*Paul Jarrett joined the BBC Television Service in 1962. He transferred to the Studio Planning and Installation Department in the BBC Engineering Division in 1968, became assistant to the Computer and Automation Developments Engineer in 1970, and was appointed as Head of Computer Systems Unit in 1977. He is currently the Head of Information Technology Engineering.*

*In the international arena, he is a member of EBU Specialist Group G/RC (Remote control of production equipment) and is a member of the SMPTE T14 Committee on Television Video Technology, in which capacity he has chaired a number of Working Groups where he has supported the EBU's contributions to the development of the ESbus.*

P. H. Jarrett

# The ESbus remote control system

## An introduction for prospective users

*The ESbus remote control system for broadcasting production equipment is the fruit of several years' joint study by the EBU and the SMPTE. The aim has been to set a durable standard able to accommodate the full range of operational control requirements in present-day broadcast production systems and incorporating the flexibility needed to cope with future developments. This article introduces the basic features of the ESbus, and explains the specialized terminology, to assist readers in their understanding of the complete specifications.*

### 1. Background

Long gone are the days when every item of equipment under the control of an engineer, producer or tape editor could conveniently be mounted in a single control desk, with every push-button, knob or lever forming an integral part of its associated box of electronic hardware. The miniaturization achieved through the substitution of electron valves by transistors and then by integrated circuits has been outstripped by demands for ever-larger numbers of channels, greater functional complexity and enhanced ergonomics.

Gone too are the days when each production area was equipped permanently with all the apparatus it needed for its particular contribution to programme-making. As the equipment has become more complex — and hence more costly — and as broadcasting organizations have become increasingly aware of the need to function cost-effectively, the trend has been to group together in centralized areas the most expensive equipment items and to arrange for them to work in time-shared mode for a number of different production areas. On a smaller scale, separation of the man/machine interface and the signal processing equipment may also be required within a small studio, simply to ensure a convenient arrangement of units within the space available.

Separation of control panels and controlled equipment implies a need for communications links between them. It was evident that the development of a universal standard for remote-control equipment would be the best means of preventing the introduction of several mutually-incompatible remote-control systems. The SMPTE therefore took the initiative in undertaking studies with the aim of defining a digital remote-control system that would enable equipment from different manufacturers to be interconnected for control purposes.

Similarly, the EBU had set up its own group with the aim of defining a standard which would satisfy the remote-control requirements of the EBU Members. The EBU and SMPTE groups collaborated closely and the result of the initial phase of this work, known as the ESbus standard (from *E*BU/*S*MPTE *bus*), was published in December 1984 as EBU document Tech. 3245 [1]. A set of SMPTE publications gives functionally equivalent specifications [2, 3, 4, 5].

Having established the transparent message-routing system, the joint EBU/SMPTE studies continued with the creation of standards for the messages to be carried. Supplement 1 to document Tech. 3245 [6] describes the "system service messages" needed to ensure the correct internal functioning (housekeeping) of the message routing system, and a set of "common messages" which will be applicable to all types of production equipment operating under ESbus control.

Further supplements containing details of "type-specific messages" are being prepared and issued as dictated by the needs of industry. These are message sets devised for the control of individual equipment types — VTRs, telecines, mixers, etc. — each of which has its own requirements.

A casual glance at the detailed specifications, and more particularly the various Supplements, will reveal that they use a vocabulary and style of presentation related more closely to the user guides for computer systems than to the possibly more familiar style of documentation on video and audio systems. The following sections of this article will introduce this specialized vocabulary, and the concepts it describes, to those unfamiliar with the language of data processing.

## 2. The principles of ESbus

It was decided at an early stage that the standard system should make optimum use of modern digital technologies and, in view of the large numbers of data links likely to be used in a large production centre, that the basic hardware connection should take the form of a "bus" carried over a low-cost 4-wire "balanced" line for full-duplex data transfer. It was also decided that it would be appropriate to design the system in accordance with the general principles of the standard developed by the Inter-

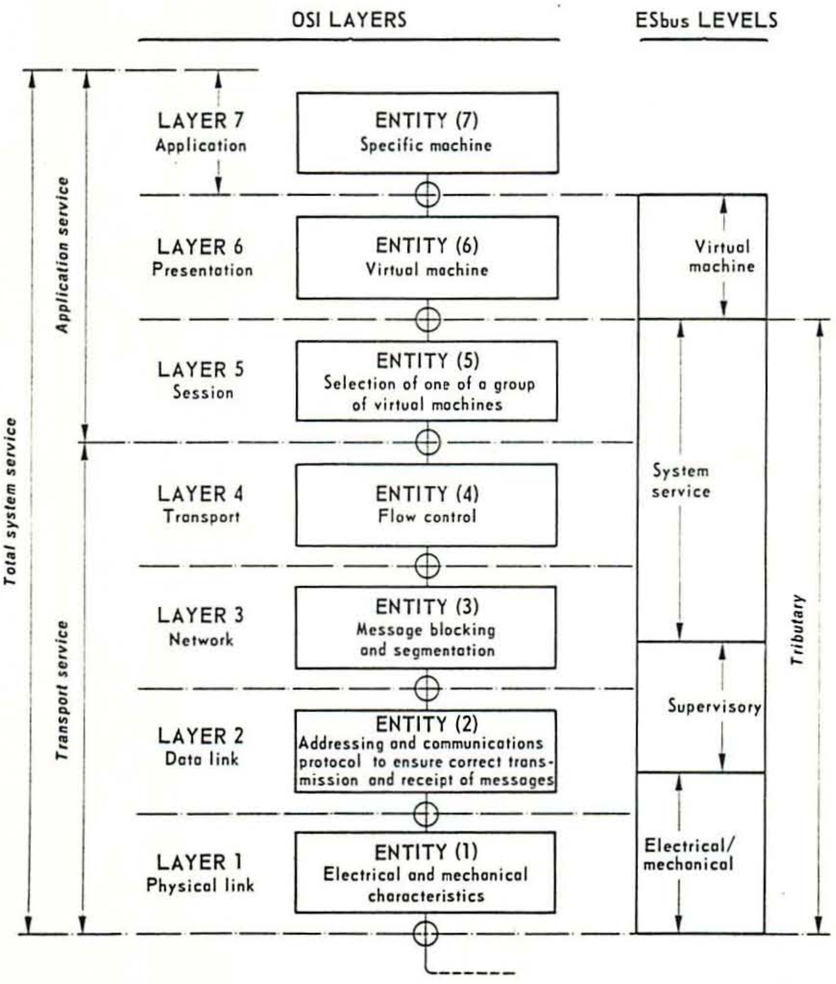


Figure 1
The layers of the OSI model and their relationship to the levels of the ESbus system

national Organization for Standardization (ISO) known as the "Open Systems Interconnection" model (OSI) [7]. The ESbus remote control system is therefore divided into logical levels along the lines of the OSI model, although for reasons of simplicity only four logic levels have been employed, compared with the seven layers of the full OSI model. The electrical/mechanical, supervisory, system service and virtual machine levels of the ESbus system provide, individually or jointly, all the necessary features of the ISO physical, data link, network, transport, session and presentation layers. The correspondence between the ESbus levels and OSI layers is shown in *Fig. 1*.

