# ATIS-0100312.1991(R2006)

# VOICE PACKETIZATION – PACKETIZED VOICE PROTOCOL

AMERICAN NATIONAL STANDARD FOR TELECOMMUNICATIONS

ATIS is the leading technical planning and standards development organization committed to the rapid development of global, market-driven standards for the information, entertainment and communications industry. More than 250 companies actively formulate standards in ATIS' 20 Committees, covering issues including: IPTV, Service Oriented Networks, Home Networking, Energy Efficiency, IP-Based and Wireless Technologies, Quality of Service, Billing and Operational Support. In addition, numerous Incubators, Focus and Exploratory Groups address emerging industry priorities including "Green", IP Downloadable Security, Next Generation Carrier Interconnect, IPv6 and Convergence.

ATIS is the North American Organizational Partner for the 3rd Generation Partnership Project (3GPP), a member and major U.S. contributor to the International Telecommunication Union (ITU) Radio and Telecommunications' Sectors, and a member of the Inter-American Telecommunication Commission (CITEL). For more information, please visit < http://www.atis.org >.

## AMERICAN NATIONAL STANDARD

Approval of an American National Standard requires review by ANSI that the requirements for due process, consensus, and other criteria for approval have been met by the standards developer.

Consensus is established when, in the judgment of the ANSI Board of Standards Review, substantial agreement has been reached by directly and materially affected interests. Substantial agreement means much more than a simple majority, but not necessarily unanimity. Consensus requires that all views and objections be considered, and that a concerted effort be made towards their resolution.

The use of American National Standards is completely voluntary; their existence does not in any respect preclude anyone, whether he has approved the standards or not, from manufacturing, marketing, purchasing, or using products, processes, or procedures not conforming to the standards.

The American National Standards Institute does not develop standards and will in no circumstances give an interpretation of any American National Standard. Moreover, no person shall have the right or authority to issue an interpretation of an American National Standard in the name of the American National Standards Institute. Requests for interpretations should be addressed to the secretariat or sponsor whose name appears on the title page of this standard.

**CAUTION NOTICE:** This American National Standard may be revised or withdrawn at any time. The procedures of the American National Standards Institute require that action be taken periodically to reaffirm, revise, or withdraw this standard. Purchasers of American National Standards may receive current information on all standards by calling or writing the American National Standards Institute.

## Notice of Disclaimer & Limitation of Liability

The information provided in this document is directed solely to professionals who have the appropriate degree of experience to understand and interpret its contents in accordance with generally accepted engineering or other professional standards and applicable regulations. No recommendation as to products or vendors is made or should be implied.

NO REPRESENTATION OR WARRANTY IS MADE THAT THE INFORMATION IS TECHNICALLY ACCURATE OR SUFFICIENT OR CONFORMS TO ANY STATUTE, GOVERNMENTAL RULE OR REGULATION, AND FURTHER, NO REPRESENTATION OR WARRANTY IS MADE OF MERCHANTABILITY OR FITNESS FOR ANY PARTICULAR PURPOSE OR AGAINST INFRINGEMENT OF INTELLECTUAL PROPERTY RIGHTS. ATIS SHALL NOT BE LIABLE, BEYOND THE AMOUNT OF ANY SUM RECEIVED IN PAYMENT BY ATIS FOR THIS DOCUMENT, WITH RESPECT TO ANY CLAIM, AND IN NO EVENT SHALL ATIS BE LIABLE FOR LOST PROFITS OR OTHER INCIDENTAL OR CONSEQUENTIAL DAMAGES. ATIS EXPRESSLY ADVISES ANY AND ALL USE OF OR RELIANCE UPON THIS INFORMATION PROVIDED IN THIS DOCUMENT IS AT THE RISK OF THE USER.

NOTE - The user's attention is called to the possibility that compliance with this standard may require use of an invention covered by patent rights. By publication of this standard, no position is taken with respect to the validity of this claim or any patent rights in connection therewith. The patent holder has, however, filed a statement of willingness to grant license under these rights on reasonable and nondiscriminatory terms and conditions to applicants desiring to obtain such a license. Details may be obtained from the publisher.

ATIS-0100312.1991(R2006), *Voice Packetization – Packetized Voice Protocol*

Formerly known as T1.312-1991(R2006).

Is an American National Standard developed by the **ATIS Network Performance, Reliability, and Quality of Service Committee (PRQC)**.

*Published by*

**Alliance for Telecommunications Industry Solutions**
**1200 G Street, NW, Suite 500**
**Washington, DC 20005**

Printed in the United States of America.

**T1.312-1991 (R2006)**®

American National Standard
for Telecommunications –

# Voice Packetization –
# Packetized Voice Protocol

Secretariat

**Alliance for Telecommunications Industry Solutions**

Approved May 28, 1991

**American National Standards Institute, Inc.**

## Abstract

The purpose of this American National Standard is to standardize the protocol for packetized speech, the Packetized Voice Protocol (PVP). PVP defines formats and procedures for the transport of voice information and channel-associated signaling over a packet network. This is to allow vendors to provide compatible equipment for the U.S. marketplace and to permit both exchange and interexchange carriers to operate compatibly.

## Contents

Page

**Tables**

**Figures**

**Annexes**

## Foreword (This foreword is not part of American National Standard T1.312-1991.)

The purpose of this American National Standard is to specify a protocol for packetized speech, the Packetized Voice Protocol (PVP). PVP defines formats and procedures for the transport of voice information and channel-associated signaling over a packet network. This allows vendors to provide compatible equipment for the US marketplace and both exchange and interexchange carriers to operate compatibly. It is expected that both equipment manufacturers and carriers will utilize this standard.

This standard contains two annexes, both of which are informative and are not part of this standard.

Suggestions for improvement of this standard are welcome. They should be sent to the Exchange Carriers Standards Association, 5430 Grosvenor Lane, Bethesda, MD 20814.

This standard was processed and approved for submittal to ANSI by Accredited Standards Committee on Telecommunications, T1. Committee approval of the standard does not necessarily imply that all members voted for its approval. At the time it approved this standard, the T1 Committee had the following members:

Ivor N. Knight, Chairman
Arthur K. Reilly, Vice-Chairman
O. J. Gussella, Secretary

| *Organization Represented* | *Name of Representative* |
|---|---|
| **EXCHANGE CARRIERS** | |
| Ameritech Services, Inc. | Laurence A. Young |
| | Robert Koren (Alt.) |
| Bell Atlantic Corporation | John W. Seazholtz |
| | Roger Nucho (Alt.) |
| Bellcore | Gary G. Schlanger |
| | Ralph E. Jensen (Alt.) |
| BellSouth Services | Leonard Strickland, Jr. |
| | William J. McNamara, III |
| (Alt.) | |
| Centel Corporation | (Representation vacant) |
| | Bruce Becker (Alt.) |
| Cincinnati Bell Telephone | William P. Keidel |
| | T. C. Grimes (Alt.) |
| Contel Service Corporation | Russell G. DeWitt |
| Exchange Carriers Standards Association | Joseph Mendoza |
| | Gregory L. Theus(Alt.) |
| GTE SC/Telephone Operations | Gregory L. Theus |
| | Richard L. Cochran (Alt.) |
| National Telephone Cooperative Association | Joseph M. Flanigan |
| NYNEX Service Company | Bencilla Jenkins |
| | Leo Katz (Alt.) |
| Pacific Bell | Fred Doell |
| | Tom Garcia (Alt.) |
| Puerto Rico Telephone Company | Segundo Ruiz |
| Southern New England Telephone | H. D. Mulla |
| | J. K. Vij (Alt.) |
| Southwestern Bell Corporation | Joseph Mendoza |
| | C. C. Bailey (Alt.) |
| US Telephone Association (USTA) | Paul K. Hart |
| | Thomas Gajeski (Alt.) |
| United Telecommunications, Inc. | Robert P. McCabe |
| | Harold L. Fuller (Alt.) |
| U S West | James L. Eitel |
| | James Dahl (Alt.) |

| *Organization Represented* | *Name of Representative* |
|---|---|
| **INTEREXCHANGE CARRIERS** | |
| AT&T Communications | Gerald H. Peterson<br>Dennis Thovson (Alt.) |
| CNCP Telecommunications | David H. Whyte<br>John D. Rogers (Alt.) |
| COMSAT Corporation | Carl A. Sederquist<br>Mark T. Neibert (Alt.) |
| Contel ASC | Michael K. Ward<br>Kap Kim (Alt.) |
| International Telecharge, Inc. | Dennis Garaghty<br>Diane Harbaugh (Alt.) |
| MCI Telecommunications Corporation | Stephen J. Engelman<br>Whitey Thayer (Alt.) |
| National Telecommunications Network | John Gurzick<br>Phillip E. Belevre (Alt.) |
| Operator Assistance Network | Steve Pannell<br>Joseph E. Sharkey (Alt.) |
| Telecom Canada | G. T. Clark<br>D. G. Wherry (Alt.) |
| US Sprint | Peter J. May<br>Eric Scace (Alt.) |
| **MANUFACTURERS** | |
| ADC Telecommunications, Inc. | Patrick Cameron<br>Steve Grady (Alt.) |
| AG Communications System | Nigel J. E. Reynolds<br>J. C. Gibson (Alt.) |
| ALCATEL Network Systems Corporation | Kenneth P. Ray<br>Neven Karlovac (Alt.) |
| Amdahl Corporation | Paul Lue<br>Peter Arseniu (Alt.) |
| AMP, Inc. | George Lawrence<br>Jack Bradberry (Alt.) |
| Ant Telecommunications, Inc. | James Seago<br>Neville Ronan (Alt.) |
| Apple Computer, Inc. | Mike Lawler<br>Karen Higginbottom (Alt.) |
| Arix Corporation | Berry Needham<br>Gene Manno (Alt.) |
| AT&T Technologies | Stanley W. Johnston<br>Lleana Fleites-LaSalle (Alt.) |
| Digital Equipment Corporation | Thomas Szczepanski<br>Richard Hovey (Alt.) |
| DSC Communications Corporation | Allen Adams<br>Kishan Shenoi (Alt.) |
| ECI Telecom, Inc. | Douglas E. Peterson<br>Ron Murphy (Alt.) |
| Ericsson, Inc. | David Breeding<br>Guy Campbell (Alt.) |
| Fujitsu America, Inc. | Steven A. Minneman<br>Rodney Boehm (Alt.) |
| General DataComm, Inc. | Frederick Cronin<br>Frederick Lucas (Alt.) |
| Harris Corporation | Allen Jackson<br>Michael E. Weissberger (Alt.) |
| Hekimian Laboratories, Inc. | J. P. Drzewiecki<br>Mike F. Toohig (Alt.) |
| Hewlett-Packard | Don C. Loughry<br>Richard Van Gelder (Alt.) |
| IBM Corporation | Robert M. Amy<br>Ralph Guenther (Alt.) |
| Infotron Systems Corporation | Stig Persson<br>Phillip J. Bird (Alt.) |
| LICOM, Inc. | Joel Martin |
| Lorain Products | Richard L. Haag<br>John Mitchell (Alt.) |

| *Organization Represented* | *Name of Representative* |
|---|---|
| Memorex Telex Corporation | William Greason<br>Douglas O. Kendrick (Alt.) |
| Mitel Corporation | Keith Richardson<br>David Gleave (Alt.) |
| Mitsubishi Electronics America | Philip Jongeneel |
| Motorola, Inc. | G. David Forney, Jr.<br>David Morgan (Alt.) |
| NCR Corporation | Thomas W. Kern<br>Jerome F. Kemp (Alt.) |
| NEC America, Inc. | Art Graham<br>Wayne Lohman (Alt.) |
| Northern Telecom, Inc. | Joseph Samuel<br>Myron Allen (Alt.) |
| Plantronics, Inc. | M. Farrant<br>Brian Cole (Alt.) |
| Racal-Milgo, Inc. | Donald O'Connor<br>Peter Brackett (Alt.) |
| Rockwell International Corporation | Everett Turvey<br>James T. Carter, Jr. (Alt.) |
| Siemens Communication Systems, Inc. | Michael A. Pierce<br>Fred Fink (Alt.) |
| Stromberg-Carlson Corporation | Robert E. Poignant<br>Dennis L. Edinger (Alt.) |
| Tau-Tron, Inc. | Donovan Nak<br>Dave Ludwig (Alt.) |
| Telco Systems, Inc. | Thomas R. Helmes<br>Marshall Hudson (Alt.) |
| Telcom Solutions | M. J. Narasimha<br>Robert Yapp (Alt.) |
| Telecommunications Techniques Corporation | Joseph A. Sciulli<br>Bernard E. Worne (Alt.) |
| TELINQ Systems, Inc. | Steve Tabaska<br>Darrell Heflin (Alt.) |
| Tellabs, Inc. | Charles Rohrs<br>Michael J. Birck (Alt.) |
| Timeplex, Inc. | D. Protopapas<br>L. H. Eberl (Alt.) |
| Verilink Corporation | William J. Buckley<br>Robert Beebe (Alt.) |

**GENERAL INTEREST**

| | |
|---|---|
| American Broadcasting Company | Ken Michel |
| Ashford Associates | Donald A. Ashford |
| Base-2 Systems, Inc. | Douglas M. Brady<br>John Huft (Alt.) |
| Carter Hawley Hale Information Services | James Rothenberger<br>Jose M. Vasquez (Alt.) |
| CBS Broadcasting Group, CBS, Inc. | Joseph McNulty<br>George Knights (Alt.) |
| Creative Communications Consulting | Richard Bobilin<br>James Boe (Alt.) |
| Defense Communications Agency | C. Joseph Pasquariello<br>Granger Kelley (Alt.) |
| Electronic Data Systems Corporation | Douglas Zolnick<br>Mark Terry (Alt.) |
| National Broadcasting Company | Ronald J. Gnidziejko |
| National Communications System | Charles D. Bodson<br>Thomas Drury (Alt.) |
| National Institute of Standards and Technology | Robert Rountree, Jr.<br>Michael D. Hogan (Alt.) |
| National Telecommunications and Information Administration/Institute for Telecommunication Sciences (NTIA/ITS) | William F. Utlaut<br>Neal B. Seitz (Alt.) |
| NTT America, Inc. | Ken-Ichi Aihara<br>Takashi Nishimura (Alt.) |
| OMNICOM, Inc. | Harold C. Folts |
| Rural Electrification Administration | W. Fred Albrecht<br>Gerald S. Schrage (Alt.) |

| *Organization Represented* | *Name of Representative* |
|---|---|
| U. S. General Services Adminstration | Douglas K. Arai<br>Larry L. Jackson (Alt.) |

At the time it approved this standard, the Technical Subcommittee T1Y1 on Specialized Subjects had the following members:

Marshall G. Schachtman, Chairman
Zigmunds Putnins, Secretary

| *Organization Represented* | *Name of Representative* |
|---|---|
| AG Communication Systems | G. Chandran<br>O. Kohout (Alt.) |
| Ameritech Services, Inc. | Richard J. Gillitzer<br>John Berkich (Alt.) |
| AT&T Communications | Anthony Schiano<br>Frank J. Fiederlein (Alt.) |
| AT&T Technologies | Fred C. Rossol<br>Duncan Sparrell (Alt.) |
| Base-2 Systems, Inc. | Douglas M. Brady<br>John Huft (Alt.) |
| Bell Atlantic | Grant Helvey<br>Robert A. Langhage (Alt.) |
| Bellcore | Marshall G. Schachtman<br>Milton Anderson (Alt.) |
| BellSouth Services | H. W. Chen<br>A. G. Drott (Alt.) |
| Compression Labs Inc. | Jonathan Zingman |
| COMSAT Corporation | J. Rieser |
| Contel Service Corporation | Patrick E. Dobbins<br>Larry Day (Alt.) |
| Creative Communications Consulting | Richard T. Bobilin<br>James Boe (Alt.) |
| Defense Communications Agency | Michael Morgan<br>Granger Kelley (Alt.) |
| DSC Communications Corporation | Kishan Shenoi<br>Tom MacDonald (Alt.) |
| ECI Telecom, Inc. | Ron Murphy |
| Ericsson, Inc. | Goran Benjamin<br>Bengt Lagerstedt (Alt.) |
| GTE SC/Telephone Operations | Norman Epstein<br>Ronald E. White (Alt.) |
| Infotron Systems Corporation | Steve Mitropoulos<br>Philip J. Bird (Alt.) |
| Lorain Products | Richard L. Haag<br>John Mitchell (Alt.) |
| Martin-Marietta Information and Communications | (Representation vacant)<br>J. Roberts (Alt.) |
| MCI Telecommunications Corporation | Donald Francis<br>David Reft (Alt.) |
| Mitel Corporation | Jeff Williams<br>Keith Richardson (Alt.) |
| Mitsubishi Electronics America | Philip Jongeneel |
| Motorola, Inc. | Ron Cheung<br>Tim Williams (Alt.) |
| National Communications System | Andre Rausch<br>Gary Redstad (Alt.) |
| National Telecommunications Network | Phillip E. Belevre<br>John Gurzick (Alt.) |
| National Telecommunications and Information Administration/Institute for Telecommunication Sciences (NTIA/ITS) | Robert F. Kubichek<br>Edmund A. Quincy (Alt.) |
| NEC America, Inc. | Wayne Lohman |
| Newbridge Networks Corporation | Scott Marshall<br>Sab Ventola (Alt.) |
| Northern Telecom, Inc. | Joseph Samuel<br>Scott Shaw (Alt.) |

| *Organization Represented* | *Name of Representative* |
|---|---|
| NYNEX Service Company | F. T. Burns<br>A. E. Diblanda (Alt.) |
| Pacific Bell | Tom Garcia<br>Jan Benjamins (Alt.) |
| PictureTel Corporation | Staffan Ericsson<br>Craig Clapp (Alt.) |
| Racal-Milgo, Inc. | Michael S. Clott<br>Ming-Luh Kao (Alt.) |
| Ratelco, Inc. | Alvin G. Graham<br>Mark A. Resler (Alt.) |
| Rockwell International Corporation | Charlie Havens<br>Bob Jones (Alt.) |
| Siemens Communication Systems, Inc. | Lloyd Tarr<br>Michael A. Pierce (Alt.) |
| Southwestern Bell Corporation | Hal Holzwarth<br>Richard E. Weadon (Alt.) |
| Telecom Canada | Paul A. Garrett<br>Rakesh Prasad (Alt.) |
| Telecom Solutions | M. J. Narasimha<br>Richard T. Bobilin (Alt.) |
| Tellabs, Inc. | Jim L. Melsa<br>Jim Mills (Alt.) |
| Timeplex, Inc. | Zigmunds Putnins<br>P. Chen (Alt.) |
| United Telecommunications, Inc. | Calvin M. Theiss<br>Harold L. Fuller (Alt.) |
| U.S. Intelco Networks, Inc. | Nancy D. Clayton |
| US Sprint | Tom Croda<br>Peter J. May (Alt.) |
| US Telephone Association (USTA) | Thomas Gajeski<br>Dennis Byrne (Alt.) |
| U S West | D. L. Kaus<br>William G. Petersen (Alt.) |

Working group T1Y1.2, which developed this standard, had the following officers and participants:

Steven J. Engelman, Chairman
Duncan K. Sparrell, Secretary

Milton M. Anderson
D.J. Atkinson
Yair Be’ery
Udaya Bhaskat
Dick Bobilin
Douglas M. Brady
Hugh W. Chen
Juin-Hwey Chen
Harprit S. Chhatwal
Sein Chow
Michael Clott
Billie Cool
Richard V. Cox
Antony Crossman
Vladimir Cuperman
Spiros Dimolitsas
Franklin D. Fite, Jr.
John Gangluff
Tom Garcia
Paul Garrett
Allen Gersho

Dick Gillitzer
P.K. Guria
Hal Holzwarth
Jas Jain
Wu Ming-Juag
Ming-Luh Kao
David Kaus
Bob Kepley
Robert J. Koren
Bob Kubichek
Wayne E. Lohman
Chi-Smi Lzu
Tom MacDonald
Robert P. McCabe
Steve Mitropoolos
Michael Morgan
James J. Mossman
Ron Murphy
Ming D. Ni
Peter P. Nuspi
Bruce Orford

Orni Petrushka
Zigmunds Putnins
Ed Quincy
Jack H. Rieser
G. Ferrell Sandy
Marshall Schachtman
Mostafa Hashem Sherif
Richard J. Skulins
Doyle Sturms
Allen Swig
Lloyd Tarr
Robert P. Testard
Calvin M. Theiss
Reinaldo Valenzuela
Vijay K. Varma
Sab Ventola
Jeff Williams
Tim A. Williams
Phil Wilson
Cliff Woodbury
Teresa A. Zoller

American National Standard
for Telecommunications –

# Voice Packetization –
# Packetized Voice Protocol

## 1 Scope, purpose, and application

### 1.1 Scope

This standard defines a Packet Voice Protocol (PVP) for speech packetization in permanent virtual circuit applications. The packetized voice protocol can be used, for example, at bit rates above 64 kbit/s and less than 150 Mbit/s and in fractional DS1 applications.

