (GT = Ø and 1) and video "film strip" (GT = 2).

4. SUMMARY

This section summarizes the format recommendations contained in this report, in the order of the IS-14/CVCC-TS1ØØ sections to which they pertain.

4.1. Framing Code

(Reference IS-14/CVCC-TS1ØØ: Article 2.2.3.)

* Use BS = 1ØØØØ1ØØ to identify 8-bit data services.

4.2. Error Protection

(Reference IS-14/CVCC-TS1ØØ: Articles 3.3., 3.4.)

* For 7-bit data, specify the C and Bundle codes.

* For 8-bit data, Reed-Solomon and majority-logic-decodable cyclic codes need further study as a basic packet-level code.

The codeword should include the whole packet as defined in NABTS Article 3.1. A "Bundle-like" Product code must also be studied for use with 8-bit data.

4.3. Data Group Header

(Reference IS-14/CVCC-TS1ØØ: Article 4.2.1.)

* Define new data group headers:

  (GT = 1), GC, S1, S2, F1, F2
  (GT = 2)

4.4. Record Header

(Reference IS-14/CVCC-TS1ØØ: Article 5.2.1.)

* Define RT(b8) as follows:

  RT(b8) = Ø: clear data
  RT(b8) = 1: data following record header is encrypted

* For GT = 1, define the following record types:

Use	RT	Record Header Format
Broadcast	Ø or 8	RT
User-Addressed Data	1 or 9	RT, A1-A9
Key Distribution	1Ø	RT, A1-A9

* For GT = 2, the record header format is as follows:

A1-A9, N1-N2,

where A1-A9 is the frame identification number, and N1-N2 is the number of frames, following the present one, in the video sequence.

* For GT = Ø and 1, define the following record types:

RT = 4: non-roman alphabet session
RT = 5: telesoftware session

4.5. Header Extension Field

(Reference IS-14/CVCC-TS1ØØ: Article 5.2.8.2.)

* To define a Header Extension Field containing a number to be sent through a secondary request channel when the user invokes the "more" function, the form used is

EI (b6,b4,b2) = 1Ø1.

4.6. Key Format

(Reference IS-14/CVCC-TS1ØØ: Article 7.7.)

* Each entry in the key sequence is defined as follows:

SM,SL,(KS1-KS2)

where:

```
SM(b8) = CHAIN: b8 = Ø means this is the last key
                b8 = 1 means another key follows
SM(b7,b6,b5) = Ø
SM(b4,b3,b2,b1) = P1(b8,b6,b4,b2)

SL(b8,b7,b6,b5) = P2(b8,b6,b4,b2)
SL(b4,b3,b2,b1) = P3(b8,b6,b4,b2)
```

KS1-KS2 is the seed used in calculating the key for decryption of packets having packet address P1,P2,P3.

Customer	
Model	
Serial Number	

10 Hearst Way, Kanata, Ontario K2L 2P4 613-592-4164 Telex 053-4174