The system is based also on the concept of distributed intelligence. Each functional unit, whether it be "controlling" (such as a remote-control panel for a telecine) or "controlled" (the corresponding telecine) is attached to the system via an intelligent interface which itself performs most of the "computation" needed by that particular device. Such intelligent units are known as tributaries. Other computations, and in particular those concerned with supervision of communications between all the devices connected in a single system, are performed by a bus controller. A complete functional system, comprising in its minimum configuration two tributaries, one bus controller and the interface bus, is known as a local network.

A fundamental principle of the OSI model, which is retained in the ESbus, is that each layer is transparent to the layers on either side. Hence the communications features must be considered as being totally independent of the control language. Furthermore, each level in the system has an independent language which is relevant — and indeed can be interpreted and acted upon — only within its intended level. Referring to *Fig. 1*, which can represent equally well the logic structure within a controlling tributary or a controlled tributary, the virtual machine level (which is effectively a software representation of the production equipment item connected to it) has a language which cannot be interpreted, or modified, during its transmission through the lower levels. Likewise, supervisory level control characters have no direct influence on any virtual machine: they merely form part of the "packaging" into which the virtual machine control messages are inserted.

Functionally, it is as though controlling and controlled virtual machines are interconnected directly by wires, in one-to-one correspondence (*Fig. 2*). In fact, the messages pass down through the lower levels of the controlling tributary (where they will pick up additional data for routing purposes, for example), along the wires of the electrical/mechanical level, and up through the receiving tributary where the additional data will be stripped off in the lower levels, so leaving the bare data for interpretation by the controlled virtual machine.

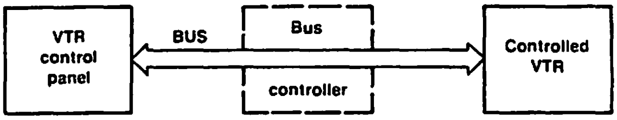


Figure 2
Logical control path

As noted briefly above, the system works in full-duplex mode. Before a command sequence originated at a control panel is regarded as being complete, the relevant command message must be delivered to the controlled tributary and this must have returned a response message indicating to the control panel tributary the action (if any) it has taken on receipt of the command. It is important to bear in mind also that in the ESbus, a command may equally well serve to deliver an instruction or to enquire into the status of a controlled tributary. Also, the message-handling procedure is identical regardless of whether the controlling tributary is sending a command to the controlled tributary, or the latter is sending a response to the controller.

## 3. Supervisory level

### 3.1. Supervisory level structure

Before describing the syntax of the control message language, a brief description of the tributary supervisory level structure is essential. In every tributary this must conform in all respects to the protocol defined in document Tech. 3245, Chapter 3.

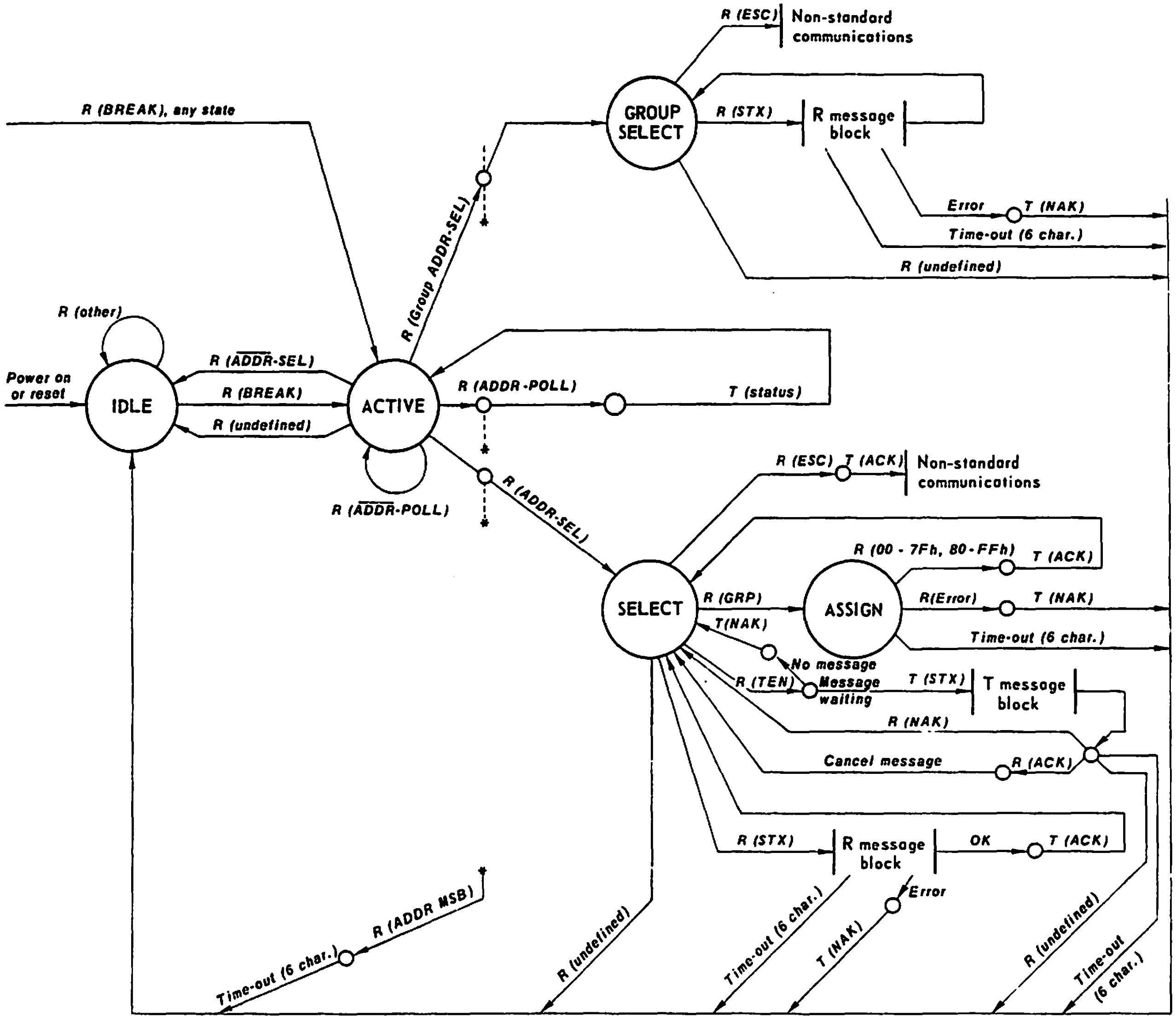
Every tributary is allocated two principal supervisory level addresses — its POLL address and its SELECT address. Tributaries also have a "GROUP — ALL CALL" address for "broadcast" messages and additionally can be allocated one or more of 127 GROUP addresses at the discretion of the user.