The protocol defines formats and procedures for the transport of voice information and channel-associated signaling over a wideband packet network.

The standard accommodates additional future types, including optional internetworking capabilities with the digital cellular radio networking application under development in the TIA standards forums (TR45.3).

The extension of this standard for baseband facsimile traffic is currently under study.

This standard does not address performance issues.

This standard does not address methods for coding speech samples, although particular coding standards are specified in the protocol (e.g., the algorithms in ANSI T1.303-1989). In particular, the standard allows dynamic bandwidth allocation and graceful congestion control when the speech samples are coded with embedded algorithms such as those specified in ANSI T1.310-1991.

This standard does not address the following issues:

– services based on the interface;

– implementation techniques;

– performance guidelines relative to the use of packetized speech;

– equipment aspects;

– trunk-signaling, link-establishment, and call-establishment procedures for switched virtual circuits;

– data-only issues, combined data/voice issues, and frame relay issues;

– speech packetization in Asynchronous Transfer Mode (ATM) (B-ISDN) systems.

NOTE – The user’s attention is called to the possibility that compliance with this standard may require use of an invention covered by patent rights.

By publication of this standard, no position is taken with respect to the validity of this claim or of any patent rights in conjunction therewith. The patent holder has, however, filed a statement of willingness to grant a license under these rights on reasonable and nondiscriminatory terms and conditions to applicants desiring to obtain such a license. Details may be obtained from the publisher.

### 1.2 Purpose and application

This standard can be applied for packetization of speech and in fractional DS1 applications.

## 2 Normative references and definitions

### 2.1 Normative references

The following standards contain provisions which, through reference in this text, constitute provisions of this American National Standard.

At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this American National Standard are encouraged to investigate the possibility of applying the most recent editions of the standards indicated below.

ANSI T1.303-1989, *Telecommunications – Digital processing of voice-band signals – Algorithms for 24-, 32-, and 40-kbit/s adaptive differential pulse-code modulation (ADPCM)* [1)]

ANSI T1.306-1990, *Telecommunications – Digital processing of audio signals – Algorithm and line format for transmission of 7-kHz audio signals at 64/56 kbit/s* [1)]

ANSI T1.310-1991, *Telecommunciations – Digital processing of voice-band signals – Algorithms for 5-, 4-, 3-, and 2-bits/sample embedded adaptive differential pulse-code modulation (ADPCM)* [1)]

CCITT Recommendation G.703, *Physical/electrical characteristics of hierarchical digital interfaces* [1), 2)]

CCITT Recommendation G.704, *Synchronous frame structures used at primary and secondary hierarchical levels* [1), 2)]

CCITT Recommendation G.711, *Pulse-Code Modulation (PCM) of voice frequencies* [1), 2)]

CCITT Recommendation G.722, *7-kHz audio-coding within 64 kbit/s* [1), 2)]

CCITT Recommendation G.726, *40-, 32-, 24-, 16-kbit/s adaptive differential pulse-code modulation (ADPCM)* [1), 2)]

CCITT Recommendation G.727, *5-, 4-, 3-, and 2-bit/sample embedded adaptive differential pulse-code modulation* [1), 2)]

CCITT Recommendation H.221, *Frame structure for a 64 to 1920 kbit/s channel in audiovisual teleservices* [1), 2)]

CCITT Recommendation I.431, *Primary rate user-network interface – Layer 1 specification* [1), 2)]

CCITT Recommendation Q.921/I.441, *ISDN user-network interface – Data link layer specification* [1), 2)]

ISO 3309:1984, *Information processing systems – Data communications – High-level data link control procedures – Frame structure* [1)]

### 2.2 Definitions

This standard includes the following definitions:

**2.2.1 Adaptive Differential Pulse-Code Modulation (ADPCM):** ADPCM algorithms are predictive, variable quantization coding schemes that code the difference between successive speech samples to reduce the bit rate and increase transmission capacity.

**2.2.2 ALARM state:** The state of a PVP node that has detected a facility alarm, has stopped receiving signaling packets, or has received a signaling packet with the N/A bit set to 1.

**2.2.3 block-dropping indicator (BDI):** Used by a PVP node to track the status of block-dropping within a voice packet (i.e, the number of blocks that have been dropped and the maximum number that can be dropped).

**2.2.4 blocks:** A specific group of octets within a voice frame packet that is derived from the speech samples from the channelized side of a permanent voice circuit defined at a PVP node. A block is made up of bits of the same significance collected from the coded speech samples that are to be packetized. It contains 128 bits assembled in order of receipt of the 128 samples that make up the voice format packet. The number of bits in a sample is dependent on the CT used.

**2.2.5 build-out delay:** The maximum variable transmission and processing delay that is permitted in a wideband network. Each PVP node records the delay that it adds and updates accordingly the TS value in the packet header. When the terminating endpoint of a PVP node receives a voice packet, the provisioned build-out and the time stamp value are compared. If the TS value is larger than the build-out, the PVP node drops the packet. If the delay is less than the provisioned build-out, the packet is delayed by the difference between the two values.

[1)] Available from American National Standards Institute, Inc., 11 West 42nd Street, New York, NY 10038.

[2)] All CCITT Recommendations referenced in this standard were published in the 1988 Blue Book.

**2.2.6 bursts:** Periods of speech activity or high-energy content signals present in the access channel of a wideband network.

**2.2.7 check sequence (CS):** A 16-bit sequence in the last two octets of a frame (excluding flags) that offers a cyclic redundancy check (CRC). The CRC is derived over either the header in UIH format frames or over the entire packet frame for UI frames (excluding flags). The algorithm used is described in ISO 3309.

**2.2.8 coding type (CT) field:** The CT field of a voice packet indicates the method of coding speech samples used at the originating endpoint before packetization.

**2.2.9 command/response (C/R) bit:** Used in LAPD (CCITT Recommendation Q.921/I.441) to distinguish between packets that initiate communication (C/R = 0) and packets that are responses to initiated communication (C/R = 1). In PVP, the C/R bit is always set to 0.

**2.2.10 congestion level indicator (CLI):** A local variable of a PVP node that indicates the number of blocks to be dropped from packets containing droppable blocks (0, 1, 2, or 3 blocks may be specified).

**2.2.11 data link connection identifier (DLCI):** A 13-bit field that defines the destination address of a packet. The address is local on a link-by-link basis.

**2.2.12 embedded Adaptive Differential Pulse-Code Modulation (ADPCM):** Similar to non-embedded ADPCM with the following advantage. In a congested voice network, they allow an intermediate node to reduce the bit allocation for particular incoming channels without having to exchange control messages between the various nodes in the path of the connection.

**2.2.13 extension bit:** The first bit of each octet of the address field of a packet. The first octet is identified by the extension bit set to 0, and the second octet is identified by the extension bit set to 1.

**2.2.14 frame abort:** Occurs when a link layer entity ignores a frame that is currently receiving. It is initiated by receipt of a frame with an invalid flag, or receipt of seven or more contiguous 1 bits.

**2.2.15 frame check sequence (FCS):** A 16-bit CRC CS that is derived over an entire packet frame (excluding flags) of a UI format packet. The CRC algorithm is described in ISO 3309.

**2.2.16 gaps:** Periods of no speech activity or low-energy content signals present in the access channel of a wideband packet network.

**2.2.17 header check sequence (HCS):** A 16- bit CRC CS that is derived from the first 8 octets (excluding flags) of a UIH format packet. The CRC algorithm is described in ISO 3309.

**2.2.18 keep-alive alarm:** An alarm generated by a terminating endpoint that experiences loss of keep-alive packets as determined by the expiration of the TSIG_KA timer. This indicates that a failure has occurred within the packet portion of the connection and that normal signaling transport has been interrupted.

**2.2.19 more (M) bit:** Set to 1 for all packets except for the last packet of a burst, which is set to 0. It is used to indicate that more packets in sequence are to be expected by the terminating endpoint. This allows the terminating endpoint to recover from packet loss.

**2.2.20 NORM state:** Exists on a permanent virtual circuit connection when there are no alarm conditions on either the originating or terminating endpoints. The absence of alarms must be satisfied on both the channelized side and the packetized side. During the NORM state, signaling packets have their N/A bit set to 0.

**2.2.21 noise field:** Indicates the level of background noise that the terminating endpoint may play out in the absence of packets.

**2.2.22 normal/alarm (N/A) state bit:** Used in signaling packets to transfer information on alarm status across a permanent virtual circuit in the direction of transmission from the channelized side to the packetized side. The N/A bit set to 0 indicates normal operation. The N/A bit set to 1 indicates the existence of an alarm on the channelized access facility or error condition on the permanent virtual circuit.

**2.2.23 originating endpoint:** The originating endpoint of a PVP node receives channelized traffic at 64 kbit/s and transmits packets into the wideband packet network.

**2.2.24 packet header:** The first 8 octets of the packet. The header contains the address field, the control field (UI or UIH), the PD, and TS.

**2.2.25 packetization interval:** Defines the duration of the sampled speech of the access channel that has been collected, coded, and packetized. The packetization interval for voice is 16 ms.

**2.2.26 permanent virtual circuit applications:** Transmission applications in which nodes receive channelized traffic at the rate of 64 kbit/s and transform the information into packet format for transmission into a wideband packet network. The node must also receive packetized traffic and retransform it into channelized traffic at the rate of 64 kbit/s.

**2.2.27 poll (P) bit:** Bit 5 of a UI or UIH frame control field.

**2.2.28 protocol discriminator (PD) field:** The first octet of the packet header. The PD is used to distinguish the protocol used in the packet frame.

**2.2.29 refresh packets:** Signaling packets that are transmitted on a periodic basis across a permanent virtual circuit in the direction of transmission from the channelized side to the packetized side. This allows the originating endpoint to update the knowledge of the terminating endpoint regarding the signaling information on the incoming channel and the status of the permanent virtual circuit status.

**2.2.30 remote alarm:** In a PVP node, an indication that the far-end PVP node is experiencing an alarm condition. This occurs because of either the loss of keep-alive signaling packets, or the receipt of signaling packets with the N/A bit set to 1.

**2.2.31 scheduled play-out time:** The time at which a received packet is to be played out. All packets are to be played out at regular intervals despite the variability in their arrival times. Since the provisioned build-out value is the maximum variable transmission delay, packets arriving early must be held by the terminating endpoint until their scheduled play-out time. Late packets are dropped.

**2.2.32 sequence number (SEQ):** The SEQ of a voice packet is used by the endpoints in the build-out process to: (a) determine the first packet of a burst and (b) determine if a packet has been lost. The SEQ, in conjunction with the TS, allows the removal of variability in network delay.

**2.2.33 signaling transition:** A signaling transition is defined for channel-associated signaling as a change in state of the A bit for 2-state signaling, A and/or B bit in 4-state signaling, or the A, B, C, and/or D bit for 16-state signaling.

**2.2.34 terminating endpoint:** The terminating endpoint of a PVP node receives packetized traffic and retransforms it into channelized traffic at the rate of 64 kbit/s.

**2.2.35 time stamp (TS):** Records the cumulative variable queuing delays experienced by a packet in transversing the network with a resolution of 1 ms. To prevent wrap-around, the maximum valid value in the TS field shall not exceed 199 ms. If, after update, the variable delay exceeds 199 ms, the value is set to 199 ms.

**2.2.36 unnumbered information (UI) frames:** Used to transfer unacknowledged information between two link layer entities, i.e., without affecting data link layer variables. UI command frames do not carry a sequence number and, therefore, the UI frame may be lost without notification.

The control field for the UI command frame is a single octet in length. The format and encoding are the same as specified in CCITT Recommendation Q.921/I.441. The CRC is derived over the entire frame and is inserted in the CS field.

**2.2.37 unnumbered information with header check (UIH) frame:** The UIH frame is similar to the UI frame except that the CRC sequence is derived over the frame and packet headers (the first 8 octets excluding flags) rather than over the entire frame. The CS fills the last two octets of the UIH frame.

The control field of the UIH is a single octet in length. The format and encoding are the same as specified in CCITT Recommendation Q.921/I.441.

**2.2.38 voice frame:** A UIH frame that contains a voice packet in its information field.

**2.2.39 voice information field:** The voice information field of a voice packet contains blocks arranged according to the significance of the bits. The first block contains the most significant bits (MSBs) of all samples, the second contains the second MSBs, and so on. Within a block, bits are ordered according to their corresponding sample numbers.

## 3 Overview

This standard contains the specification of a protocol for packetized speech. The packetized voice protocol defines formats and procedures for the transport of voice information and channel-associated signaling over a packet network.

Before packetization, the input speech samples may be coded at the originating endpoint of the transmitting side by one of the coding methods indicated in this document. The stream of coded speech is transformed into packets with the format specified in this document. The samples are collected over a period of 16 ms and divided into blocks of 128 bits each. Silent intervals may be removed. The blocks are arranged to facilitate block-dropping.

Periods of activity and inactivity are respectively called "bursts" and "gaps." It is not necessary to transmit packets during gaps.

The terminating end at the receiving side reconstructs a continuous stream of speech from the incoming packets using the information in the packet header. The build-out delay procedure described in this document compensates for the variable delay that packets may experience within the network. Packets that arrive before their scheduled play-out time are placed in the proper sequence in a packet queue. Packets that arrive after their scheduled play-out time are discarded. The voice packet header contains information about the level of noise that was measured by the originating endpoint. The terminating endpoint uses this information to play out a matching noise level.

An additional feature of PVP is the ability to drop blocks from a packet as a congestion-control mechanism. The *n* th block consists of the *n*th bit from each sample collected during the sampling interval. The packet header indicates the number of droppable blocks contained in the packet. Congested nodes may use this information to drop the least significant block from packets to abate the congested state.

The signaling associated with each voice connection shall be transported in signaling packets. Signaling packets shall be sent separately on a different logical channel. Transport of the signaling information requires a set of procedures, similar to those for voice transport, which are described in this document.

## 4 Formats

### 4.1 Physical layer

For operations at 1536 kbit/s or 1920 kbit/s, the electrical characteristics and formats of the interface are those defined in CCITT Recommendation I.431 (G.703 and G.704) for the primary rates of 1544 kbit/s and 2048 kbit/s, respectively. The packetized signal consists of one digital stream sent over conventional primary rate facilities. Hybrid situations containing one or more $N \times 64$ kbit/s packet streams and $M$ conventional 64 kbit/s channels are also considered.

#### 4.1.1 Bit inversion

For primary rate applications requiring code restrictions that maintain ONE's density, bit inversion is necessary to enable the combined result of bit stuffing and bit inversion to prevent the all 0 octet and to satisfy the ONE's density requirements of restricted DS1 facilities.

#### 4.1.2 Order of transmission

Bit 1 is the least significant bit (LSB) and is transmitted first. Bit 8 is the most significant bit (MSB) and is transmitted last.

### 4.2 Link layer

The link layer of PVP uses a similar approach to CCITT Recommendation Q.921/I.441 with the additions indicated in this standard. In particular, PVP uses the unnumbered information frame with header check (UIH), described in 4.2.3.2, that is not currently specified in CCITT Recommendation Q.921/I.441. Frames that transport voice and frames that transport channel-associated signaling are assigned different layer-2 addresses, i.e., they are carried on two separate logical links. This, together with the use of a different unnumbered frame type for each type of traffic, provides an additional measure of security to protect from the misrouting of signaling information.

#### 4.2.1 Address field

The address field is two octets in length, with the first bit of each defined as an extension bit and bit 2 of octet 1 defined as the command/response (C/R) bit. The 13 bits that remain are concatenated to form a single data link connection identifier (DLCI). Address assignment starts with 128 and ends with 8063. Layer-2 addresses are already assigned, and the implementation starts from the DLCI_ ASSIGNED state.

#### 4.2.2 Command/response bit

The C/R bit (bit 2 of octet 1) is set to 0.

#### 4.2.3 Frame types

The following two frame types are allowed in PVP.

##### 4.2.3.1 Unnumbered information frames

When a layer-3 or management entity requests unacknowledged information transfer, the unnumbered information (UI) command is used to send information to its peer without affecting data link layer variables. UI command frames do not carry a sequence number and, therefore, the UI frame may be lost without notification.