The supervisory level can adopt any one of the states shown in the diagram in *Fig. 3*, which indicates the effect of messages on the state of tributaries or of operational equipment. Each state is represented by a circle, and the most important of these contain a label (IDLE, ACTIVE, etc.). All possible transitions between these states are indicated by arrows, labelled with the event which causes the transition.

Changes of state can occur only on receipt of an appropriate supervisory character: these characters are single bytes in the range 00h - 7Fh*. They identify all communication sequences and also provide status information as defined.

---

* The notation XYh is used here to designate hexadecimal values.



* Time-out between address bytes

Figure 3
Tributary supervisory protocol
R: message received T: message transmitted

The IDLE state is entered after "power-on" or "reset"; all tributaries must assume this condition initially.

The ACTIVE state is attained on receipt of a unique "BREAK" synchronizing character from the bus controller. Only the bus controller can issue BREAK, and it has a non-standard length of $20^{-0}_{+1}$ bits which is detected as a sequence of between 17 and 22 consecutive bits in the datastream in the SPACE condition, followed by a subsequent return to the MARK condition. This makes it unambiguous and no other character can be interpreted as BREAK. Any sequence of more than 22 consecutive bits in the SPACE condition is interpreted as an error condition.

From the ACTIVE state, the supervisory level will perform one of the following operations:

- if it recognizes its own POLL address, it will assume a temporary condition in which it responds by transmitting a status character, and then reverts to the ACTIVE state; any other POLL address will be ignored;
- in response to its select address, it will enter the SELECT state. This is the major state in which it can communicate to and from the bus controller, and from which it can be driven into the further state "ASSIGN" which allows it to be allocated to one of the 127 "group addresses" for joint control purposes;
- in response to its group-select address, it will enter the GROUP SELECT state, in which it can

receive further supervisory level characters, but cannot respond (except in an error condition).

In response to any other SELECT address, the tributary will move to the IDLE condition and await the next BREAK character before returning to the ACTIVE state.

The maximum permissible block-length in the supervisory level is 256 bytes. This is the reason for having two special system service level functions known as "blocking" and "segmentation" (see Section 5). Certain bus controller supervisory level services are accessed by system service level commands (Assign Supervisory Level Group for example).

It must be stressed, however, that any system service control message which ultimately affects features in the bus controller supervisory level must be passed initially, and transparently, through the supervisory level to the bus controller system service level parser. (It is the role of the parser to scrutinize the received data stream, split it into its component parts and distribute the various message bytes to the appropriate parts of the decoding system for interpretation and execution.) On receipt of such a message, this parser will instruct the bus controller supervisory level to generate the appropriate supervisory characters in order to perform the required function.

As noted in Section 2, the supervisory level itself cannot *decode* any control message; it produces, and responds to, supervisory level characters only.

### 3.2. Time delay

The time delay through the interface must be taken into account when transmitting any message, as this can become very significant within a polled environment. Polling is a strategy in which the bus controller interrogates each tributary in the local network in turn to discover whether any of them is waiting to be "serviced". The bus controller and communication channel will introduce a delay in any bit sequence transmitted from the control panel to the controlled machine; the absolute delay will be dependent on the time that elapses between a service request flag being raised by the tributary and the next poll request from the bus controller. In the case of a single control panel connected to a single controlled machine, this delay would be dependent solely on the polling frequency, as there would be no other network devices which might also be requesting service. The system would therefore be totally deterministic; messages would be received within a time window whose width is determined by the bus speed, the polling frequency and the length of the particular message. The presence of further tributaries connected to the system would delay receipt of any specific message; the length of the delay would be determined not only by the number of additional polling periods, but also by the requirement for service, if any, of each additional tributary. System designers must recognize this constraint and plan network configurations accordingly.

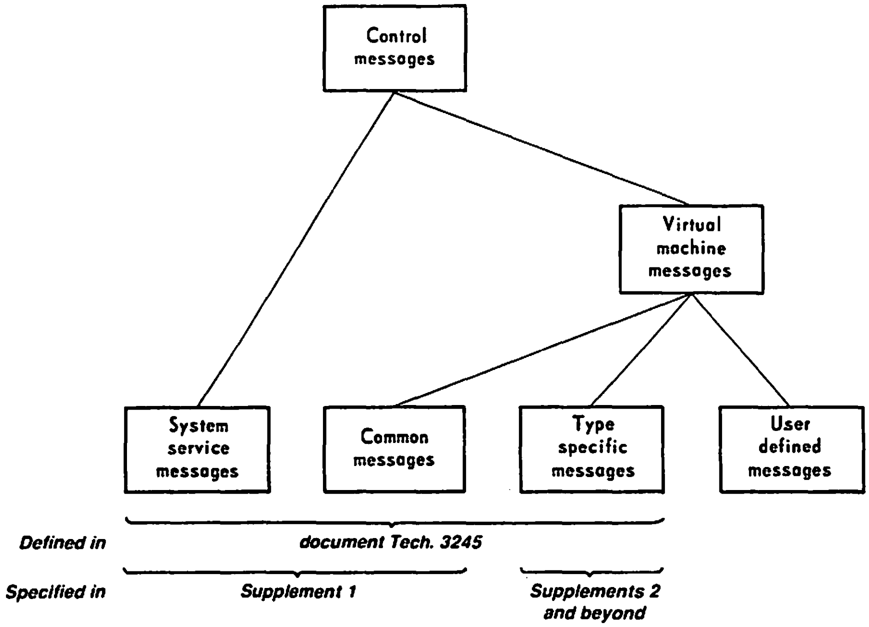


Figure 4
Control message classification

## 4. Control messages

### 4.1. General structure

Control messages are divided into two distinctly different sub-sets — system service messages and virtual machine messages, as shown in *Fig. 4*.

The message structure, in terms of both syntax and semantics, is common to both sub-sets, however, and is described fully in EBU Tech. 3245, Chapter 2.

Every message begins with a KEYWORD which describes the function of the message. Each keyword exists as a plain-English descriptive word or expression and a four-letter mnemonic for man/machine interface purposes, and as a two-character hexadecimal number in a data-stream*. The keyword may be followed by qualifying expressions known as parameters. In effect, the keyword dictates *what* is to be done and the parameters contain auxiliary data or specify the *manner* in which the action is to be executed.

The minimum message length therefore is that of a single keyword: the VTR "STOP" command — 41h — is one example. In other cases parameters are required.

Parameters are constructed in the form of information fields, which are data strings, comprising:

a) An INFORMATION FIELD NAME

- this is a two-character hexadecimal number which, when required, is inserted to follow the hexadecimal keyword within the data stream. (In a manner similar to that for the keyword, it is also described by a plain-English word or expression, and as a four-letter mnemonic for man/machine interface purposes.)

b) A qualifying expression

- this may contain a value, a logical (YES/NO or ON/OFF) function, or an alpha-numeric character string.

Where an information field name is clearly unambiguous, it is deemed to be "understood" by the virtual machine parser, and is therefore omitted from that particular command in order to minimise redundant bus traffic. In such cases the relevant ESbus Supplement defines the parameter solely as the "value" following directly after the keyword. In all other cases the information field name is included.

The contents of information fields are held as arrays of data within each virtual machine. They may be of a form which is fixed for a specific virtual machine (e.g. VIRTUAL MACHINE TYPE, which identifies the sort of machine attached to the tributary) or may be updated by a virtual machine as a result of its own action (e.g. TIMECODE).

Whilst many messages have a fixed number of parameters, others can be of variable length. The virtual machine parser is able to recognize the end of such messages only by the use of either a byte count, or by BEGIN and END delimiters; codes 01h (begin) and 02h (end) are reserved for this purpose. In such cases the information field name must preceed the value of each parameter.

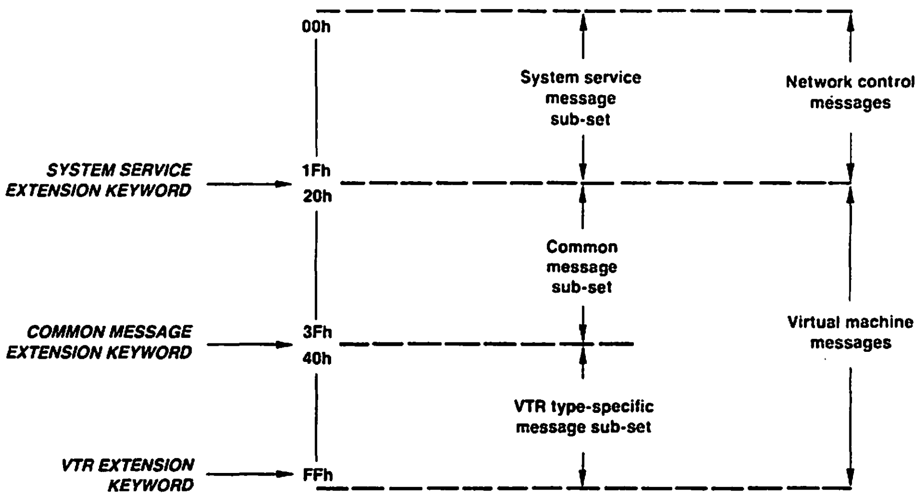


Figure 5
Message code map for video tape recorders

---

* Document Tech. 3245 uses "keyword" almost exclusively to refer to the hexadecimal bytes in the data-stream; the plain-language "keyword" is often referred to as a "command".

### 4.2. Control message coding

Control message keywords are identified uniquely by their hexadecimal codes. A complete message code "map" for a network consisting solely of VTRs is shown in *Fig. 5*.

System service messages can be seen to occupy the range 00h to 1Fh; virtual machine messages having common usage follow in the range 20h to 3Fh and the VTR type-specific messages are contained in codes 40h upwards. In order to accommodate code space for possible future developments, codes 1Fh, 3Fh and FFh, have been reserved as "extension" codes for the system service, common and type-specific sub-sets respectively. Less-frequently-used commands in each sub-set are being allocated to the extension sets from the outset, in order to maximise the amount of "short" (single byte) code space available.

Extensions to each sub-set are permitted over the entire range of available extension set codes: for example, the system service extension set resides in the range 1F00h to 1FFFh and 1FFFh is a further extension code to the range 1FFF00h to 1FFFFFh, and so on. It can be seen therefore that there is no theoretical restriction to the code space available for each control message sub-set.

As the system service and common message sub-sets are applicable to all equipment types, the message code map for a "hybrid" network will be as shown in *Fig. 6*.

This code mapping arrangement explains why the system service and common message sub-sets have been separated from each virtual machine set and are issued as a separate single supplement (Supplement 1) to document Tech. 3245. Type-specific message sub-sets are being issued independently. It will be appreciated, therefore, that document Tech. 3245 plus Supplement 1 and at least one further Supplement together provide the essential information for the design of a complete network.

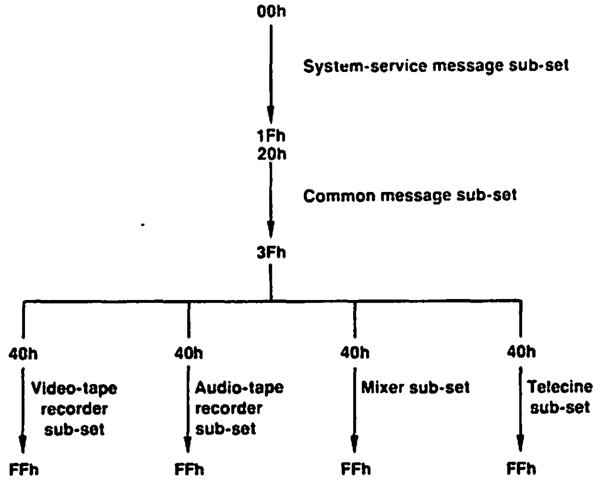


Figure 6
Message code map for a typical hybrid network

### 5. System service messages

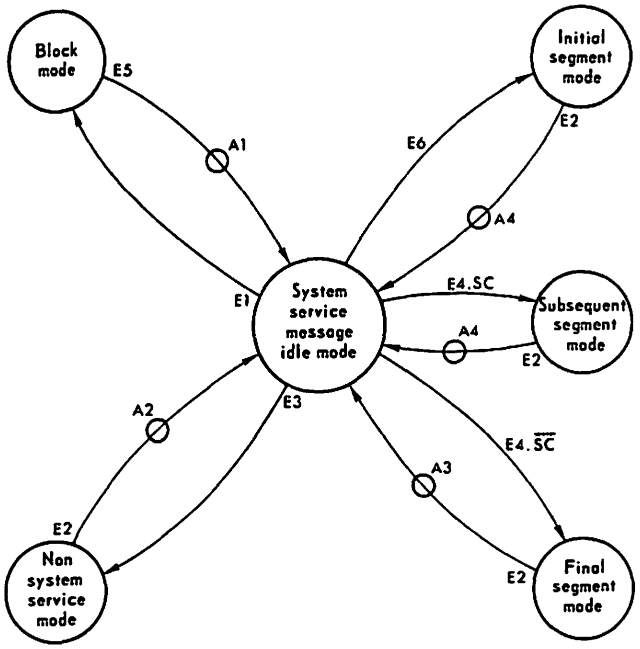
System service messages are also denoted by KEYWORDS, and are explained in Supplement 1 to document Tech. 3245. The system service message sub-set may be directed either to the bus controller or to a tributary. It performs three basic functions:

- Establishment and maintenance of the network relationships between tributaries through the management of entries within the bus controller linkage directory. The linkage directory is in effect a repertory of permitted inter-tributary connections, listed as a set of "virtual circuits". It is created at the start of each working session and may be changed by the human operator to take account, for example, of the withdrawal of an item of equipment from service for repair.
- Maximization of the transmission path efficiency by "blocking", in which the start and the byte count of short virtual machine messages are marked, thereby allowing them to be concatenated within a single transmission (supervisory level) block.
- Dis-assembly and re-assembly of long messages which, owing to the maximum message length of 256 bytes in the supervisory level, have to be broken into "segments" of shorter length for transmission purposes.