The control field for the UI command frame is a single octet in length. The format and encoding are the same as specified in CCITT Recommendation Q.921/I.441.

The UI frame is used to transport channel-associated signaling.

##### 4.2.3.2 Unnumbered information with header check frame

The unnumbered information with header check (UIH) frame has the same applications as the UI frame. The difference between the two is that the cyclic redundancy check (CRC) sequence is derived over the frame and packet headers (the first 8 octets, excluding flags) rather than over the entire frame. The check sequence fills the last two octets of the UIH frame.

The control field of the UIH is a single octet in length and is shown in figure 1.

The UIH frame is used to transport voice. [3)]

#### 4.2.4 Poll bit

The poll (P) bit is bit 5 of the UI/UIH frame control field. The P bit shall be set to 0.

#### 4.2.5 Check sequence

The check sequence (CS) algorithm is the same as that described in ISO 3309. The CS field shall be a 16-bit sequence. It shall be the ONEs complement of the sum (modulo 2) of:

– The remainder of ($x$ raised to $k$ power) ($x^{15} + x^{14} + x^{13} + x^{12} + x^{11} + x^{10} + x^9 + x^8 + x^7 + x^6 + x^5 + x^4 + x^3 + x^2 + x^1 + 1$) divided (modulo 2) by the generator polynomial $x^{16} + x^{12} + x^5 + 1$, where $k$ is the number of bits in the frame existing between, but not including, the final bit of the opening flag and the first bit of the first octet of the nondroppable octets for the header check sequence or the first bit of the check sequence for the frame check sequence, excluding bits inserted for transparency:

– The remainder of the division (modulo 2) by the generator polynomial $x^{16} + x^{12} + x^5 + 1$ of the product of $x^{16}$ by the content of the frame existing between, but not including, the final bit of the opening flag and the first bit of the first octet of the nondroppable octets for the HCS or the first bit of the CS for the FCS, excluding bits inserted for transparency.

As a typical implementation at the transmitter, the initial content of the register of the device computing the remainder of the division is preset to all ONEs and is then modified by division by the generator polynomial (as described above) of the address, control, and appropriate portion of the information fields; the ONEs complement of the resulting remainder is transmitted as the 16-bit CS.

As a typical implementation at the receiver, the initial content of the register of the device

[3)] The CRC check of the UIH frame protects 8 octets, which contain the address field (to ensure correct delivery), the control field (to guarantee the validity of the frame type), and the layer-3 header. It does not protect the voice information because voice traffic is more sensitive to delay due to retransmission than to bit errors, and because this allows reduction of voice information by block-dropping under congestion without recalculating the CRC check. As a consequence, the test for invalid frames in 4.2.7 uses the minimum frame length of 10 octets.

| 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit number |
|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | P | 1 | 1 | 1 | 1 | |

**Figure 1 – Control field of the UIH frame**

computing the remainder is preset to all ONEs. The final remainder after multiplication by $x^{16}$ and then division (modulo 2) by the generator polynomial $x^{16} + x^{12} + x^5 + 1$ of the serial incoming protected bits and the CS, will be 0001110100001111 ($x^{15}$ through $x^0$, respectively) in the absence of transmission errors.

### 4.2.6 Frame abort

Receipt of seven or more contiguous 1 bits shall be interpreted as a frame abort, and the link layer entity shall ignore the frame currently being received. A frame following an abort must begin with an opening flag.

### 4.2.7 Invalid UI/UIH frames for PVP

For the purposes of PVP, an invalid UI/UIH frame is a frame that:

- is not properly bounded by two flags; or
- has fewer than 10 octets between flags; or
- has greater than 490 octets between flags; or
- does not consist of an integral number of octets prior to ZERO bit insertion or following ZERO bit extractions; or
- contains an FCS error.

Invalid frames shall be discarded without notification to the sender. No action is taken as the result of that frame.

### 4.3 Packet layer

The packet layer procedures apply to the information transfer phase only. Call-control procedures are outside the scope of this standard.

### 4.3.1 Voice packet format

The format of voice packets is shown in figure 2 within the UIH voice frame.

NOTE – The reserved bits are set to 0 at the originating endpoint. They shall be ignored at the terminating endpoint. They shall not be used for testing or maintenance purposes in anticipation of possible future uses.

### 4.3.1.1 Protocol discriminator

The protocol discriminator field is the first octet of the packet header (octet 4 of the frame in figure 2). Its value for the PVP is given in figure 3.

### 4.3.1.2 Block-dropping indicator

The block-dropping indicator tracks the status of block-dropping within the voice packet. A block consists of bits of the same significance collected from all speech samples that are packetized. The size of the block is 128 bits, which, for a sampling rate of 8 kHz, corresponds to a packetization interval of 16 ms. Blocks are arranged in decreasing order of significance. (See figure 7.)

The format of the BDI is shown in figure 4.

The combination C1 and C2 form the C-subfield that indicates the number of blocks that are still droppable at any intermediate node in the network, as shown in table 1.

As blocks are dropped from the packet, the value in the C-subfield is decremented to reflect the number of blocks still available for dropping.

The combination M1 and M2 forms the M-subfield that indicates the total number of blocks that can be discarded from the packet as it traverses the network during periods of network congestion, as shown in table 2. The value in the M-subfield is not changed from its initial value.

For fixed-rate coding, both the M-subfield and the C-subfield are set to 0.

### 4.3.1.3 Time stamp

The time stamp records the cumulative variable queuing delays experienced by a packet in traversing the network with a resolution of 1 ms. To prevent wrap-around, the maximum valid value in the TS field shall not exceed 200 ms. If, after update, the variable delay exceeds 200 ms, the value is set to 200 ms.

### 4.3.1.4 Coding type

The coding type field indicates the method of coding the speech samples at the originating endpoint before packetization. Table 3 shows the valid encodings for the field.

<table>
<tr><th>8</th><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th></th></tr>
<tr><td colspan="6">Address (upper subfield)</td><td>0</td><td>0</td><td>Octet 1</td></tr>
<tr><td colspan="7">Address (lower subfield)</td><td>1</td><td>Octet 2</td></tr>
<tr><td colspan="8">UIH control field</td><td></td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>P</td><td>1</td><td>1</td><td>1</td><td>1</td><td>Octet 3</td></tr>
<tr><td colspan="8">Protocol discriminator</td><td></td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>Octet 4</td></tr>
<tr><td colspan="8">Block-dropping indicator</td><td>Octet 5</td></tr>
<tr><td colspan="8">Time stamp</td><td>Octet 6</td></tr>
<tr><td>M</td><td>R</td><td>R</td><td colspan="5">Coding type</td><td>Octet 7</td></tr>
<tr><td colspan="4">Sequence number</td><td colspan="4">Noise</td><td>Octet 8</td></tr>
<tr><td colspan="8">Nondroppable blocks</td><td></td></tr>
<tr><td colspan="8">Optionally droppable blocks</td><td></td></tr>
<tr><td colspan="8">Check sequence 2 octets</td><td></td></tr>
</table>

M = More bit
P = Poll bit = 0 (See 4.2.4)
R = Reserved for future use

**Figure 2 – UIH voice frame format**

| 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit number |
|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | |

**Figure 3 – Protocol discriminator for the PVP**

<table>
<tr><th colspan="8">Bit number</th></tr>
<tr><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td></tr>
<tr><td>R</td><td>R</td><td>M2</td><td>M1</td><td>R</td><td>R</td><td>C2</td><td>C1</td></tr>
</table>

R = Reserved for future use

**Figure 4 – Block-dropping indicator format**

**Table 1 – C-subfield codings**

| C2 | C1 | Number of droppable blocks |
|---|---|---|
| 0 | 0 | 0 Blocks |
| 0 | 1 | 1 Block |
| 1 | 0 | 2 Blocks |
| 1 | 1 | 3 Blocks |

**Table 2 – M-subfield codings**

| M2 | M1 | Number of droppable blocks |
|---|---|---|
| 0 | 0 | 0 Blocks |
| 0 | 1 | 1 Block |
| 1 | 0 | 2 Blocks |
| 1 | 1 | 3 Blocks |

When the coding type for voiceband signals is Fixed or Embedded Adaptive Differential Pulse Code Modulation (ADPCM), the polarity of ADPCM samples shall comply with CCITT Recommendation G.726. When the coding type for voiceband signals is 8-bit Pulse-Code Modulation (PCM), the polarity of PCM samples shall comply with CCITT Recommendation G.711.

#### 4.3.1.5 M-bit

The M-bit is set to 1 for all packets except for the last packet of a burst where it is set to 0. The M-bit may be used by the terminating endpoint to recover from packet loss.

#### 4.3.1.6 Sequence number

The sequence number (SEQ) is used by endpoints in the build-out process to determine the first packet of a burst and to determine if a packet has been lost. The SEQ, in conjunction with the TS, allows the removal of variability in network delay.

SEQ 0 is placed in the first packet of a voice burst. Subsequent packets in the same burst are given numbers 1 to 15, rolling back to 1.

#### 4.3.1.7 Noise field

The noise field indicates a background noise level as shown in table 4. The receiving end uses this field to determine the level of the background noise that may be played in the absence of packets.

#### 4.3.1.8 Voice information field

The voice information field contains blocks arranged according to the significance of the bits. The first block contains the MSBs of all samples, the second contains the second MSBs, and so on. Within a block, bits are ordered according to their corresponding sample numbers.

Figure 5 shows the bit nomenclature convention before packetization and the information field format after packetization. Figure 6 depicts the format of the entire voice packet where the voice is coded using an embedded (5,2) ADPCM algorithm. Here, up to three blocks can be dropped. Note the most significant bit for PCM is the sign bit.

### 4.3.2 Signaling packet format

The format of the channel-associated signaling packets is shown in figure 7 within a UI signaling frame format. See 4.3.1 regarding the setting of the reserved bits.

#### 4.3.2.1 Protocol discriminator

The format and encoding of the PD field are the same as in the voice packet format (see 4.3.1.1).

#### 4.3.2.2 Block-dropping indicator

The format of the BDI field is the same as in the voice packet format (see 4.3.1.2). Both the C-subfield and the M-subfield are set to 0.

#### 4.3.2.3 Time stamp

The format of the TS field is the same as in the voice packet format (see 4.3.1.3).

#### 4.3.2.4 Normal/alarm state bit

The normal/alarm (N/A) bit is used to transfer information on alarm status across a virtual circuit in the direction of transmission from the full rate access side to the packet side. The N/A bit set to 0 indicates normal operation. The N/A bit set to 1 indicates the existence of an alarm on the full rate access facility or error condition on the virtual circuit.

#### 4.3.2.5 M-bit

The M-bit shall be set to 0 in all signaling packets.

Table 3 – Coding type format

| Bit number | | | | | Coding type [1] |
|---|---|---|---|---|---|
| 5 | 4 | 3 | 2 | 1 | |
| 0 | 0 | 0 | 0 | 0 | 8 bits/sample [2] |
| 0 | 0 | 0 | 0 | 1 | 1 bit/sample [2] |
| 0 | 0 | 0 | 1 | 0 | 2 bits/sample [2] |
| 0 | 0 | 0 | 1 | 1 | 3 bits/sample [2] |
| 0 | 0 | 1 | 0 | 0 | 4 bits/sample [2] |
| 0 | 0 | 1 | 0 | 1 | 5 bits/sample [2] |
| 0 | 0 | 1 | 1 | 0 | 6 bits/sample [2] |
| 0 | 0 | 1 | 1 | 1 | 7 bits/sample [2] |
| 0 | 1 | 0 | 0 | 0 | 8 bit A-law PCM [3] |
| 0 | 1 | 0 | 0 | 1 | 8 bit μ-law PCM [3] |
| 0 | 1 | 0 | 1 | 0 | 2 bits/sample ADPCM [4] |
| 0 | 1 | 0 | 1 | 1 | 3 bits/sample ADPCM [4] |
| 0 | 1 | 1 | 0 | 0 | 4 bits/sample ADPCM [4] |
| 0 | 1 | 1 | 0 | 1 | 5 bits/sample ADPCM [4] |
| 0 | 1 | 1 | 1 | 0 | Reserved for future use |
| 0 | 1 | 1 | 1 | 1 | Reserved for future use |
| 1 | 0 | 0 | 0 | 0 | Reserved for future use |
| 1 | 0 | 0 | 0 | 1 | Reserved for future use |
| 1 | 0 | 0 | 1 | 0 | Reserved for future use |
| 1 | 0 | 0 | 1 | 1 | Reserved for future use |
| 1 | 0 | 1 | 0 | 0 | (4,2) embedded ADPCM [5] |
| 1 | 0 | 1 | 0 | 1 | (5,2) embedded ADPCM [5] |
| 1 | 0 | 1 | 1 | 0 | Reserved for future use |
| 1 | 0 | 1 | 1 | 1 | Reserved for future use |
| 1 | 1 | 0 | 0 | 0 | (8,6) embedded ADPCM [6] |
| 1 | 1 | 0 | 0 | 1 | Reserved for future use |
| | | · · · | | | |
| | | · · · | | | |
| 1 | 1 | 1 | 1 | 1 | Reserved for future use |

[1] Reserved coding types will be assigned as other voice-coding algorithms are standardized. In particular, coding types will be assigned for 8-kbit/s and 4-kbit/s coding algorithms intended for digital cellular radio applications. The specific voice packet information field format and length, and the voice transport procedures, will depend on the coding algorithm. In particular, these matters will depend on whether the coding algorithm has a natural block length, has variable rates, or has droppable bits.

[2] This is to carry 8×N kbit/s on transparent channels.

[3] Interoperability issues are items for further study.

[4] ANSI T1.303/CCITT Recommendation G.726.

[5] Applications of the (5,4), (5,3), (4,3), (3,3), and (3,2) ADPCM algorithms are left for further study.

[6] ANSI T1.306/CCITT Recommendation G.722. The procedure for going from an H.221 frame structure to ANSI T1.312 is an item for further study.

Table 4 – Noise field format

| Bit number<br>4321 | Noise level<br>(dBrnc0) |
|---|---|
| 0000 | Idle Code |
| 0001 | 16.6 |
| 0010 | 19.7 |
| 0011 | 22.6 |
| 0100 | 24.9 |
| 0101 | 26.9 |
| 0110 | 29.0 |
| 0111 | 31.0 |
| 1000 | 32.8 |
| 1001 | 34.6 |
| 1010 | 36.2 |
| 1011 | 37.9 |
| 1100 | 39.7 |
| 1101 | 41.6 |
| 1110 | 43.8 |
| 1111 | 46.6 |

#### 4.3.2.6 Sequence number

The SEQ for signaling packets is always set to 0.

#### 4.3.2.7 ABCD signaling bits

The originating endpoint uses the ABCD signaling bits to indicate to the terminating endpoint on the far side the current signaling state of the full rate access channel in the direction of transmission. The value of the A-bit alone is meaningful in two-state signaling systems. The values of the A and B bits alone are meaningful in four-state signaling systems. The values of all ABCD bits are meaningful in 16-state signaling systems. The values of the ABCD bits have no significance when there is no channel-associated signaling.

The number of signaling states must be the same for both the originating and terminating endpoints on a virtual circuit basis. The values coded in this field depend on the framing type and the number of signaling states.

## 5 Link layer procedures

### 5.1 Addressing

Voice and signaling transport packets are transmitted on different layer-2 addresses.

### 5.2 Endpoint procedures

These procedures apply for UI and UIH frames.

#### 5.2.1 Transmitting UI frames

Information received by the link layer entity from layer 3 by means of a DL_UNIT_DATA_REQUEST shall be transmitted as unnumbered information with an FCS. The P bit shall be set to 0. The list of all primitives used in the procedures is given in clause 10.

#### 5.2.2 Transmitting UIH frames

Information received by the link layer entity from layer 3 by means of a DL_UNIT_H_DATA_REQUEST shall be transmitted as unnumbered information with an HCS. The P bit shall be set to 0.

#### 5.2.3 Receiving UI frames

When a link layer entity is not in a receiver busy condition and receives a valid UI frame, the link layer entity shall pass the information field of this frame to layer 3 using the primitive DL_UNIT_DATA_INDICATION.

#### 5.2.4 Receiving UIH frames

When a link layer entity is not in a receiver busy condition and receives a valid UIH frame, the link layer entity shall pass the information field of this frame to layer 3 using the primitive DL_UNIT_H_DATA_INDICATION.

### 5.3 Intermediate node procedures

#### 5.3.1 Transmitting a frame

Whenever a frame is received from the link layer entity receive procedure, the frame shall be transmitted with the same frame type, including the P bit value, and the C/R bit value as in the received frame.

#### 5.3.2 Receiving a frame

Detected invalid frames (e.g., failed FCS or HCS, unassigned DLCI) shall be discarded with no indication passed to layer 3. The control field of the frame shall be examined. Upon recognition of UI or UIH frame types, the frame is passed to layer 3 with the DL_PVP_DATA_INDICATION primitive for UI frames and DL_PVP_ H_DATA_INDICATION primitive for UIH frames.