The linkage message ASSIGN LINKAGE used in the first of these functions can be regarded rather like a dialled telephone number; it establishes the connection between two subscribers (tributaries), but has no bearing whatever on the content of the subsequent conversation (control message exchange). It nevertheless originates from the user of the system.

Similar messages are provided to assign tributaries to supervisory level groups, and virtual machines to virtual machine groups, for joint control purposes, and also to break down linkage assignments established previously.

Message "blocking" and "segmentation" are system service level functions provided by the system designer. They are transparent to the virtual machine and require no action on its part. On receipt, system service messages are executed only within the system service level; they will never be passed on to the virtual machine by the system service level parser (see *Fig. 7*). In the case of linkage messages, the information they contain may nevertheless originate from a human operator and be passed down through an appropriate virtual machine (an assignment station, for example).



Events

E1 = Block keyword
E2 = Last byte of supervisory level message
E3 = Keyword not system service message : i.e. ≠ Block : ≠ Segment
E4 = Subsequent segment keyword
E5 = Last byte of block data
E6 = Initial segment keyword

Conditions

SC = Segment count ≠ 0
$\overline{SC}$ = Final segment : segment count = 0

Actions

A1 = Pass data block transparently for higher level parsing
A2 = Pass data transparently, no parsing required on message level
A3 = Pass concatenated segments transparently
A4 = Store incoming segment

Figure 7
State diagram for the segmentation and blocking processes

## 6. Virtual machine messages

As noted earlier, each type-specific virtual machine is furnished with a dedicated set of type-specific control messages. The virtual machine is considered to be a "state" machine, in that it mimics the current state of the item of production apparatus to which it is connected.

Certain virtual machine messages can change its state. Some are mutually exclusive (the VTR tape motion commands STOP and PLAY for example). Other virtual machine messages, termed "common" messages, have general application and are concerned primarily with information transfer; they cannot directly change the state of any virtual machine.

### 6.1. Common messages

In all the message sets, the common messages are contained in the range 20h - 3Fh. The functions of some of them will be described individually.

#### *6.1.1. Information field data-exchange messages*

As seen in Section 4.1, the information fields hold essential operational and status data within the virtual machine. These data can be interrogated remotely over the network by the use of the common message keyword READ followed by the specific information field name.

For example: (READ) (VIRTUAL MACHINE TYPE) or (READ) (TIMECODE)

> will cause the tributary to respond immediately with the VIRTUAL MACHINE TYPE (which remains fixed) or its up-dated TIMECODE.

The CYCLE command demands the periodic transfer of the contents of the named information field, at the specified time interval.

For example: (CYCLE) (HOURS) (MINUTES) (SECONDS) (FRAMES) (TIMECODE)

> will cause the tributary to send the instantaneous value of TIMECODE regularly at a time interval specified by HOURS, MINUTES, SECONDS AND FRAMES.

The UPDATE command requires the transfer of the contents of the named information field whenever its value changes.

For example: (UPDATE) (TIMECODE)

> will cause the virtual machine to report the new value, at each change of TIMECODE.

#### *6.1.2. Procedures*

The mechanism employed to define a complex function (an edit for example) is the "procedure". A procedure, as may be deduced from its computer programming counterpart, is a series of commands which are to be performed consecutively in response to a call to "execute".

Such a sequence may be for immediate action, or may be used in the deferred mode, by defining the action as taking place at a specific timeline event (see section 6.1.3), or by giving the procedure a name and commanding the execution of the procedure bearing that name at some later time or times. The system of procedures takes full advantage of the "intelligence" resident within each virtual machine; the degree to which this intelligence is exploited is decided by the tributary/network designer, according to the needs of the particular system.

A number of messages, defined within the common message sub-set, permit the establishment, use and removal of procedures.

DEFINE PROCEDURE is followed by a list of all the commands requiring sequential execution under a single "call".

A PROCEDURE NAME is given as part of the process of defining the procedure, and allows single or repeated execution by simple reference to this name. Once established, a procedure will reside within the virtual machine until a common reset CRESET or an individual DELETE PROCEDURE message is received.

A resident procedure is called by the (EXECUTE PROCEDURE) (PROCEDURE NAME) command.

### *6.1.3. Timeline*

In order to guarantee the coincidence of two or more time-dependent operations, some form of time reference is needed.

The timeline is a timing reference source which is maintained within each time-sensitive virtual machine; it enables time relationships between individual "events" to be specified in advance, each event being related to a specific time occurrence on the timeline.

For many purposes the bus controller may be considered as the "keeper of the clock", and hence as the timeline synchronizing source. Using the supervisory level GROUP ALL CALL address, it can distribute a common "time-stamp" to all the tributaries, which are then maintained in synchronism by means of an external physical feed of clock pulses (television field-blanking pulses for example) which is supplied to each tributary's timeline generator. The general arrangement is shown in *Fig. 8*.

A typical situation in which the timeline would be used to maintain complex relationships between time-critical functions is tape editing, where the relative timings of the entry and exit points on individual tracks will be specified in advance, in terms of reference timeline time.

Common messages are provided to:

- select the source of the timeline (internal/external),

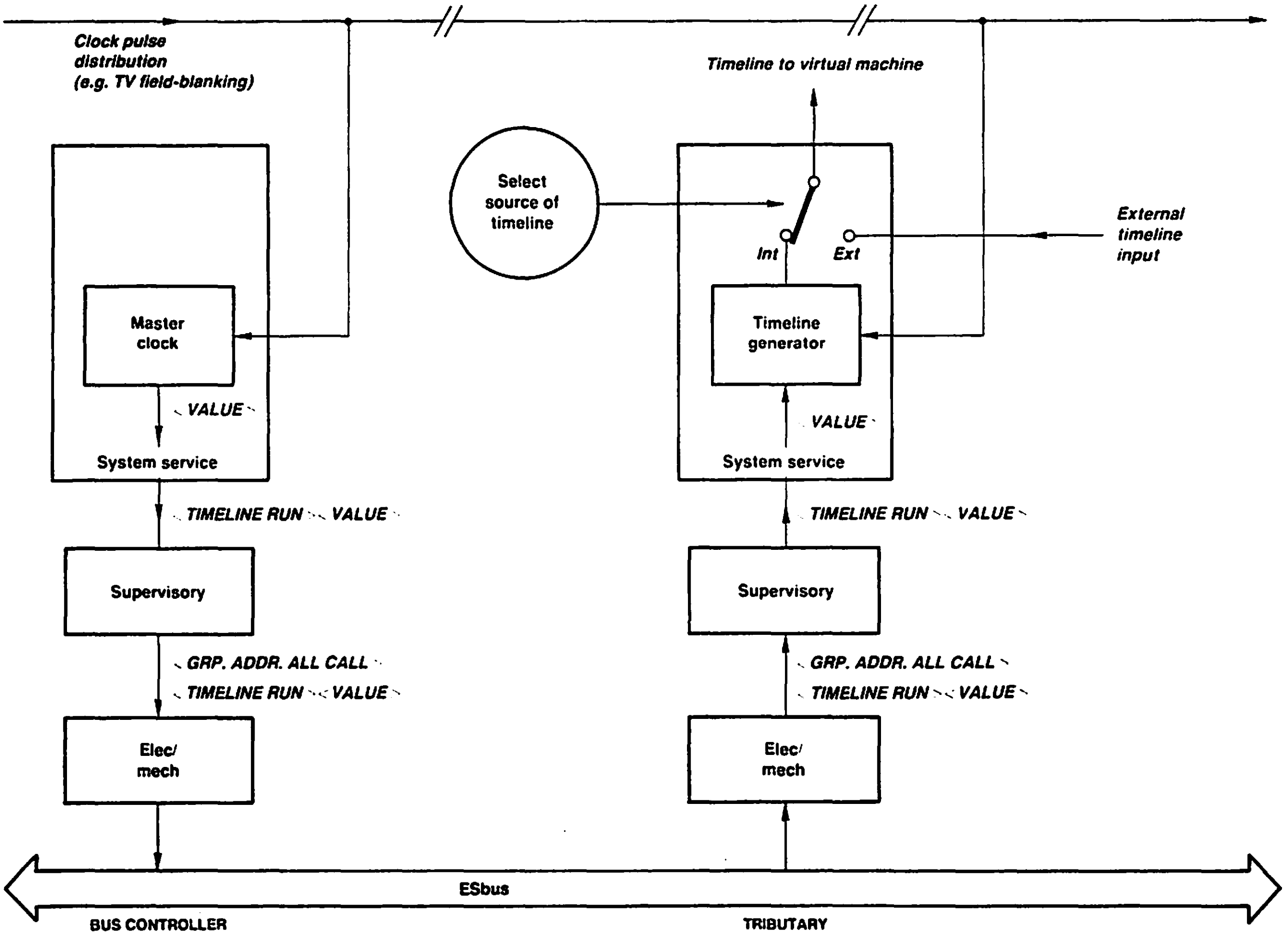


Figure 8
Timeline generation

- cause the timeline to start incrementing from a specified time,
- cause the timeline to stop incrementing.

### 6.2. Type-specific messages

Messages which relate to the functions of specific types of virtual machine are called "type-specific messages". A separate sub-set — a dialect — is to be provided for each virtual machine type and this will be found in the command range 40h to FFh (see *Fig. 6*); in common with the system service and common message sub-sets, FFh is reserved as an "extension" keyword.

The type-specific message sub-sets are currently being compiled by ad-hoc sub-committees representing manufacturing and user interests, each working under the guidance of a non-partisan chairman.

The commands being specified cover the greatest possible range of functions in general use in each case. The degree of refinement in the control message sub-sets may be gauged from an examination of Supplement 2 to document Tech. 3245 giving the type-specific messages for VTRs. Account is taken, for example, of the fact that there will be differences in behaviour between an "ideal" VTR and a real one (and furthermore that no two real machines will function identically). Apart from ordinary commands such as track and channel selection, tape control (normal speed, shuttle, reverse, etc.) and record/play switching, there is provision for various methods of synchronizing the machine, for differing tape transport run-up times, control of the colour framing (2, 4 or 8-field locking to suit NTSC, SECAM, PAL), a variety of tape codes (timecodes, tape timers), etc.

It is recognized, however, that control functions are in a continuing state of evolution; in any high technology industry new requirements, or further enhancements, appear regularly as manufacturers seek to obtain a larger share of the market. It is essential therefore that provision should be made for the implementation of such new features within the basic command structure; the USER DEFINED message is provided for this purpose.

On receipt of a USER DEFINED keyword, the virtual machine parser will pass the entire message, unchanged, to the specific machine which it serves. Thus, new features can be implemented by any manufacturer within the framework of the remote control specification without prior reference to any other party. Security of industrial development is thus assured.

If, at a later date, and in response to market forces, the new features become commonly employed throughout the industry, a specific command or commands may then be introduced, by agreement, to permit their generalised application.

## 7. Local network interconnection

Remote control networks of the type described above are considered primarily to be for localized use; within a video-tape editing suite, for example.

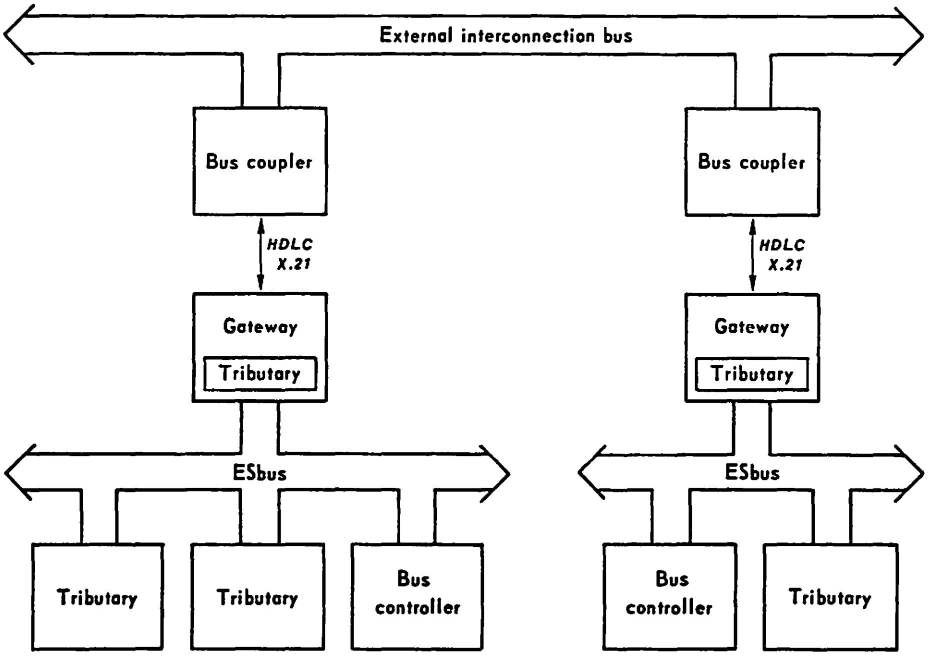


Figure 9
Interconnection of individual local networks

Leaving aside possible organizational constraints on the dimensions of individual local networks, it should be borne in mind that there is a nominal limit of 1200 m on the maximum length of the 4-wire data transmission line.

Where control over greater distances is necessary the signals on the ESbus local network must be transferred to another transmission protocol functioning on a separate interconnection bus. This is also necessary where the interconnection of local networks is required. The interface between each local network and the coupler feeding the interconnection bus is called a gateway and is another specialised form of tributary.

The gateway performs all necessary address conversions and will route all messages to and from its remote host. Its essential features are described in Chapter 4 of document Tech. 3245. In essence, the ESbus specification requires connections to the "outside world" to conform to ISO Standards 3309 and 4335 (HDLC) for functions corresponding to the OSI link layer (see *Fig. 1*), together with a physical layer connection operating according to CCITT Recommendation X.21. *Fig. 9* illustrates the arrangement schematically. It should be noted that whilst the specification defines the link and physical layer standards, it does not define any particular standard for the external interconnection bus mechanism. This is left to the discretion of the system designer.

Where multiple external tributaries are to be connected to a local network, multiple gateways could be provided. However, where message traffic permits, multiple external tributaries could also be connected by multiplexing several tributary addresses onto a single gateway supervisory level, and polling/selecting each one as if it were a specific individual physical connection.

## 8. Illustration of ESbus implementation

It may help to explain the mechanism by which control messages are transferred between tributaries, if we consider a local network comprising:

- a bus controller,
- an individual network assignment station labelled "X" and six other tributaries called, for simplicity, "A" to "F".

The complete network would therefore appear as shown in *Fig. 10*, and we will assume that the VTR control panel, located at tributary "A", requires linkage to the VTR located at tributary "C" for the current working session.

*Fig. 11* traces the detailed control message path between the assignment station (X) and the bus controller linkage directory.

It should be noted here that the virtual machine identifier (in the range 00h to FFh) *must* be appended to each tributary address, although (as in the case shown) this may be the default condition of 00h.

*Fig. 12* illustrates a "snapshot" in the normal running condition of the network, tracing the complete mechanism for the transfer of a VTR "START" command from the control panel to the VTR. The message is transferred initially from tributary "A" to the bus controller and then, following the synchronizing "BREAK" character, to its ultimate destination at tributary "C".

*Figs. 11* and *12* incorporate all the significant ESbus local network features:

- changes of STATE in the supervisory level of the tributaries,
- supervisory-level control character exchanges,

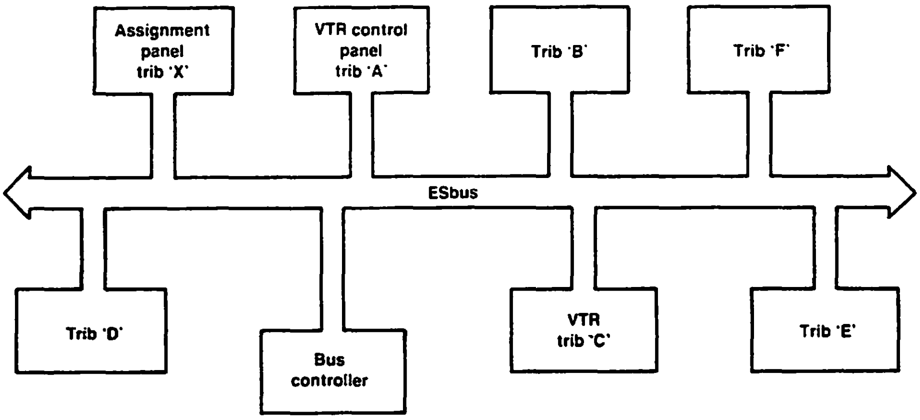


Figure 10
A typical local network

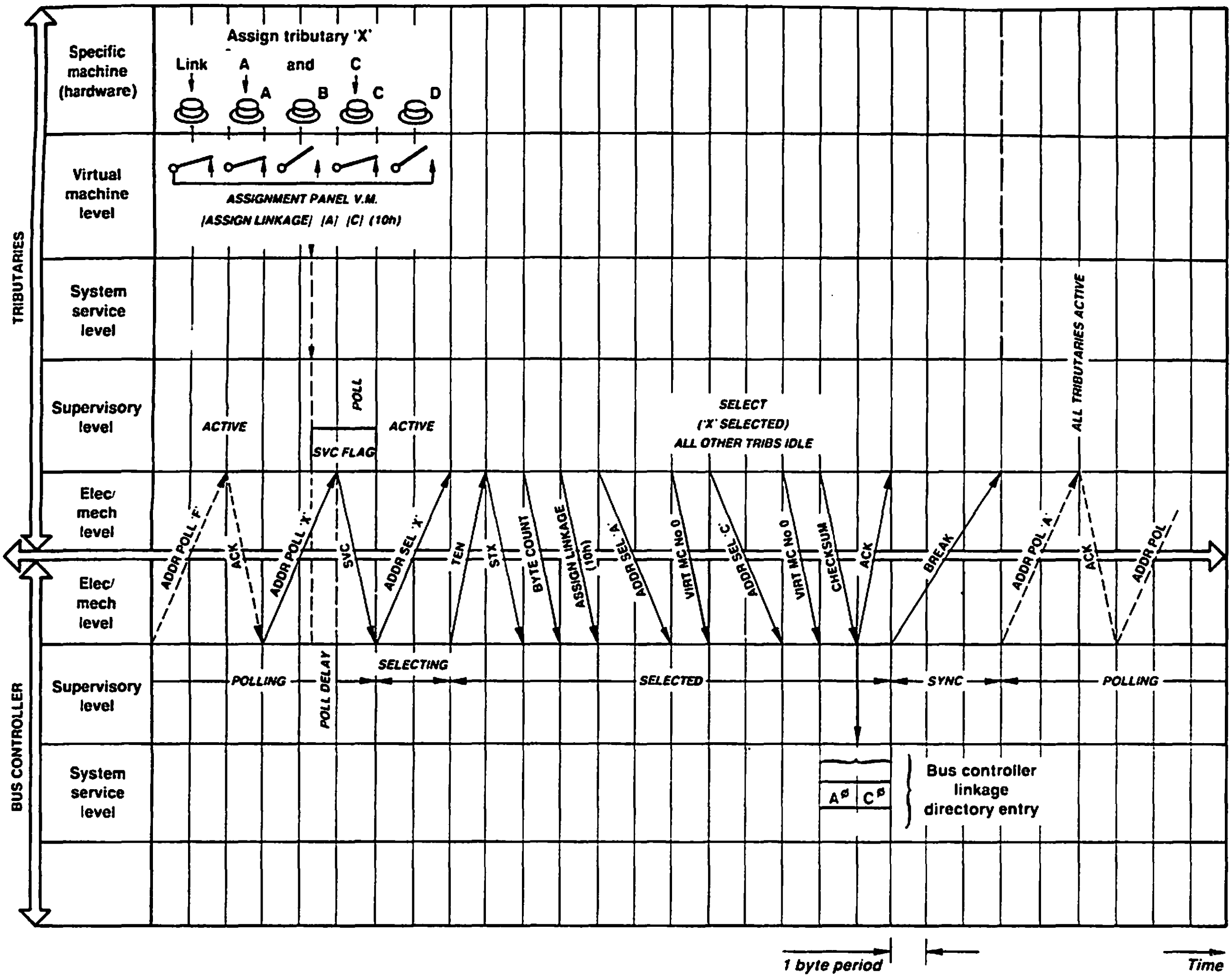


Figure 11
Tributary linkage assignment

- a "poll delay" following the raising of a service request (SVC) flag,
- use of the bus controller linkage directory,
- use of the BREAK synchronizing character,
- the relaying of a control message from one tributary to another by the bus controller.

For simplicity the blocking and segmentation system service mechanisms available to tributaries have not been illustrated; however, these processes are transparent to the control message content, which therefore does not differ from that shown.