The Address and CS are the only fields modified during the link layer procedure.

| Sample number | | | | |
|---|---|---|---|---|
| S1 | MSB/S1 | (MSB1 – 1)/S1 | ... | LSB/S1 |
| S2 | MSB/S2 | (MSB – 1)/S2 | ... | LSB/S2 |
| . | | | . | |
| . | | | . | |
| . | | | . | |
| S128 | MSB/S128 | (MSB – 1)/S128 | ... | LSB/S128 |

***(a) Pre-packetization bit format***

| Bit number | 8 | 7 | | 1 |
|---|---|---|---|---|
| MSB | MSB/S8 | MSB/S7 | ... | MSB/S1 |
| Block | | | ⋮ | |
| | MSB/S128 | MSB/S127 | ... | MSB/S121 |
| MSB – 1 | (MSB-1)/S8 | (MSB-1)/S7 | ... | (MSB-1)/S1 |
| Block | | | ⋮ | |
| | (MSB-1)/S128 | (MSB-1)/S127 | ... | (MSB-1)/S121 |
| . | | | . | |
| . | | | . | |
| . | | | . | |
| LSB | LSB/S8 | LSB/S7 | ... | LSB/S1 |
| Block | | | ⋮ | |
| | LSB/S128 | LSB/S127 | ... | LSB/S121 |

***(b) Voice-packet information field***

**Figure 5 – Bit-ordering within voice-packet information fields**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Frame header | Layer 3 header | Most significant nondroppable bits | 2nd most significant nondroppable bits | Last 128 droppable bits | Second 128 droppable bits | First 128 droppable bits | Frame trailer |

**Figure 6 – Block arrangement for (5,2) ADPCM coding**

| 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | |
|---|---|---|---|---|---|---|---|---|
| Address (upper subfield) | | | | | | 0 | 0 | Octet 1 |
| Address (lower subfield) | | | | | | | 1 | Octet 2 |
| UI control field | | | | | | | | |
| 0 | 0 | 0 | P | 0 | 0 | 1 | 1 | Octet 3 |
| Protocol discriminator | | | | | | | | |
| 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | Octet 4 |
| Block-dropping indicator | | | | | | | | |
| R | R | 0 | 0 | R | R | 0 | 0 | Octet 5 |
| Time stamp | | | | | | | | Octet 6 |
| 0<br>M | R | R | R | R | R | R | N/A | Octet 7 |
| Sequence number | | | | A | B | C | D | Octet 8 |
| Check sequence 2 octets | | | | | | | | |

M = More bit
P = Poll bit = 0 (See 4.2.4)
R = Reserved for future use

**Figure 7 – UI signaling frame format**

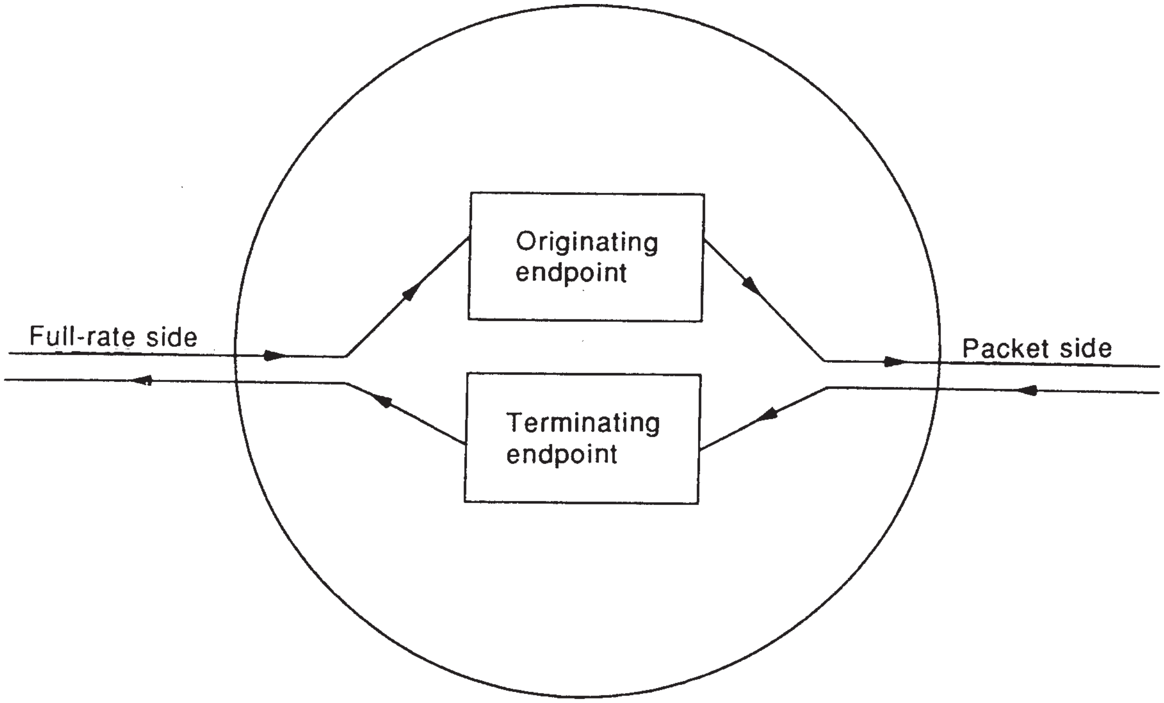


Figure 8 – Endpoint node

## 6 Voice transport procedures

Voice transport procedures are divided into originating, intermediate, and terminating node (endpoint) procedures. Originating endpoints are nodes where user data is formatted into PVP packets for transport. Intermediate nodes are nodes that do not alter the packet format, but simply receive and transport PVP packets. Terminating endpoints are the destination nodes for PVP packets. The list of all primitives used in the procedures is given in clause 10. It is assumed that the processing of primitives requires a fixed amount of time. Any time variance in the processing of primitives shall be accounted for in the value of the timer TVDELAY_V.

Figure 8 illustrates a functional viewpoint of an endpoint node, which shows that it consists of an originating endpoint and a terminating endpoint.

### 6.1 Originating endpoint procedures

The originating endpoint receives segmented data from a higher layer entity via the primitives PL_START_REQUEST(CODE,NOISE), PL_DATA_REQUEST(CODE,NOISE) and PL_STOP_REQUEST(CODE,NOISE). These primitives include information on the type of encoding and noise level associated with the packet.

#### 6.1.1 Receipt of PL_START_REQUEST primitive

The higher-level entity will send to the layer-3 entity the PL_START_REQUEST(CODE, NOISE) primitive after it has collected all the samples of the first packet. When the originating endpoint receives the PL_START_REQUEST(CODE, NOISE) primitive, the layer-3 entity shall start the timer TVDELAY_V associated with that first packet and shall form a voice packet with M-bit set to 1 and SEQ set to 0. The BDI, CT, and Noise fields are encoded based on the coding type and noise level indicated in the primitive PL_START_REQUEST(CODE, NOISE).

The layer-3 entity sets the send sequence state variable (SSEQ) to 1 and checks its own congestion level indicator (CLI) to determine whether blocks should be dropped from the packet and the number of blocks to be dropped. If the CLI is greater than 0, then the block-dropping procedures of 6.4 shall be followed. If the CLI is 0, then the block-dropping procedures are omitted. The CLI is a local parameter of the node.

The layer-3 entity shall buffer the packet until notified by the layer-2 entity that layer 1 is ready to transport data. In the absence of facility alarms, this notification shall be conveyed by the primitive DL_L1_READY_INDICATION. Upon receipt of the primitive, the timer TVDELAY_V is stopped and its value shall be copied to the TS field. The value of the TS field shall not exceed 200.

The packet is then delivered to the layer-2 entity with the primitive DL_UNIT_H_DATA_REQUEST.

### 6.1.2 Receipt of the PL_DATA_REQUEST primitive

After receiving the PL_DATA_REQUEST (CODE,NOISE) primitive from a higher-layer entity, the PVP layer-3 entity shall start the timer TVDELAY_V associated with that packet and form a voice packet with the M-bit set to 1 and the SEQ set to the value of the SSEQ. The BDI, CT, and NOISE fields are encoded on the basis of the corresponding information in the PL_DATA_REQUEST primitive. The layer-3 entity increments the SSEQ (value from 1 to 15 with a rollover to 1). It shall check the CLI to determine the need for block-dropping. If the CLI is greater than 0, then the block-dropping procedure of 6.4 shall be followed. If the CLI is 0, then the block-dropping procedure shall be omitted.

The layer-3 entity shall await the arrival of the DL_L1_READY_INDICATION primitive from layer-2. Upon receipt of the primitive, the timer TVDELAY_V is stopped and its value shall be copied to the TS field. The layer-3 entity shall pass the voice packet to the layer-2 entity for transport using the DL_UNIT_H_DATA_ REQUEST primitive.

### 6.1.3 Receipt of the PL_STOP_REQUEST primitive

When a higher-layer entity detects a gap in the speech, it will continue the packetization until all 128 samples have been packetized. It will then send the PL_STOP_REQUEST primitive to the layer-3 entity. The layer-3 entity shall follow the procedures outlined above following receipt of the PL_DATA_REQUEST, except that it shall set the M-bit to 0.

**Table 5 – Blocks collected during the packetization interval for various coding types**

| Coding type | Number of blocks |
|---|---|
| 8 bits/sample | 8 |
| 1 bit/sample | 1 |
| 2 bits/sample | 2 |
| 3 bits/sample | 3 |
| 4 bits/sample | 4 |
| 5 bits/sample | 5 |
| 6 bits/sample | 6 |
| 7 bits/sample | 7 |
| 8 bit PCM (A-law or μ-law) | 8 |
| 2 bits/sample ADPCM | 2 |
| 3 bits/sample ADPCM | 3 |
| 4 bits/sample ADPCM | 4 |
| 5 bits/sample ADPCM | 5 |
| (4,2) embedded ADPCM | 4 |
| (5,2) embedded ADPCM | 5 |
| (8,6) embedded ADPCM | 8 |

### 6.1.4 Number of blocks and packetization interval

The packetization interval is 16 ms. The number of blocks of 128 bits collected during this interval depends on the coding type, as shown in table 5.

### 6.1.5 Coder reset

When the coding type represents that described in ANSI T1.303, T1.306, or T1.310, a voice packet at the beginning of a speech burst (i.e., with SEQ = 0) must start with a reset coder.

Interoperability issues with PCM coding is an issue for further study.

## 6.2 Intermediate node procedures

Upon receipt of the DL_PVP_H_DATA_ INDICATION primitive, the layer-3 entity will start timer TVDELAY_V associated with that packet. The layer-3 entity shall examine the value encoded in the PD field. If this value matches that

**Table 6 – Valid BDI/CT combinations**

| Coding type | M-subfield value | C-subfield value |
|---|---|---|
| PCM | 0 | 0 |
| Fixed ADPCM | 0 | 0 |
| (*m,n*) Embedded ADPCM | (*m*–*n*) | ≤(*m*–*n*) |

for PVP, the layer-3 entity shall examine the CLI that is a system variable. The CLI shall be set by the management entity to indicate the number of blocks to be dropped from packets containing droppable blocks (0, 1, 2, or 3 blocks may be specified).

If the CLI is greater than 0, then blocks may be dropped from the packet according to the block-dropping procedures described below. If the CLI is 0, then no block-dropping shall occur.

The packet shall then be buffered until the layer-3 entity receives the primitive DL_L1_READY_ INDICATION from layer 2. Upon receipt of this primitive, the variable delay timer TVDELAY_V shall be stopped and its value shall be used to update the packet's time stamp.The resolution of TVDELAY_V is 1 ms. The value of the TS field shall not exceed 200 ms.

The layer-3 entity shall then pass the information to the layer-2 entity via the DL_PVP_ H_DATA_REQUEST primitive.

## 6.3 Terminating endpoint procedures

Upon receipt of the DL_UNIT_H_DATA_ INDICATION primitive, the layer-3 entity shall examine the value encoded in the PD field. If this value matches that for PVP, the layer-3 entity will proceed as below.

### 6.3.1 Illegal BDI/coding type combination

The packet is dropped if the combination of the CT field and the BDI is illegal. The state variable RSEQ is not updated.

Illegal combinations are those in which the C-subfield value, the M-subfield value, or both values of the BDI field exceed the number of droppable bits specified by the coding type. Valid combinations are defined in table 6.

### 6.3.2 Wrong packet length

A voice packet shall be dropped if its length is not consistent with the BDI and CT fields. The state variable RSEQ shall not be updated. The following equation gives the valid packet length based on the BDI subfield values and the CT:

$$l = \frac{[\,S-(M-C)] \times R \times T}{8} + 5\,,$$

where:

*l* is the packet length in octets;

*S* is the number of bits per sample (from coding type);

*M* is the value of M-subfield (from BDI);

*C* is the value of C-subfield (from BDI);

*R* is the sampling rate (8000 samples/s);

*T* is the sampling period (16 ms).

### 6.3.3 Play-out procedures

#### 6.3.3.1 Decoder reset

When the coding type describes that of ANSI T1.303, T1.306, or T1.310, a voice packet at the beginning of a speech burst (i.e., with SEQ = 0) must cause the decoder to reset. This is done by sending to the Management Entity the primitive MPL_DECODER_RESET_REQUEST.

Interoperability issues with PCM coding is an issue for further study.

#### 6.3.3.2 Build-out delay procedures

The build-out delay is a system variable that defines at each end the maximum allowable variable delay in the transmission path. The purpose is to mask the variability in the delay that each packet may experience so that all the packets are played at regular intervals, thereby achieving packet voice synchronization. This value shall be always ≤ 199 ms. Packets that experience delays beyond the build-out value are discarded. The policy of packet replacement is left for further study.

Packets that are played out based on the TS value are:

– packets with SEQ = 0, i.e., the first packet in a voice burst and all signaling packets;

– the voice packet that follows a missing packet, i.e., SEQ in the received packet is different from receive sequence state variable (RSEQ).

When the time for playing out a packet is based on the TS, the duration it is held before play-out is given by:

(Build-out delay) – (TS value) =

(ms to hold packet before play-out)

The packet shall be played out at the end of this period.

Packets with non-ZERO sequence numbers that are received in sequence are played out without a gap after the preceding packet.

The state variable RSEQ is updated after a voice packet has been scheduled for play-out.

#### 6.3.3.3 Embedded ADPCM

For embedded ADPCM, the receiving end determines the algorithm to use for decoding the speech through the BDI and CT fields.

#### 6.3.3.4 Absence of packets

In the absence of packets to be played, the M-bit of the previous packet can be used to determine whether an interpolation procedure is necessary.

The M_LAST system variable stores the value of the M-bit in the last packet.

If M_LAST = 0, the gap is legitimate and the terminating endpoint shall play out the background noise level that corresponds to the value encoded in the NS field of the last received voice packet, as given in table 4. If M_LAST = 1, then a packet was lost. Recommended interpolation procedures, e.g., noise fill or last packet replay, are left for further study.

### 6.4 Block-dropping procedures

Embedded coding algorithms allow for droppable and nondroppable bits within the packet. Dropping the first droppable bit of each sample corresponds to dropping the last block in the packet.

When the CLI specifies the dropping of one or more blocks, the layer-3 entity shall determine from the C-subfield (in the BDI field) of a packet the number of droppable blocks available for dropping. The number of blocks that can be still dropped is given by:

min(value in C-subfield, value in CLI)

The C-subfield of the BDI shall be updated to indicate the number of blocks that can be dropped at the following nodes. This number can be set to 0 if no blocks can be dropped at the following nodes.

## 7 Signaling transport procedures

### 7.1 General principles

Channel-associated signaling is transported across the packet network using signaling packets. To minimize the incorrect delivery of signaling information, channel-associated signaling is transported in a UI frame with a different logical address than that of the corresponding UIH frame that transports the voice information.

There are two types of signaling packets: signaling transition packets and refresh packets. Both have the same structure, as shown in figure 7, and are enclosed in UI frames. The originating end sends a signaling transition packet whenever the signaling state changes. It sends refresh packets on a periodic basis to indicate that the link is still active.

The endpoints will generate and receive signaling packets for each virtual circuit provisioned for channel-associated signaling. To account for a variety of signaling schemes, the endpoints shall provide for the following variations:

– No signaling packets;

– Signaling refresh packets only;

– Signaling refresh packets and signaling transition packets for 2-state signaling using the A-bit. Changes in the A-bit result in transition packets while other bits are ignored;

– Four-state signaling using the A- and B-bits so that signaling refresh packets and signaling transition packets for changes in the A- and B-bits result in transition packets while other bits are ignored;

– Signaling refresh packets and signaling transition packets for 16-state signaling using the A-, B-, C-, and D-bits so that transitions in any of the four signaling bits trigger transition packets.

The number of signaling states must be the same for both the originating and terminating endpoints on a virtual circuit basis. The values coded in this field depend on the framing type and the number of signaling states.

It is assumed that the processing of primitives requires a fixed amount of time. Any time variance in the processing of primitives shall be accounted for in the value of the timer TVDELAY_SIG.

### 7.2 Originating endpoint procedures

The Management Entity of the originating endpoint shall perform the following procedures for each virtual circuit provisioned to support channel-associated signaling:

a) Determine, once per extended superframe, the current state of the ABCD bits and determine whether a transition has occurred in accordance with the number of signaling states supported.

b) When a transition occurs, the Management Entity shall send the MPL_SIG_PKT_ REQUEST(A,B,C,D,N/A) primitive to the originating end of the PVP entity. This originating endpoint must then transmit a transition signaling packet containing the current signaling and alarm states.

The originating endpoint starts in the NORM state. In this state, signaling packets (refresh packets) with the current signaling and alarm states as indicated by the N/A bit must be sent at least every TSIG_REF seconds. The default value for the refresh timer, TSIG_REF, is 10 seconds.

The N/A bit is set to 0 as long as there are no facility alarms (out-of-frame, Red or Yellow alarms) and that the TSIG_KA timer has not expired. The N/A bit shall be set to 1 if there is a facility alarm or if the TSIG_KA timer of the associated terminating end has expired (i.e., the terminating end is in state L_ALARM). Upon the occurrence of a facility alarm, the originating end of the PVP entity shall move from the NORM state to the ALARM state and shall stop transmitting transition packets. It shall continue to send refresh packets every TSIG_REF seconds with the N/A bit set to 1.

The signaling state is frozen until the alarm condition is terminated and the originating endpoint moves back to the NORM state. In the NORM state, the originating endpoints send packets with the N/A bit set to 0 after the associated terminating endpoint has received a signaling packet.

### 7.3 Intermediate node signaling procedures

Upon receipt of the DL_PVP_DATA_INDICATION primitive, the layer-3 entity will start the timer TVDELAY_SIG associated with that packet. The layer-3 entity shall examine the value encoded in the PD field. If this value matches that for PVP, the layer-3 entity shall buffer the packet until it receives the primitive DL_L1_READY_ INDICATION from layer 2. Upon receipt of this primitive, the variable delay timer TVDELAY_SIG shall be stopped and its value shall be used to update the packet's time stamp. The resolution of TVDELAY_SIG is 1 ms. The value of the TS field shall not exceed 200 ms.

The layer-3 entity shall then pass the signaling information to the layer-2 entity via the DL_PVP_DATA_REQUEST primitive.

### 7.4 Terminating endpoint procedures

The terminating endpoint shall perform the following procedures for each 64 kbit/s stream provisioned for channel-associated signaling:

a) In the NORM state: Upon receipt of a signaling packet, the terminating endpoint shall build out the packet according to the time stamp, and reinsert the ABCD bits into the PCM bit stream. The build-out procedure is the same as for voice packets (see 6.3.3.2) except that at play-out time, the layer-3 entity informs the management entity of the signaling packet with the primitive MPL_SIG_PKT_INDICATION(A,B,C,D, N/A). The terminating endpoint shall continue to use the most recently received signaling bits until another signaling packet is received.

b) If no signaling packet is received for a time TSIG_KA (i.e., TSIG_KA has expired), the terminating endpoint shall move to the L_ALARM state and shall initiate trunk conditioning towards the full-rate access side.