## 9. Electrical/mechanical interface

The electrical/mechanical level of the ESbus has been designed to ensure reliable low-cost data transmission, even in the presence of strong interfering fields such as might occur close to broadcast transmitters. Asynchronous bit-serial/word-serial transmission at 38.4 kbit/s is used in a balanced 4-wire bus giving full-duplex communication. The technologies used for this are entirely conventional and follow closely, with minor changes in sensitivities and rise-times, the EIA RS 422 standard. The detailed specification is given in Chapter 5 of document Tech. 3245.

## 10. Experience with ESbus

Having created a standard, the EBU and SMPTE clearly wished to verify that it was satisfactory at as early a stage as possible, and in any event before the corresponding hardware and software was delivered to the market-place. For this purpose, the two organizations conducted joint tests in Europe during June 1985, bringing together equipment developed by eight different manufacturers and

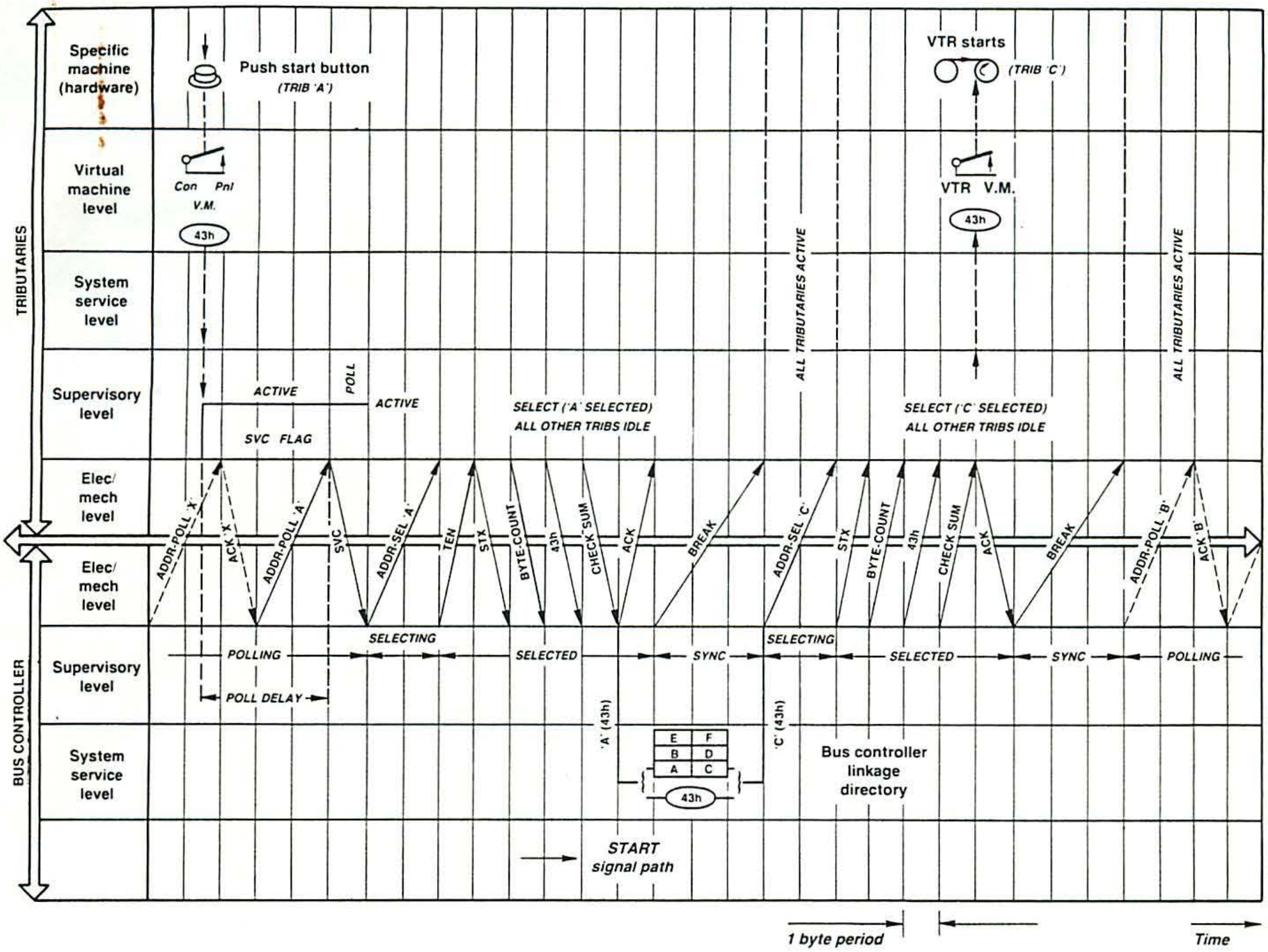


Figure 12
A typical control signal path

broadcasting organizations, each according to its own interpretation of the specifications. Audio and video tape machines, a routing switcher and control panels were included.

The tests successfully demonstrated the validity of the specifications and the ability of different items of equipment to pass control messages and return tallies between controlling and controlled devices. The validation procedure included investigation of the behaviour of the ESbus when operating under the control of bus controllers from different manufacturing sources, and tested the performance in conditions where data errors were deliberately introduced and over the maximum bus length of 1200 m.

A further series of tests conducted in the United States during November 1985 proved the validity of the type-specific message protocol for VTRs that had been approved in the meantime by the EBU and SMPTE, and lent further support to the ESbus as a working system.

It has been gratifying for the specialists who drafted the ESbus specifications to receive encouragement and support from the broadcast equipment manufacturers, and it is believed that the ESbus will establish itself in due course as the standard in this domain. The EBU has issued a general Technical Recommendation (R36-1986) [8] to the effect that broadcasting organizations planning the implementation of remote-control facilities should require the equipment to comply with the ESbus specification, and has reaffirmed its commitment to the system in Technical Recommendation R45-1987 [9] by advocating the inclusion of an ESbus connector on all versions of the D-1 format digital VTR.

It is evidently important that type-specific messages for a wider range of equipment should become available with a minimum of delay; the VTR messages have already been published, those for audio tape machines have been drafted and should be published soon, and work is proceeding with messages for mixing desks and telecines.

## Bibliographical references

[1] **Remote-control systems for broadcast production equipment.**
EBU document Tech. 3245, December 1984.

[2] SMPTE Recommended Practice RP138 (equivalent to Chapter 2 of [1]).

[3] SMPTE Recommended Practice RP113 (equivalent to Chapter 3 of [1]).

[4] SMPTE Recommended Practice RP139 (equivalent to Chapter 4 of [1]).

[5] American National Standard ANSI/SMPTE 207M (equivalent to Chapter 5 of [1]).

[6] **ESbus — System service and common messages.**
Supplement 1 to EBU document Tech. 3245, June 1986.

[7] **Data processing — Open Systems Interconnection — Basic reference model.**
International Standard ISO/DIS 7498, 1982.

[8] **Remote control interface for programme production equipment.**
EBU Recommendation R36-1986.

[9] **Interface functions and connections for the D-1 format 4:2:2 digital television tape recorders.**
EBU Recommendation R45-1987.