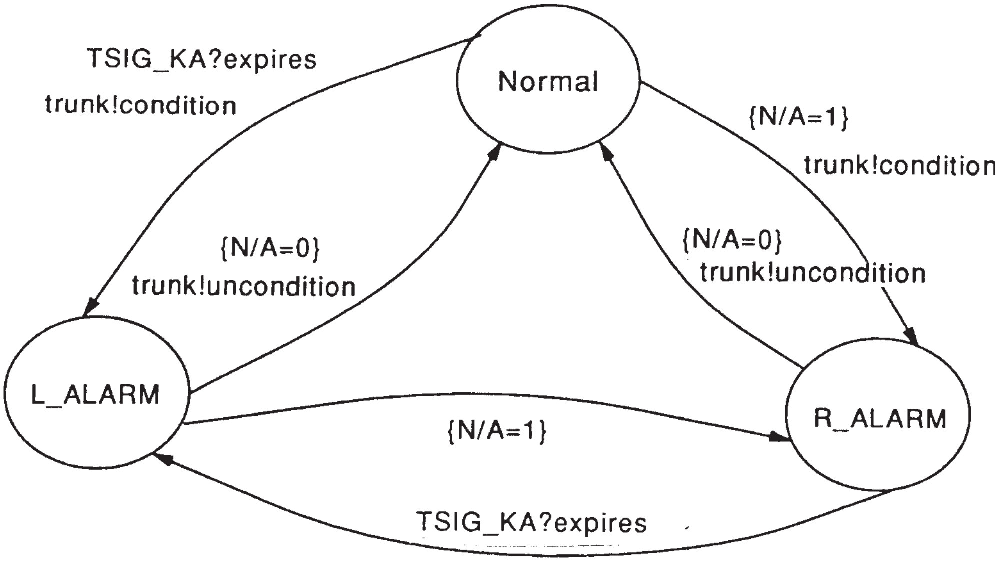


**Figure 9 – Signaling termination endpoint**

The default value for TSIG_KA is 25 seconds. Upon receipt of a signaling packet with the N/A bit set to 0, the terminating end of the PVP entity shall move to the NORM state and shall terminate the trunk conditioning.

c) In the NORM state: Upon receipt of a signaling packet with the N/A bit set to 1, the terminating end of the PVP entity shall move to the R_ALARM state and shall initiate trunk conditioning towards the full-rate access side. It shall return to the NORM state and terminate the trunk conditioning when it receives a signaling packet with the N/A bit set to 0.

d) If the TSIG_KA expires in the R_ALARM state, the terminating end shall move to the L_ALARM state.

e) If a signaling packet arrives with N/A =1 in the L_ALARM state, the terminating end shall move to the R_ALARM state.

Figure 9 shows the transition states for the signaling termination endpoint.

## 7.5 Signaling states

### 7.5.1 Originating-end signaling states

The originating end has two signaling states, as follows.

#### 7.5.1.1 Normal state

In this state, there are no access facility alarms on the full-rate access side.

#### 7.5.1.2 Alarm state

This is the state in which there is a facility alarm on the full-rate access side.

### 7.5.2 Terminating-end signaling states

The terminating endpoint has three signaling states:

#### 7.5.2.1 Normal state (NORM)

The terminating endpoint remains in this state as long as signaling packets arrive with the N/A bit set to 0 and TSIG_KA has not expired.

#### 7.5.2.2 Loss of keep alive alarm (L_ALARM)

The timer TSIG_KA has expired without the reception of a signaling packet. This indicates that a failure has occurred with the packet portion of the connection and normal signaling packet transport has been interrupted.

#### 7.5.2.3 Remote alarm (R_ALARM)

Receipt of signaling packet with the N/A bit set to 1 indicates that the far end is experiencing an alarm condition.

## 8 System variables

### 8.1 Send sequence state variable

Each transmitting endpoint shall have an associated SSEQ that stores the value of the SEQ of the next packet to be transmitted. SSEQ can take on the value of 0 through 15 and is incremented by 1 after each successful packet transmission. When SSEQ has the value 15 and another packet of the same voice burst is transmitted, the value of SSEQ is updated to 1.

### 8.2 Receive sequence state variable

Each terminating endpoint shall have an associated RSEQ that stores the sequence number of the next in-sequence voice packet expected to arrive. RSEQ can take on the value of 0 through 15 and is incremented by 1 (with rollover to 1) after the voice packet is scheduled for play-out. RSEQ takes on the value 0 only when the last packet of a voice burst has been scheduled for play-out.

### 8.3 M_LAST variable

M_LAST stores the value of the M-bit in the last packet.

### 8.4 Congestion level indicator variable

The CLI is set by the management entity to indicate the number of blocks to be dropped from packets containing droppable blocks (0, 1, 2, or 3 blocks may be specified).

## 9 Protocol parameters

### 9.1 Build-out delay

The value of the build-out delay shall be equal to the maximum allowable variable delay in the transmission path of voiceband traffic. This value shall be ≤ 199 ms. The resolution of the build-out delay is 1 ms.

### 9.2 TSIG_REF

This is the interval between successive transmissions of refresh signaling packets from the originating endpoint of the node containing the PVP entity. It may take the values of 1, 5, 10, or 20 seconds. The default value is 10 seconds.

### 9.3 TSIG_KA

This is the maximum time allowed without receiving a signaling packet at the terminating end of a node containing the PVP entity before the terminating equipment must take recovery actions. TSIG_KA is a multiple of TSIG_REF. The multiplier may be set to 1.5, 2.5, 3.5, or 4.5. The default value of the multiplier is 2.5.

### 9.4 TVDELAY_V

This timer is used to measure the variable queuing delay in a node that a voice packet encounters. It is used to update the TS field of a voiceband packet.

### 9.5 TVDELAY_SIG

This timer is used to measure the variable queuing delay in a node that a signaling packet encounters. It is used to update the TS field of a signaling packet.

## 10 Summary of primitives

### 10.1 Primitives for the interface with layer 2

The layer-2 primitives used for PVP are described in 10.1.1 – 10.1.9.

#### 10.1.1 DL_L1_READY_INDICATION

This primitive is used to indicate to layer 3 that layer 1 is ready for transmission.

#### 10.1.2 DL_UNIT_DATA_REQUEST

The DL_UNIT_DATA_REQUEST primitive is used to request layer-3 messages that are to be transmitted by the data link layer in UI frames using the unacknowledged information transfer service.

#### 10.1.3 DL_UNIT_DATA_INDICATION

The DL_UNIT_DATA_INDICATION primitive is used to indicate receipt by the data link layer of PVP at the terminating endpoint of layer-3 messages that are enclosed UI frames.

#### 10.1.4 DL_UNIT_H_DATA_REQUEST

The DL_UNIT_H_DATA_REQUEST primitive is used to request layer-3 messages that are to be transmitted by the data link layer in UIH frames using the unacknowledged information transfer service.

#### 10.1.5 DL_UNIT_H_DATA_INDICATION

The DL_UNIT_H_DATA_INDICATION primitive is used to indicate receipt by the data link layer of PVP at the terminating endpoint of layer-3 messages that are enclosed in UIH frames.

#### 10.1.6 DL_PVP_H_DATA_INDICATION

The DL_PVP_H_DATA_INDICATION primitive is used to indicate receipt by the data link layer of PVP at intermediate nodes of layer-3 messages in UIH frames.

#### 10.1.7 DL_PVP_DATA_INDICATION

The DL_PVP_DATA_INDICATION primitive is used by the data link layer of PVP intermediate nodes to indicate receipt of layer-3 messages in UI frames.

#### 10.1.8 DL_PVP_H_DATA_REQUEST

The DL_PVP_H_DATA_REQUEST primitive is used by the layer-3 entity of an intermediate node to indicate to the data link layer a layer-3 message to be transported in a UIH frame.

#### 10.1.9 DL_PVP_DATA_REQUEST

The DL_PVP_DATA_REQUEST primitive is used by the layer-3 entity of an intermediate node to indicate to the data link layer a layer-3 message to be transported in a UI frame.

### 10.2 Primitives for the interface with upper layers

#### 10.2.1 PL_START_REQUEST(CODE,NOISE)

The PL_START_REQUEST_(CODE,NOISE) primitive is used by the higher layer of the originating endpoint to request that the layer-3 entity begin formatting voice packets with the CT = CODE and the Noise = NOISE.

#### 10.2.2 PL_DATA_REQUEST_(CODE,NOISE)

The PL_DATA_REQUEST_(CODE,NOISE) primitive is used by the higher layer of the originating endpoint to continue the formatting voice packets with the CT = CODE and the Noise = NOISE.

#### 10.2.3 PL_STOP_REQUEST

The PL_STOP_REQUEST primitive is used by upper layers to indicate the end of a speech burst.

### 10.3 Primitives for the interface with the management entity

#### 10.3.1 MPL_DECODER_RESET_REQUEST

The MPL_DECODER_RESET_REQUEST primitive is used by layer 3 of the PVP terminating endpoint to request resetting of the decoder by the management entity.

#### 10.3.2 MPL_SIG_PKT_REQUEST(A,B,C,D,N/A)

This primitive is used by the management entity to request transmission of a transition signaling packet by the PVP layer 3.

#### 10.3.3 MPL_SIG_PKT_INDICATION(A,B,C,D, N/A)

This primitive is used by the layer-3 entity to inform of the management entity of the playout of a signaling packet by the PVP layer 3.

## 11 Issues for further study

The following issues are left for further study:

– applications of the (5,4), (5,3), (4,3), (3,3), and, (3,2) ADPCM algorithms;

– the policy of packet replacement and/or actions upon detection of lost packets;

– the making of PVP packets for primary interfaces (described in CCITT Recommendation G.704) carrying PCM frame structures (described in CCITT Recommendation H.221);

– interoperability of equipment using clear channels;

– internetworking with the cellular speech packetization network;

– interoperability issues for PCM coding.

# Annex A
(informative)

# Formal description of the voice packetization protocol

This annex consists of an introduction to the Augmented Protocol Specification Language (APSL) used to formally describe the protocol, the APSL description of ANSI T1.312-1991, and state transition diagrams for ANSI T1.312-1991.

## A.1 The Augmented Protocol Specification Language (APSL)

The Augmented Protocol Specification Language (APSL) is an extension of the Protocol Specification Language (PSL) developed by Sabnani and Lapone to specify protocols formally using communicating Finite State Machines (FSMs). Formal verification deals with the behavior of these FSMs with respect to the external world and does not consider the internal events of the protocol. Therefore, PSL does not include a succinct syntax to specify internal operations (e.g., the update of internal counters and setting of flags). A concise way of specifying these low-level details is necessary in the design and implementation of hardware protocol controllers. Therefore, PSL was extended to the APSL, whose syntax and semantics are described in the next clause.

## A.2 The syntax APSL

In APSL, a protocol is described as a collection of communicating FSMs, each specified as an APSL process with its own input and output message sets. External inputs and outputs are designated as belonging to *external* processes. Each communicating FSM is specified as an APSL process in the following form:

```
EXTERNAL  ext_proc1, ... , ext_procn ;
PROCESS  process_name ;
CONTEXT  variable_name ;
STATES  n0-n1, n2-n3, n4, ... ,nk-nl ;
REND   src1?msg1,src1?msg2, ...src1!msga,src1!msgb ;
INITIAL STATE  N0 ;
TRANSITIONS
transition_1,
transition_2,
...
transition_k.
```

The order of the statements is to be followed strictly. The following subclauses explain the meanings of the various declaration statements.

### A.2.1 EXTERNAL statement

The EXTERNAL statement is optional. Its purpose is to allow the independent checking of the specifications of individual processes and to model external inputs and outputs in a consistent manner.

### A.2.2 PROCESS statement

The PROCESS statement names the process. Acceptable names are alphanumeric strings that start with an alphabetic character.

### A.2.3 CONTEXT statement

The CONTEXT statement lists the context variables of the process. Acceptable variable names are alphanumeric strings that start with an alphabetic character. Context variable may be entered in any particular order.

### A.2.4 STATES statement

The STATES statement lists the states of the process. State names may be entered in any particular order but are currently restricted to numbers. The compiler ignores unused states and renumbers all states sequentially in the internal representation of the processes. A range *n0-n1* indicates all the numbers from *n0* to *n1*, both inclusive.

### A.2.5 REND statement

The REND (from *rendez-vous*) statement defines all the I/O messages associated with the process. These are the messages through which the FSMs of the protocol communicate among themselves and with the external world. Messages are of the form *a?mi* or *b!mj* , where *a?mi* indicates the reception of message *mi* from the machine *a* and *b!mj* indicates the transmission of message *mj* to the machine *b* . The semantics of this notation are those of the CSP notation developed by Hoare, the key characteristic being that it requires both a sender and a receiver and is a synchronizing primitive.

### A.2.6 INITIAL STATE statement

The initial state of the machine is specified by the INITIAL STATE statement. The initial state must be among the states specified in the STATES statement.

### A.2.7 TRANSITIONS statement

The general form of a transition in APSL is

```
i:j WHEN {condition} a?m1 * b!m2 [update-operation]
```

where the syntactic elements *condition* and *update-operation* are extensions to PSL. The *condition* is a test involving a conditional expression containing internal (context) variables. Updates of internal variables are done in the *update-operation* specification.

Although no major restrictions are placed on the *condition* and *update-operation* parts of the transition specification, hardware architectures may restrict the complexity of the condition that can be tested in one cycle. A *simple* condition is an expression of the type

```
a > b
```

or

```
a + b <= c
```

where *a* , *b* , and *c* are variables or constants. Furthermore, most hardware implementations allow only one assignment per cycle. Therefore, we define the *canonical* form of the specification to include *simple conditions* only and to have at most one *update-operations* . The user writes an initial protocol description and the compiler translates it into the *canonical* form, generating any needed intermediate states. For example, the user may input the following description:

```
i:j WHEN { C1 | (C2 & C3 ) } a?m1 * b!m2 [ A1; A2; ]
```

where *C1* are conditional statements and *A1* are update operations, becomes after translation:

```
i:t1  WHEN {C1} a?m1 * b!m2 [A1;];
t1:j  WHEN [A2;];
i:t2  WHEN {C2};
t2:i  WHEN {~C3};
t2:t1 WHEN {C3} a?m1 * b!m2 [A1;].
```

Here *t1* and *t2* are intermediate states that the APSL compiler have introduced; they remain invisible to the external world. The update operations are restricted to increment, decrement, add, subtract, and assignment operations. For example, if the specification has an input message *a?m1* in machine *b*, the specification of machine *a* must contain the output message *b!m1*. The compiler flags as errors messages those that do not occur in matched pairs. This check is by-passed, however, for machines that are specified as EXTERNAL.

## A.3 Some semantic restrictions

The machines specified in APSL are assumed to be deterministic machines. This requires that, in any given state, only one outgoing edge be enabled under the given conditions.

Consider the statements

```
i:j WHEN {C1} a?m1 * b!m2 [update-operation]
i:k WHEN {C2} c?m4 * b!m3 [update-operation]
```

When the machine is in state *i*, the conditions are checked in some order and the first condition to succeed enables the corresponding edge. Thus, if *C1* is true, the machine goes from *i* to *j* on the reception of message *m1* from machine *a*. While waiting for this message *m1*, the value of *C2* does not affect the outcome; the machine is still committed to the transition *i* to *j*. Furthermore, the set of conditions on all the outgoing edges from a state must be mutually exclusive and collectively exhaustive. In other words, if the conditions are $C_1$, $C_2$, ..., $C_n$, then

$$C_1 \cap C_2 \cap \cdots \cap C_n = \varnothing$$
$$C_1 \cup C_2 \cup \cdots \cup C_n = 1$$

This implies that we cannot mix edges with conditions and edges without conditions from the same state.

## A.4 Processes in APSL description

Table A.1 provides a list of all the processes used in the APSL specification of the packet voice protocol, with a brief description of each:

**Table A.1 – Processes of APSL in PVP**

| Process name | Description |
|---|---|
| xmtr_v<br>interm_pt_v_relay<br>rcv_v<br>playout_v | Voice-originating end<br>Voice-intermediate end<br>Voice-terminating end<br>Voice playout |
| xmtr_sig<br>interm_pt_sig_relay<br>rcv_sig<br>playout_sig | Signaling-originating end<br>Signaling-intermediate end<br>Signaling-terminating end<br>Signaling playout |
| tb_sig<br>tb_v<br>tvdelay_v<br>tvdelay_sig | Buildout-delay timer for signaling<br>Buildout-delay timer for voice<br>Voice-variable delay timer<br>Signaling-variable delay timer |
| tsig_ka<br>tsig_ref | Signaling-keep-alive timer TSIG_KA<br>Signaling-refresh timer TSIG_EF |
| clock<br>fifo<br>facility<br>trunk | Clock<br>FIFO queue<br>Detector of alarm conditions on facilities<br>Controller of trunk conditioning |
| mgmt<br>hi_layer<br>layer_2 | Management entity<br>Higher layer entity<br>Layer 2 |

The following are treated as EXTERNAL processes:

clock, facility, fifo,
hi_layer, layer2, mgmt,
tb_sig,tb_v, trunk, tsig_ka, tsig_ref.

## A.5 Functions in the APSL description of ANSI T1.312

Table A.2 provides a list of the various functions used in the APSL specification of this standard, arranged alphabetically:

**Table A.2 – Functions of APSL**

| Function | Description |
|---|---|
| convert_to_voice( ) | Converts the samples from the packets in the playout queue to a voice signal for playback and sets the value of M_LAST and CT_LAST. |
| drop_block( ) | Drops the next droppable block from the packet. |
| drop_pkt( ) | Drops the received packet because of an error condition. |
|  | Adaptive Differential Pulse-Code Modulation (ADPCM) coding. |
| is_compat( ) | Checks if the coding type (CT) field is compatible with the Block-Dropping Indicator (BDI). |
| length_ok( ) | Checks if the length of the packet is compatible with the BDI and CT fields. |
| output_delay( ) | Outputs the delay between reception and transmission of a packet at a node. This delay is measured by the tvdelay_v process for voice packets and tvdelay_sig process for signaling packets. |
| playout_last_pkt( ) | Plays back the last packet. |
| playout_noise_pkt( ) | Plays back a noise packet as defined in the Noise field. |
| queue_pkt( ) | Queues a received packet, whose play-out time has not arrived yet, in the proper sequence in a packet queue. |
| rseq_increment( ) | Same as seq_increment, except that it operates on the state variable RSEQ. |
| seq_increment( ) | Increments the sequence number (SEQ) (1 to 15 with rollover to 1). |
| ts_update( ) | Updates the time stamp by the value of the variable delay measured by tvdelay_v and tvdelay_sig timers for voice and signaling, respectively. |

## A.6 Primitives

Table A.3 provides a description of the various primitives used in the APSL description:

**Table A.3 – Primitives of APSL**

| Primitive | APSL representation |
|---|---|
| DL_L1_READY_INDICATION | dl_l1_ready_indication |
| DL_PVP_DATA_REQUEST | dl_pvp_data_request |
| DL_PVP_DATA_INDICATION | dl_pvp_data_indication |
| DL_PVP_H_DATA_REQUEST | dl_pvp_h_data_request |
| DL_PVP_H_DATA_INDICATION | dl_pvp_h_data_indication |
| DL_UNIT_DATA_INDICATION | dl_unit_data_indication |
| DL_UNIT_DATA_REQUEST | dl_unit_data_request |
| DL_UNIT_H_DATA_INDICATION | dl_unit_h_data_indication |
| DL_UNIT_H_DATA_REQUEST | dl_unit_h_data_request |
| MPL_DECODER_RESET_REQUEST | mpl_decoder_reset_request |
| PL_DATA_REQUEST(CODE,NOISE) | pl_data_request |
| PL_START_REQUEST(CODE,NOISE) | pl_start_request |
| PL_STOP_REQUEST | pl_stop_request |
| MPL_SIG_PKT_INDICATION(A,B,C,D,N/A) | mpl_sig_pkt_indication |
| MPL_SIG_PKT_REQUEST(A,B,C,D,N/A) | mpl_sig_pkt_request |

## A.7 Internal variables

Table A.4 lists internal variables used in the APSL description:

**Table A.4 – Variables of APSL**

| Variable | Description |
|---|---|
| compat | Set or cleared by the function is_compat( ) |
| cli | CLI variable |
| ct | Coding type of the current packet |
| ct_last | CT_LAST set by convert_to_voice( ) |
| delay_value | Value of the variable delay timer |
| have_room | = 1 when voice packet can be stored;<br>= 0 otherwise. |
| q_empty | = 1 when playout queue is empty;<br>= 0 otherwise. |
| length_valid | Stores the output from length_ok( ) |
| m | M-bit value |
| m_last | M_LAST variable set by convert_to_voice( ) |
| pd | Protocol discriminator |
| rseq | RSEQ variable |
| seq | SEQ variable |
| ts | Time stamp (updated by the ts_update function) |

### A.8 General assumptions

The description assumes that the strategy to replace a lost packet is to replay the last packet. The current standard has left this point for further study. Also, it is assumed that the buildout delay is 100.

```
/*
          various definitions
                              */
#define PVP  107
#define BUILDOUT 100
#define C_ADPCM 1
#define C_OTHER 0
#define TRUE   1
#define FALSE  0
/*
                    compat         Set or cleared by the function is_compat( )
                    cli            CLI variable
                    ct_last        Coding type of the previous sample
                    delay          Value of the variable delay timer
                    have_room      = 1 when voice packet can be stored;
                                   = 0 otherwise.
                    q_empty        = 1 when playout queue is empty;
                                   = 0 otherwise.
                    length_valid   Stores the output from length_ok( )
                    m              M-bit value
                    m_last         M_LAST variable
                    pd             Protocol discriminator
                    rseq           RSEQ variable
                    seq            SEQ variable
                    ts             Time stamp (updated by the ts_update function)
                                                                                  */
/*
          External Processes
                                         */
EXTERNAL hi_layer, layer2, tb_v, mgmt, fifo, clock, tsig_ref, facility,
          tb_sig, trunk, tsig_ka;
/*

          VOICE PROCESSES
```

```
*/

/*
          Originating End — figure A.1
*/
PROCESS       xmtr_v;
CONTEXT m,seq,cli;
STATES                 0-7;
REND          hi_layer?pl_start_request, hi_layer?pl_stop_request,
              hi_layer?dl_pvp_data_request,
              tvdelay_v!restart, tvdelay_v!reset,
              layer2?dl_l1_ready_indication,
              layer2!dl_unit_h_data_request;
INITIAL STATE 0;
TRANSITIONS

0:1       WHEN hi_layer?pl_start_request,
0:2       WHEN hi_layer?dl_pvp_data_request,
0:3       WHEN hi_layer?pl_stop_request,
3:4       WHEN tvdelay_v!restart [m=0;] ,
1:4       WHEN tvdelay_v!restart [m=1; seq=0;],
2:4       WHEN tvdelay_v!restart [m=1;seq_increment( );],
4:5       WHEN { cli > 0 } [ drop_block( );],
4:5       WHEN { cli == 0 } ,
5:6       WHEN layer2?dl_l1_ready_indication * tvdelay_v!reset,
6:7       WHEN [ts_update( );],
7:0       WHEN layer2!dl_unit_h_data_request.
/*

              Intermediate Point — figure A.2
                                                  */
PROCESS       interm_pt_v_relay;
CONTEXT pd;
STATES                  0-5;
REND          layer2?dl_pvp_h_data_indication,
              layer2!dl_pvp_h_data_request,
              layer2?dl_l1_ready_indication,
              tvdelay_v!restart, tvdelay_v!reset;
INITIAL STATE 0;
TRANSITIONS

0:1       WHEN layer2?dl_pvp_h_data_indication,
1:0       WHEN {pd != PVP},
1:2       WHEN {pd == PVP} tvdelay_v!restart,
2:3       WHEN {cli > 0 } [drop_block( );],
2:3       WHEN {cli == 0 },
3:4       WHEN layer2?dl_l1_ready_indication * tvdelay_v!reset,
4:5       WHEN [ts_update( );],
5:0       WHEN layer2!dl_pvp_h_data_request.
```

```
/*
          Terminating End — figure A.3(a)
                                */
PROCESS         rcv_v;
CONTEXT compat,length_valid,ts,rseq,tb_v_value,ct_last;
STATES            0-14, 50-52;
REND            layer2?dl_unit_h_data_indication, tb_v?expired,
                tb_v!restart, playout_v!enqueue, playout_v!play,
                mgmt!mpl_decoder_reset_request;
INITIAL STATE 0;
TRANSITIONS

0:1      WHEN layer2?dl_unit_h_data_indication,
1:0      WHEN {pd != PVP},
1:2      WHEN {pd == PVP} [is_compat( );],
2:0      WHEN {compat == FALSE},
2:3      WHEN {compat == TRUE} [length_ok( );],
3:0      WHEN {length_valid == FALSE},
3:4      WHEN {length_valid == TRUE},
4:5      WHEN {seq>0},
4:6      WHEN {seq == 0},
5:0      WHEN {ts > BUILDOUT},
5:50     WHEN {ts <= BUILDOUT},
6:8      WHEN {ts <= BUILDOUT},
6:0      WHEN {ts > BUILDOUT},
0:14     WHEN tb_v?expired,


/*
          Terminating End — figure A.3(b)
                              */
8:9      WHEN {ts < BUILDOUT},
9:11     WHEN playout_v!enqueue [rseq_increment( );],
11:0     WHEN tb_v!restart [tb_v_value = BUILDOUT - ts;],
8:10     WHEN {ts == BUILDOUT} playout_v!enqueue [rseq_increment( );],
10:14    WHEN {ct_last == C_ADPCM} mgmt!mpl_decoder_reset_request,
10:14    WHEN {ct_last == C_OTHER},
14:0     WHEN playout_v!play,


/*
          Terminating End — figure A.3(c)
                              */
50:0     WHEN {seq == rseq} playout_v!enqueue [rseq_increment( );],
50:51    WHEN {seq != rseq} [rseq = seq;],
51:52    WHEN playout_v!enqueue [rseq_increment( );],
52:0     WHEN tb_v!restart [tb_v_value = BUILDOUT - ts;].
```

```
/*
          Playout — figure A.4
                                  */
PROCESS       playout_v;
CONTEXT have_room,q_empty,m_last;
STATES                  0-6;
REND            rcv_v?enqueue, rcv_v?play,
                fifo!dequeue;
INITIAL STATE0;
TRANSITIONS

0:1       WHEN rcv_v?enqueue,
1:0       WHEN {have_room == FALSE} [drop_pkt( );],
1:0       WHEN {have_room == TRUE} [queue_pkt( );],
0:2       WHEN rcv_v?play,
2:3       WHEN {q_empty == FALSE},
2:4       WHEN {q_empty == TRUE},
3:0       WHEN fifo!dequeue [convert_to_voice( );],
4:6       WHEN {m_last == 0},
4:5       WHEN {m_last == 1},
5:0       WHEN [playout_last_pkt( );],
6:0       WHEN [playout_noise_pkt( );].



/*
          TVDELAY — figure A.5
                                          */
PROCESS       tvdelay_v;
CONTEXT delay;
STATES          0-1;
REND      interm_pt_v_relay?restart, interm_pt_v_relay?reset,
          xmtr_v?restart, xmtr_v?reset, clock?tick;
INITIAL STATE 0;
TRANSITIONS

0:1       WHEN  interm_pt_v_relay?restart [delay=0;],
0:1       WHEN  xmtr_v?restart [delay=0;],
1:1       WHEN  clock?tick [delay++;],
1:0       WHEN  interm_pt_v_relay?reset[output_delay( );delay=0;],
1:0       WHEN  xmtr_v?reset[output_delay( );delay=0;].


/*
          SIGNALING PROCESSES
```

```
*/
/*
            Originating End — figure A.6
                 NORMAL
                                  */
PROCESS        xmtr_sig;
STATES                     0-103;
REND           tsig_ref?expired, mgmt?mpl_sig_pkt_request,
               tvdelay_sig!reset, facility?recovery, facility?alarm,
               tvdelay_sig!restart,
               layer2!dl_unit_data_request,
               layer2?dl_l1_ready_indication,
               tsig_ref!restart;
INITIAL STATE 0;
TRANSITIONS

0:1         WHEN tsig_ref?expired,
0:1         WHEN mgmt?mpl_sig_pkt_request [ts=0;],
1:2         WHEN layer2?dl_l1_ready_indication * tvdelay_sig!reset,
2:3         WHEN [ts_update( );],
3:0         WHEN layer2!dl_unit_data_request * tsig_ref!restart,
0:100       WHEN facility?alarm,


/*
            Originating End — figure A.7
                ALARM
                                                   */
100:0          WHEN facility?recovery,
100:101        WHEN tsig_ref?expired,
101:102        WHEN layer2?dl_l1_ready_indication * tvdelay_sig!reset,
102:103        WHEN [ts_update( );],
103:100        WHEN layer2!dl_unit_data_request * tsig_ref!restart.


/*
            Intermediate Point — figure A.8
                                                         */
PROCESS        interm_pt_sig_relay;
STATES                      0-4;
REND           layer2?dl_pvp_data_indication,
               layer2!dl_pvp_data_request,
               layer2?dl_l1_ready_indication,
               tvdelay_sig!restart, tvdelay_sig!reset;
INITIAL STATE 0;
TRANSITIONS

0:1         WHEN layer2?dl_pvp_data_indication,
1:0         WHEN {pd != PVP},
1:2         WHEN {pd == PVP} tvdelay_sig!restart,
2:3         WHEN layer2?dl_l1_ready_indication * tvdelay_sig!reset,
3:4         WHEN [ts_update( );],
4:0         WHEN layer2!dl_pvp_data_request.
```

```
0/*
          Terminating End — figures A.9 through A.11
                                */
PROCESS        rcv_sig;
CONTEXT na,tb_sig_value;
STATES                  0-207;
REND           layer2?dl_unit_data_indication,
               playout_sig!enqueue, tsig_ka!restart,
               playout_sig!play, tb_sig!restart, tb_sig?expired,
               tsig_ka?expired, trunk!condition, trunk!uncondition;
INITIAL STATE 0;
TRANSITIONS


/*
          Terminating End — figure A.9
          NORM
                    */
0:1       WHEN layer2?dl_unit_data_indication,
1:0       WHEN {pd != PVP},
1:2       WHEN {pd == PVP},
2:0       WHEN {ts > BUILDOUT},
2:3       WHEN {ts <= BUILDOUT} playout_sig!enqueue*tsig_ka!restart,
3:4       WHEN {ts == BUILDOUT},
4:5       WHEN playout_sig!play,
5:0       WHEN {na == 0},
5:200     WHEN {na == 1} trunk!condition,
3:7       WHEN {ts < BUILDOUT},
7:0       WHEN tb_sig!restart [tb_sig_value = BUILDOUT – ts;],
0:4       WHEN tb_sig?expired,
0:100     WHEN tsig_ka?expired * trunk!condition,


/*
          Terminating End — figure A.10

          L_ALARM
                                      */
100:101             WHEN layer2?dl_unit_data_indication,
101:100             WHEN {pd != PVP},
101:102             WHEN {pd == PVP},
102:100             WHEN {ts > BUILDOUT},
102:103             WHEN {ts <= BUILDOUT} playout_sig!enqueue*tsig_ka!restart,
103:104             WHEN {ts == BUILDOUT},
104:105             WHEN playout_sig!play,
105:0               WHEN {na == 0} trunk!uncondition,
105:100             WHEN {na == 1},
103:107             WHEN {ts < BUILDOUT},
107:100             WHEN tb_sig!restart [tb_sig_value = BUILDOUT – ts;],
100:104             WHEN tb_sig?expired,
/*
        Terminating End — figure A.11

        R_ALARM
                                */
200:201         WHEN layer2?dl_unit_data_indication,
201:200         WHEN {pd != PVP},
201:202         WHEN {pd == PVP},
202:200         WHEN {ts > BUILDOUT},
202:203         WHEN {ts <= BUILDOUT} playout_sig!enqueue*tsig_ka!restart,
203:204         WHEN {ts == BUILDOUT},
204:205         WHEN playout_sig!play,
205:0           WHEN {na == 0} trunk!uncondition,
205:200         WHEN {na == 1},
203:207         WHEN {ts < BUILDOUT},
207:200         WHEN tb_sig!restart [tb_sig_value = BUILDOUT – ts;],
200:204         WHEN tb_sig?expired,
200:100         WHEN tsig_ka?expired.

/*
        PLAYOUT — figure A.12
                                        */
PROCESS         playout_sig;
STATES                  0-2;
REND            rcv_sig?enqueue, rcv_sig?play,
                mgmt!mpl_sig_pkt_indication;
INITIAL STATE 0;
TRANSITIONS

0:1     WHEN rcv_sig?enqueue,
1:0     WHEN {have_room == TRUE} [queue_pkt( );],
1:0     WHEN {have_room == FALSE} [drop_pkt( );],
0:2     WHEN rcv_sig?play,
2:0     WHEN mgmt!mpl_sig_pkt_indication.


/*
TVDELAY — figure A.13
                         */
PROCESS         tvdelay_sig;
STATES                  0-1;
REND            interm_pt_sig_relay?restart, interm_pt_sig_relay?reset,
                xmtr_sig?restart, xmtr_sig?reset, clock?tick;
INITIAL STATE 0;
TRANSITIONS

0:1     WHEN interm_pt_sig_relay?restart [delay=0;],
0:1     WHEN xmtr_sig?restart [delay=0;],
1:1     WHEN clock?tick [delay++;],
1:0     WHEN interm_pt_sig_relay?reset[output_delay( );delay=0;],
1:0     WHEN xmtr_sig?reset[output_delay( );delay=0;].
```

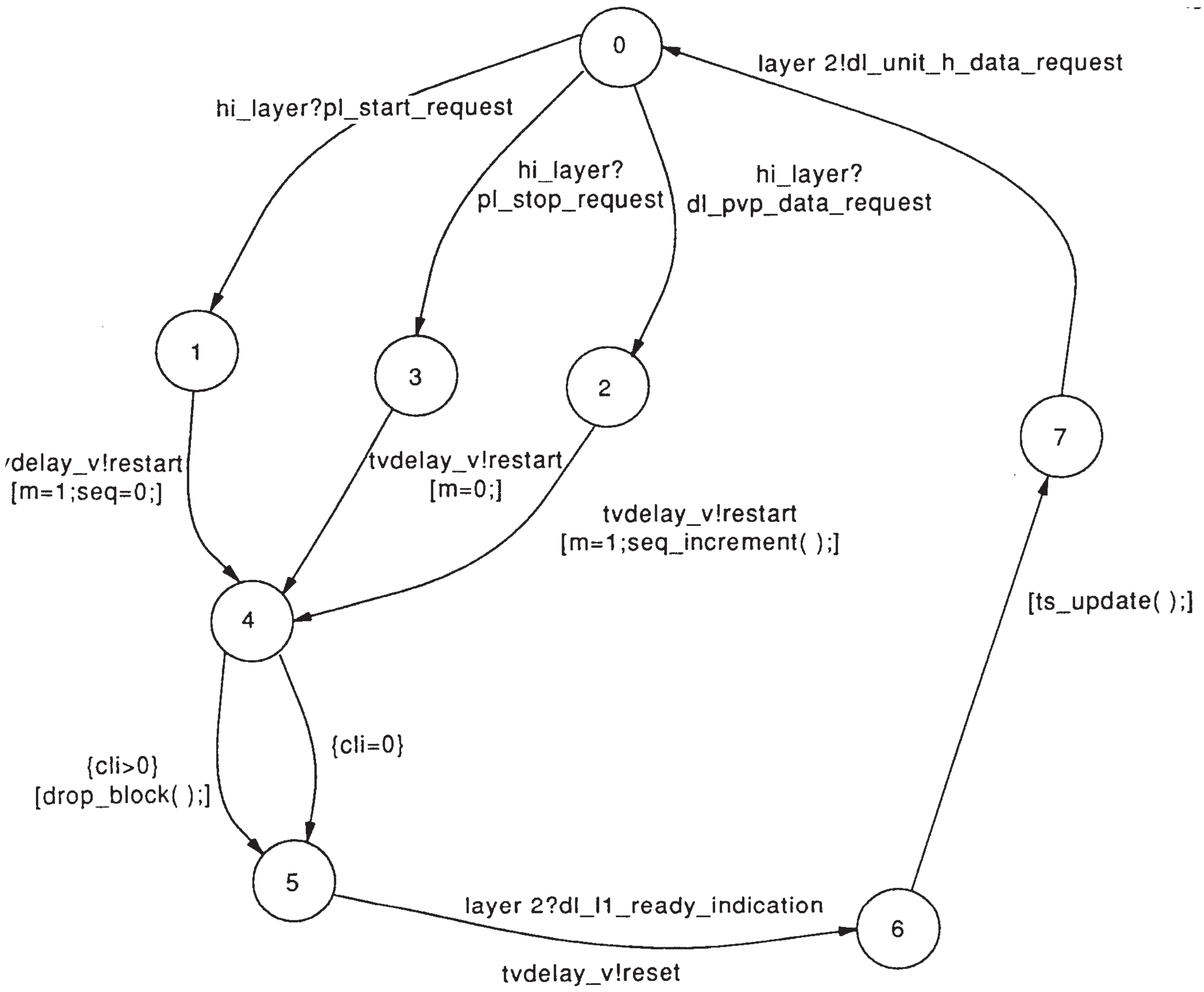


**Figure A.1 xmtr_v**

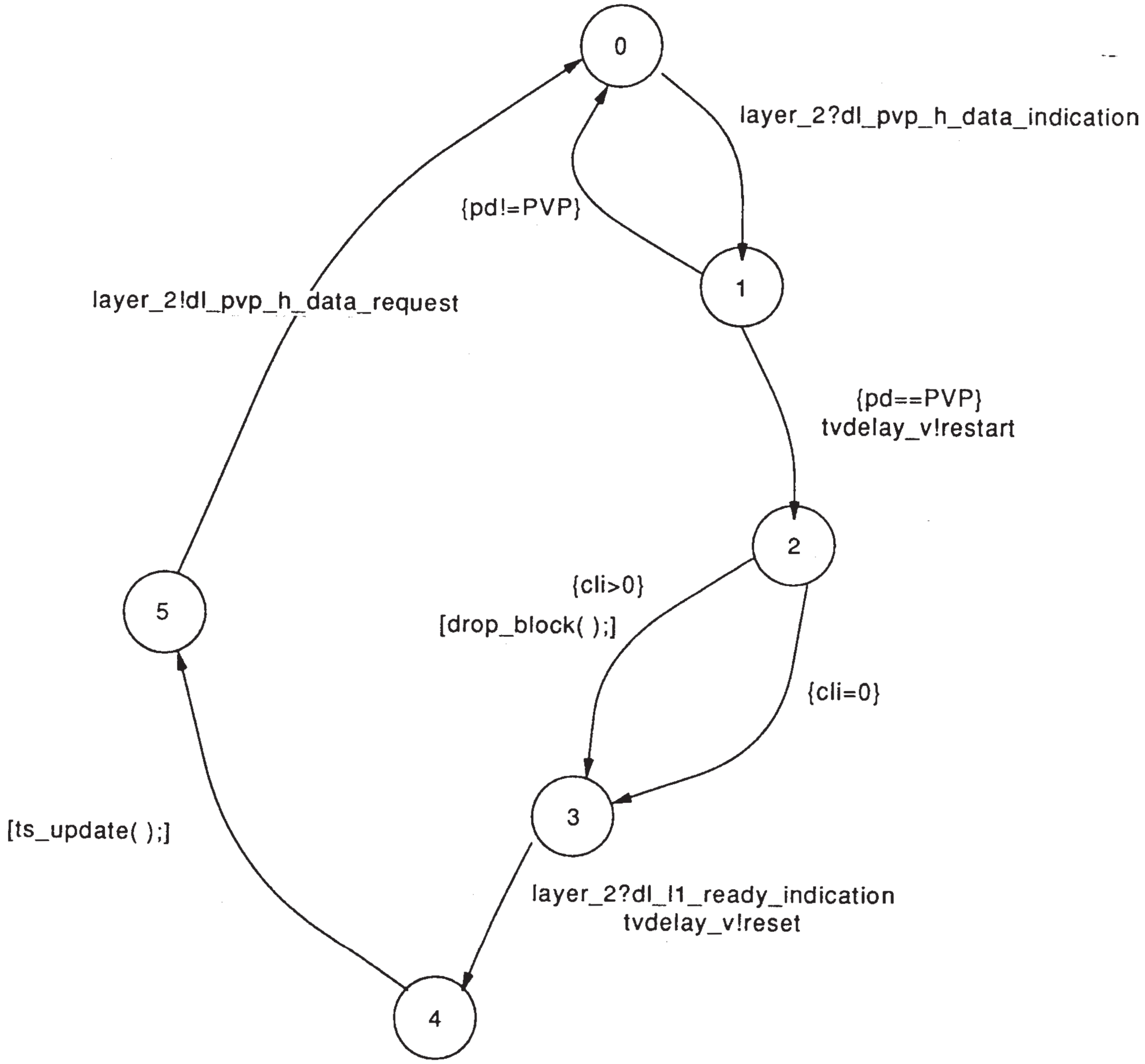
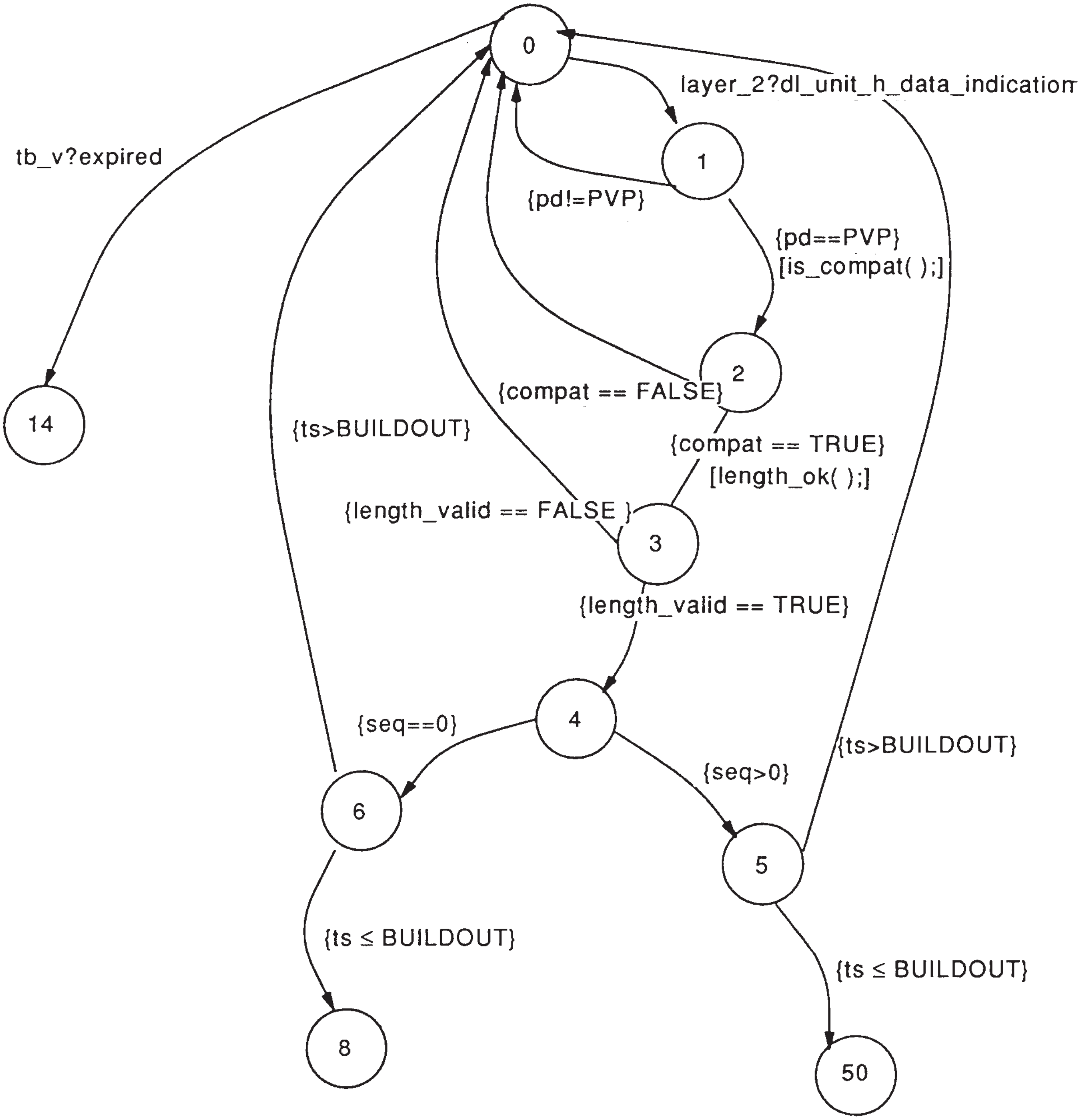
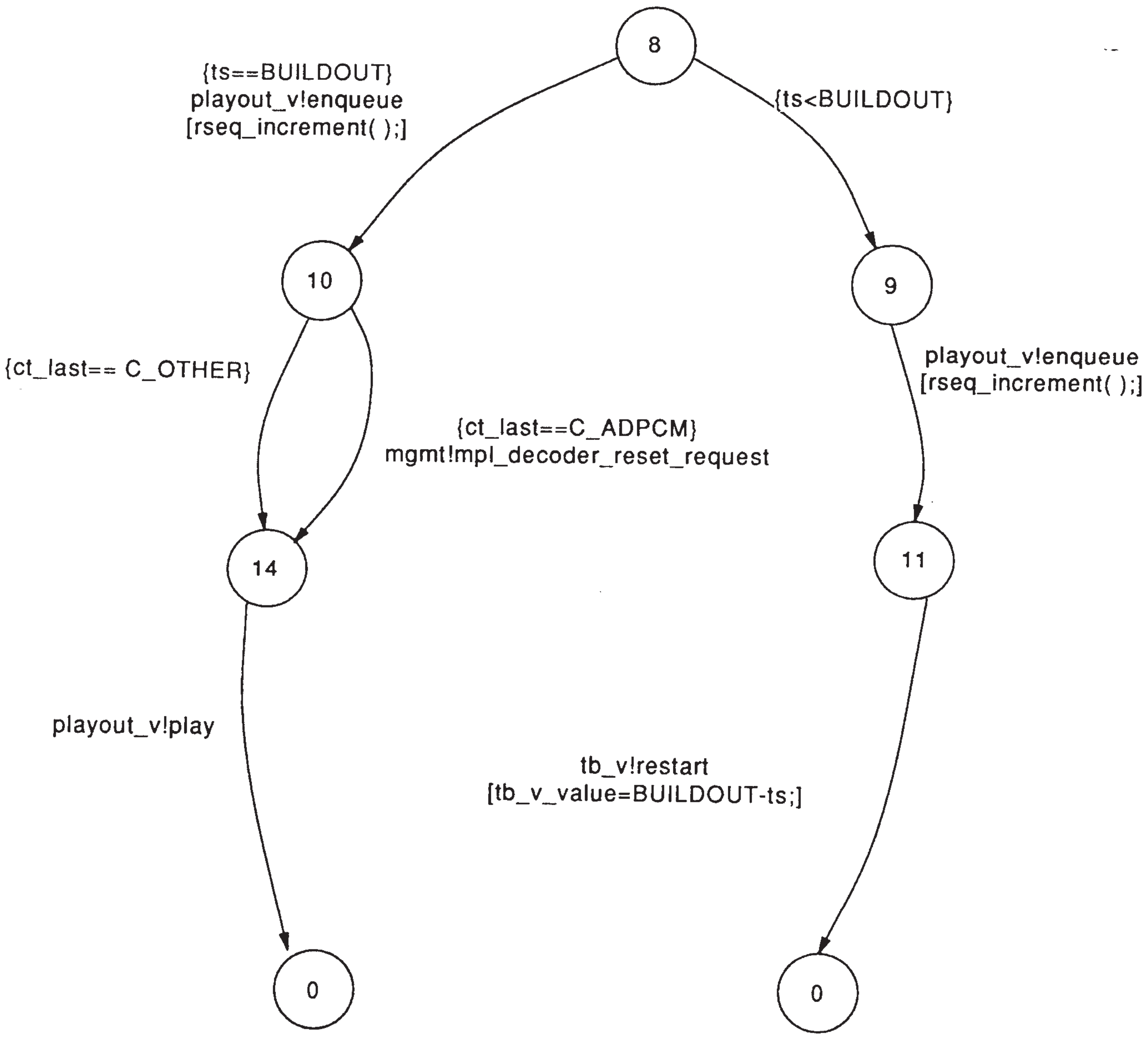


**Figure A.2 interm_pt_v_relay**



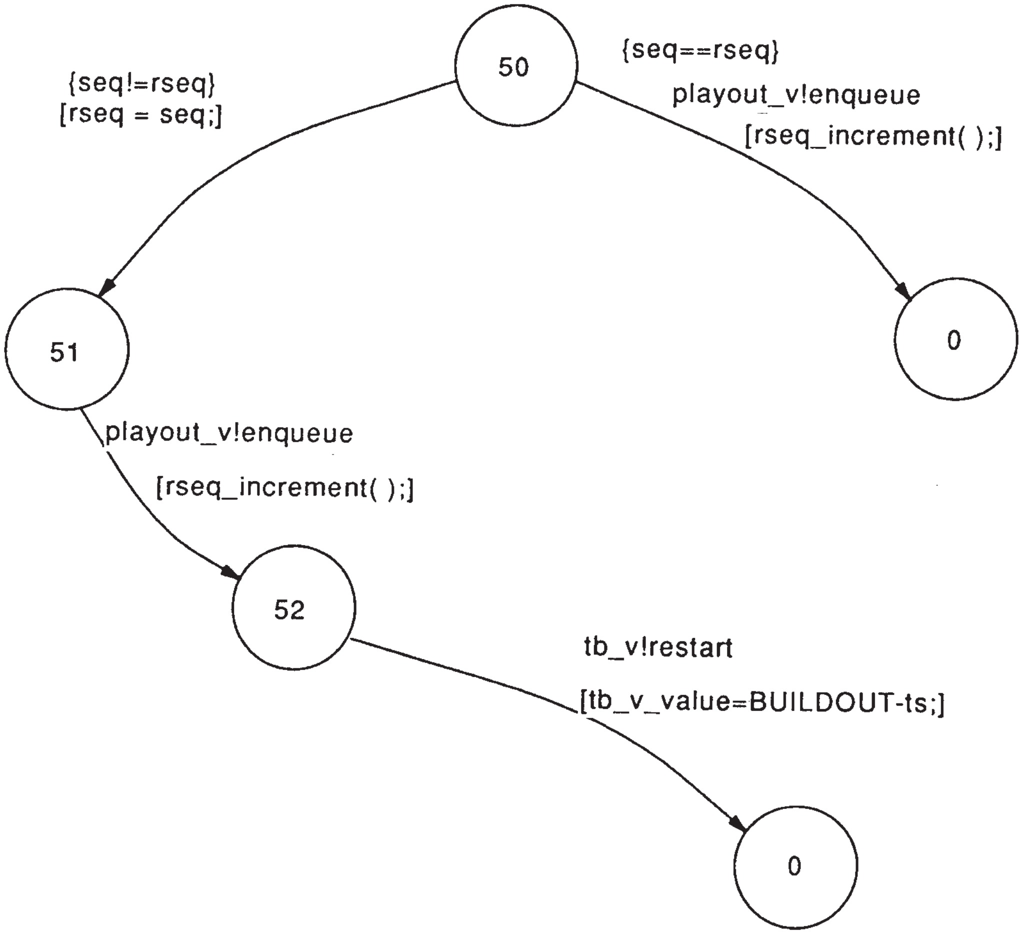
***Part a***

**Figure A.3 rcv_v**



***Part b***

**Figure A.3** *(continued)*



***Part c***

**Figure A.3** *(concluded)*

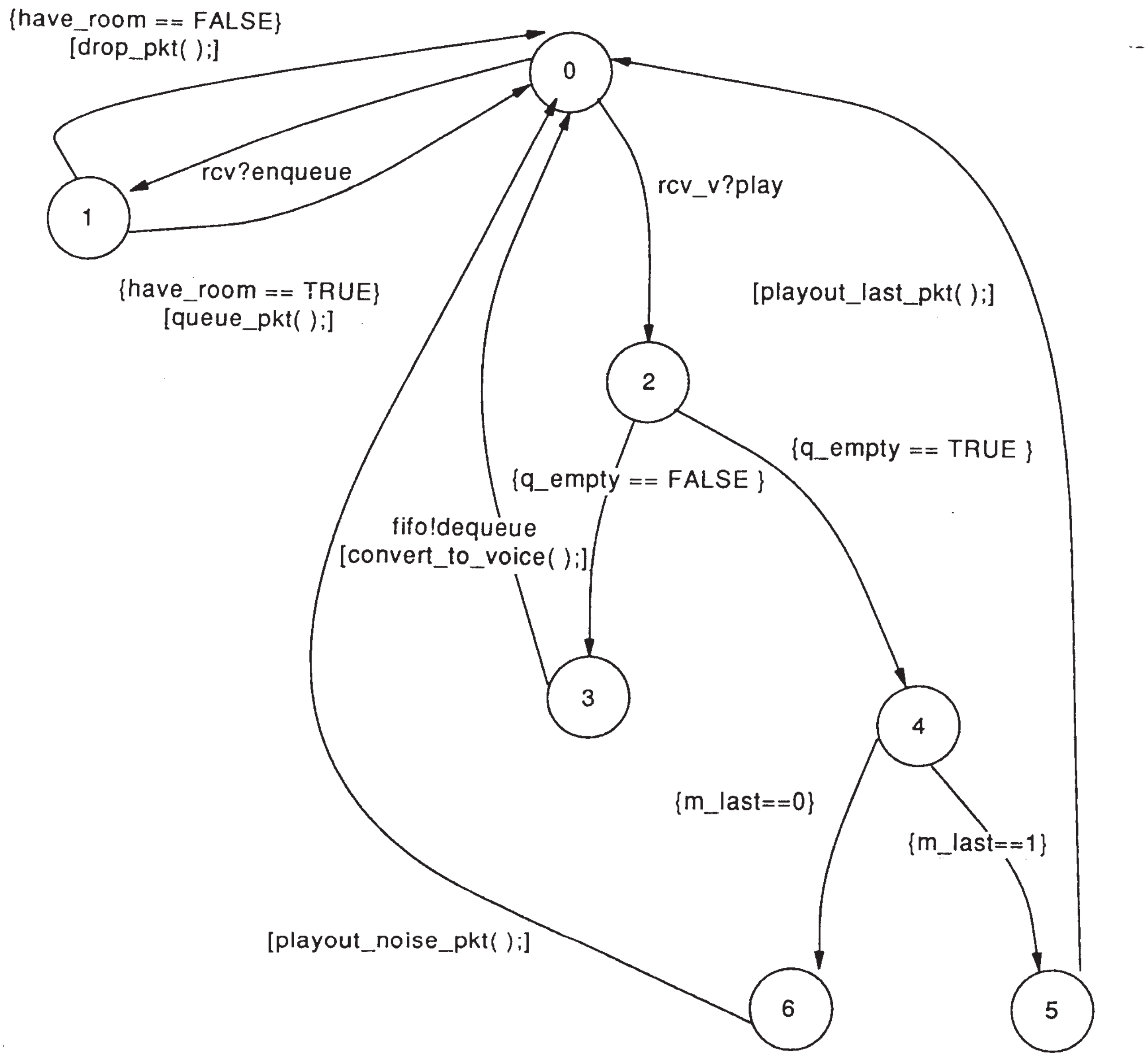


**Figure A.4 playout_v**

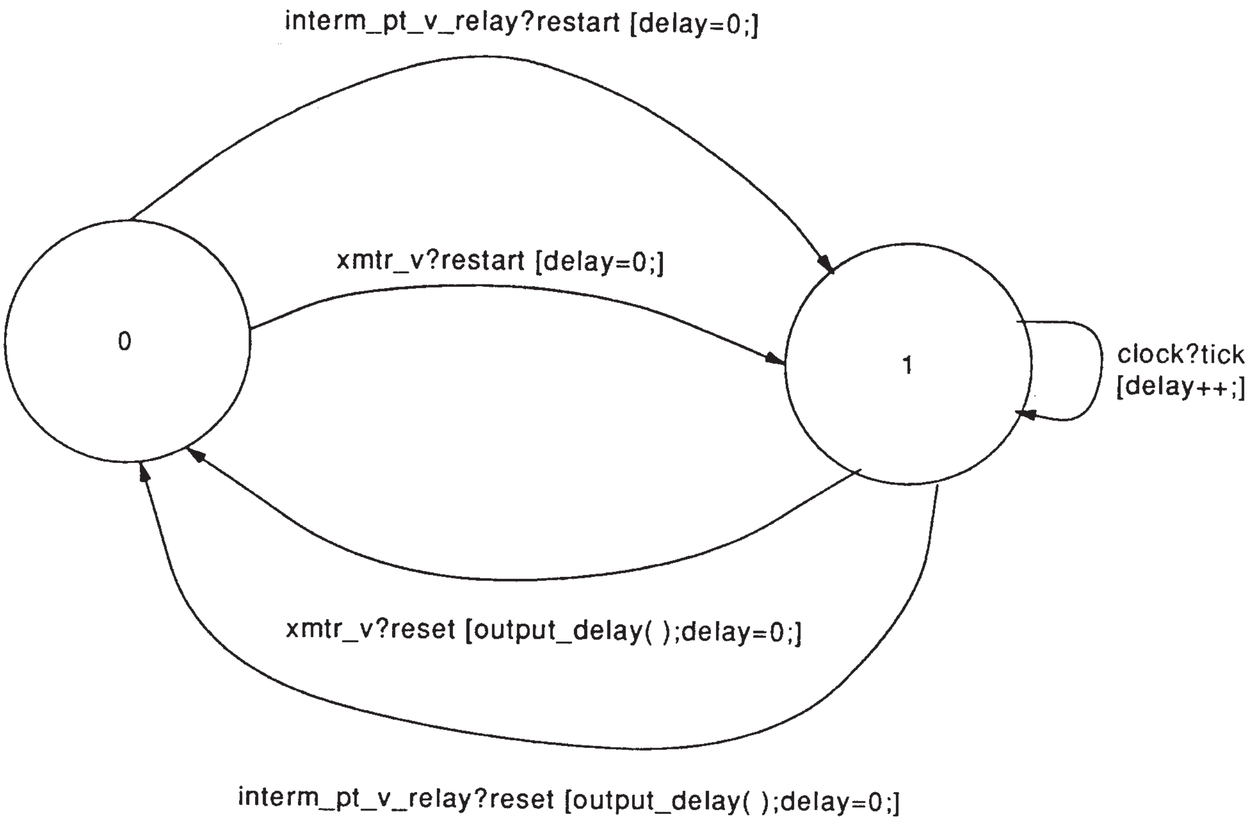

interm_pt_v_relay?restart [delay=0;]
xmtr_v?restart [delay=0;]
0
1
clock?tick
[delay++;]
xmtr_v?reset [output_delay( );delay=0;]
interm_pt_v_relay?reset [output_delay( );delay=0;]

**Figure A.5 tvdelay_v**

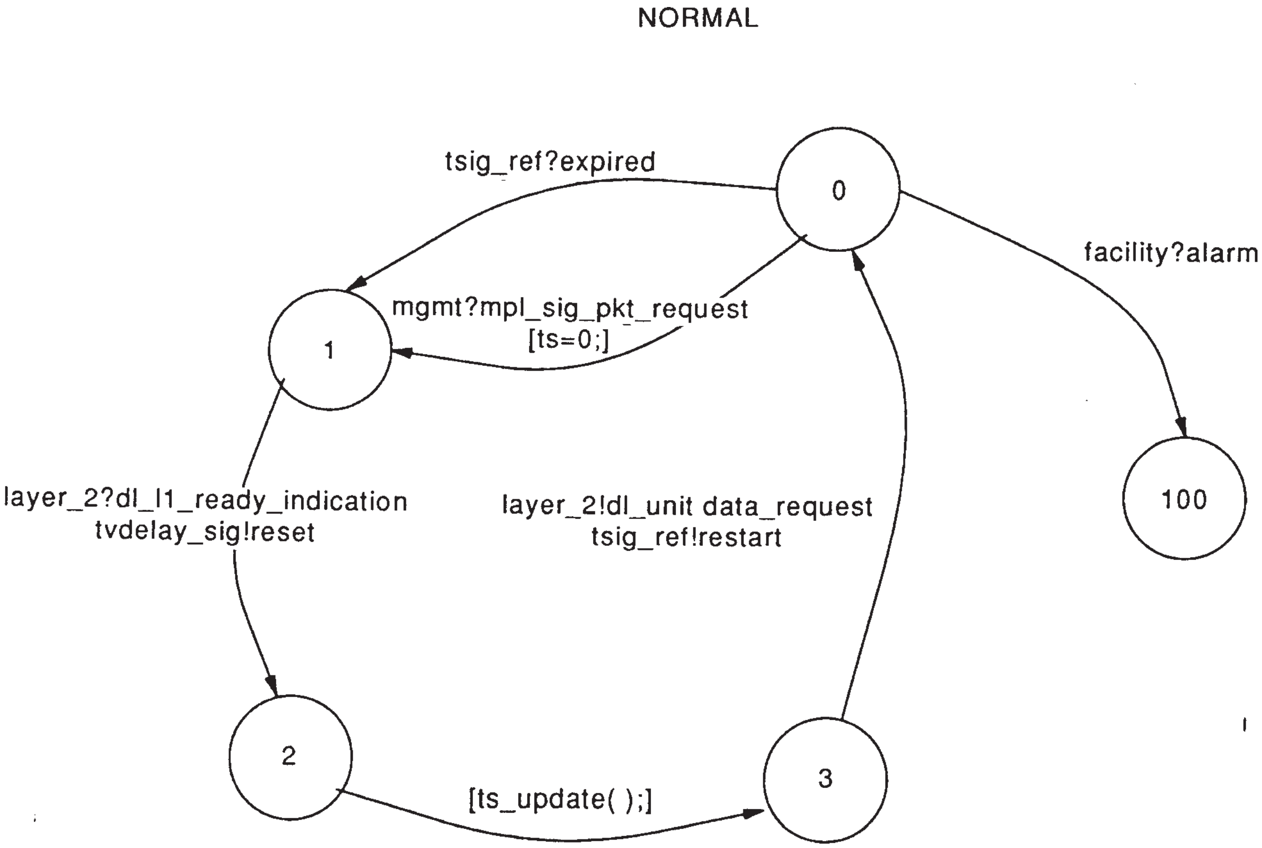


**Figure A.6 xmtr_sig**

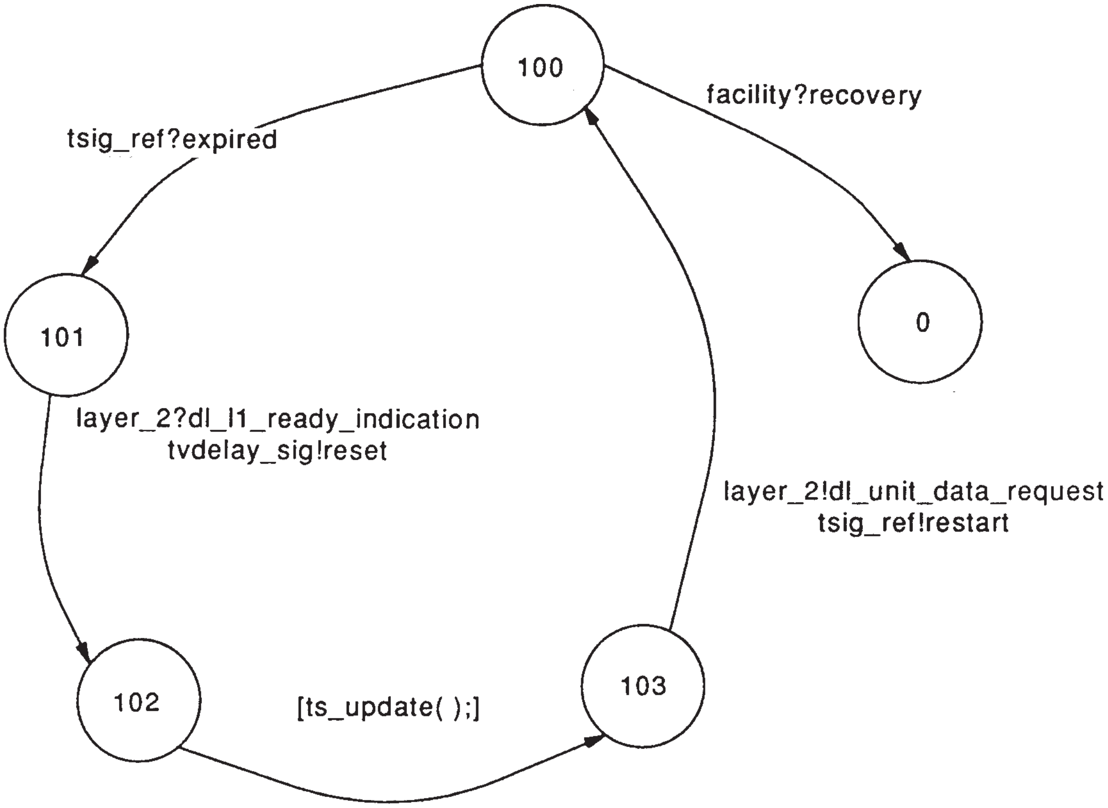


**Figure A.7 xmtr_sig**

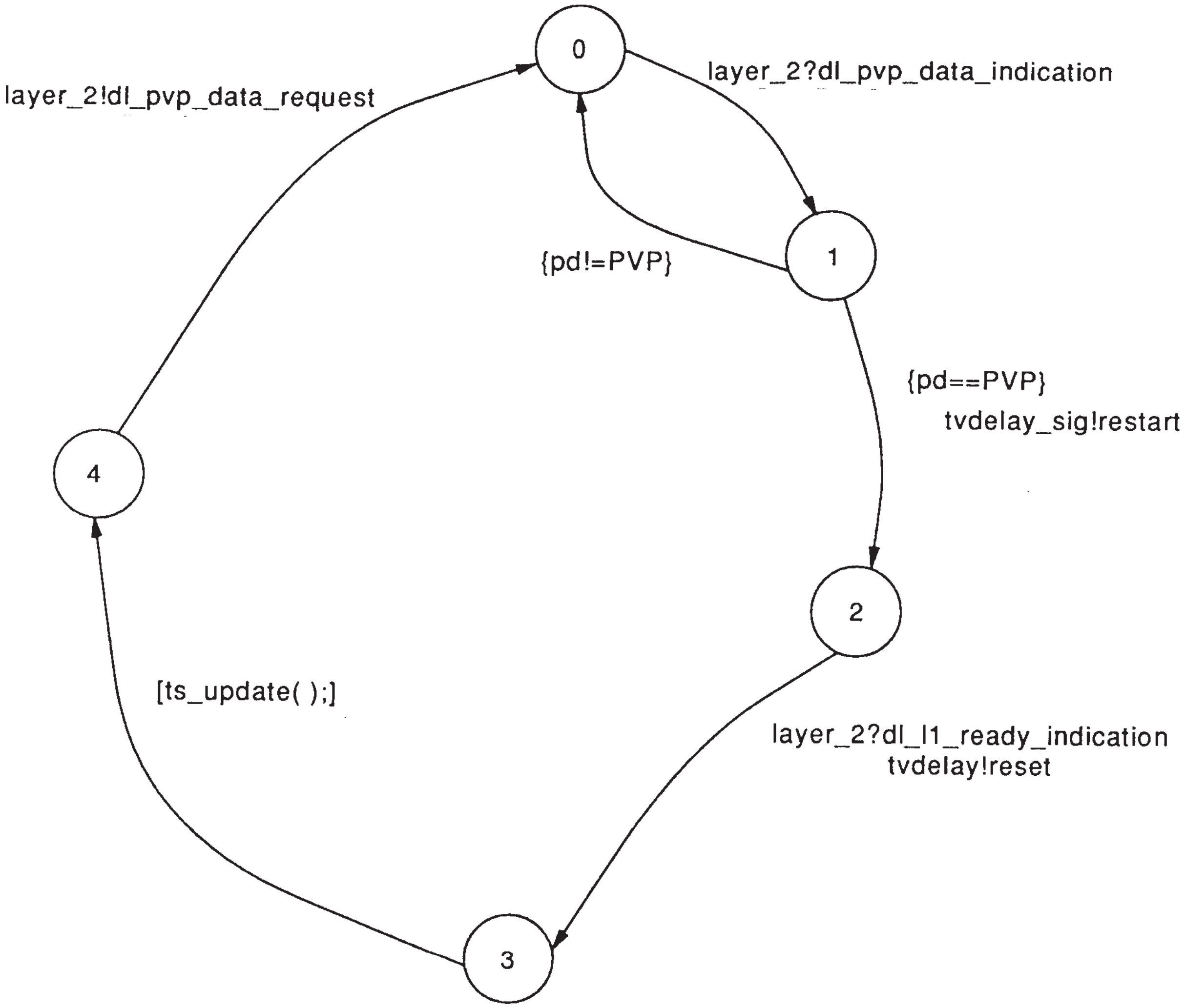

0
layer_2?dl_pvp_data_indication
layer_2!dl_pvp_data_request
1
{pd!=PVP}
{pd==PVP}
tvdelay_sig!restart
4
2
[ts_update( );]
layer_2?dl_l1_ready_indication
tvdelay!reset
3

**Figure A.8 interm_pt_sig_relay**

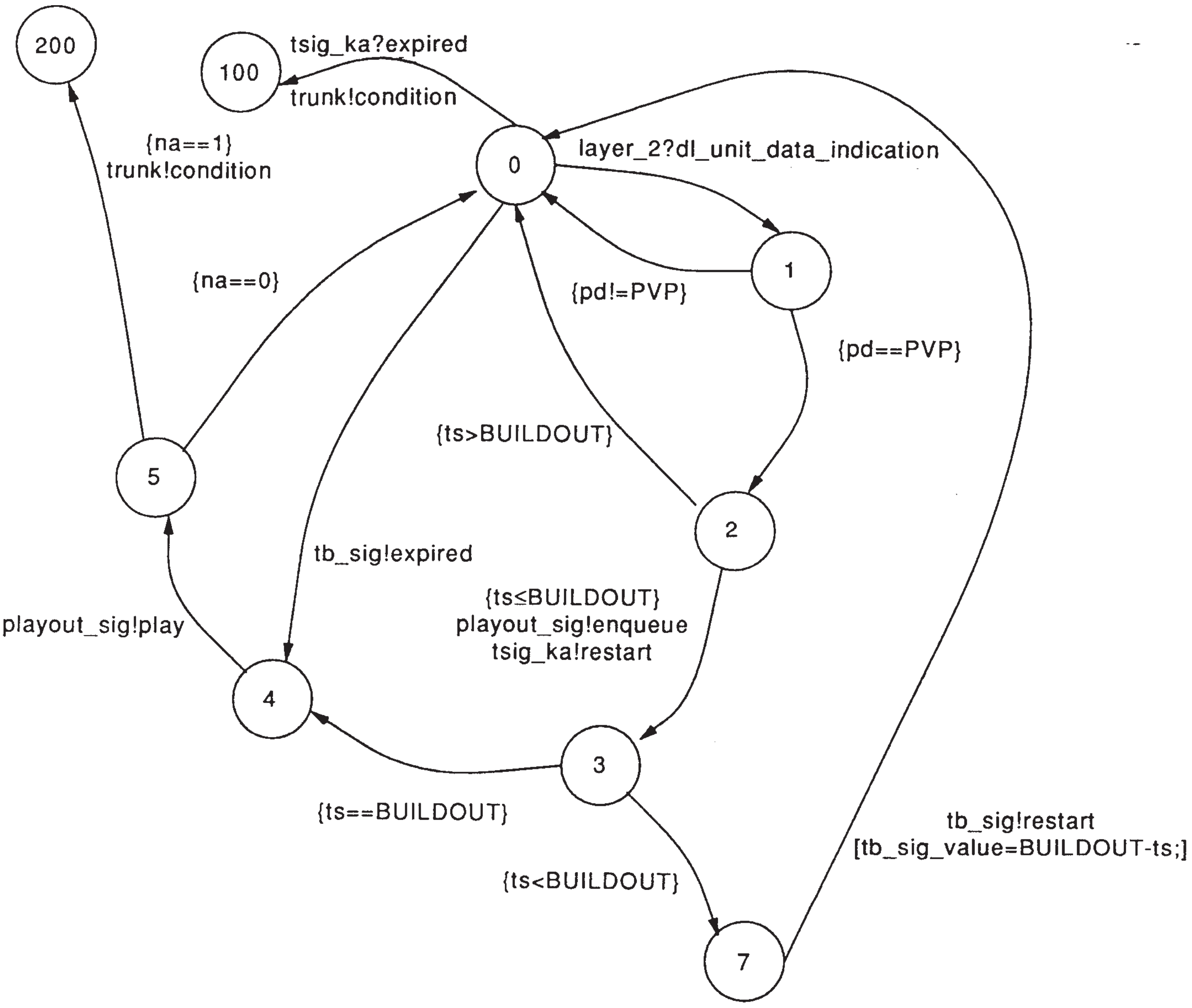


**Figure A.9 rcv_sig (norm)**

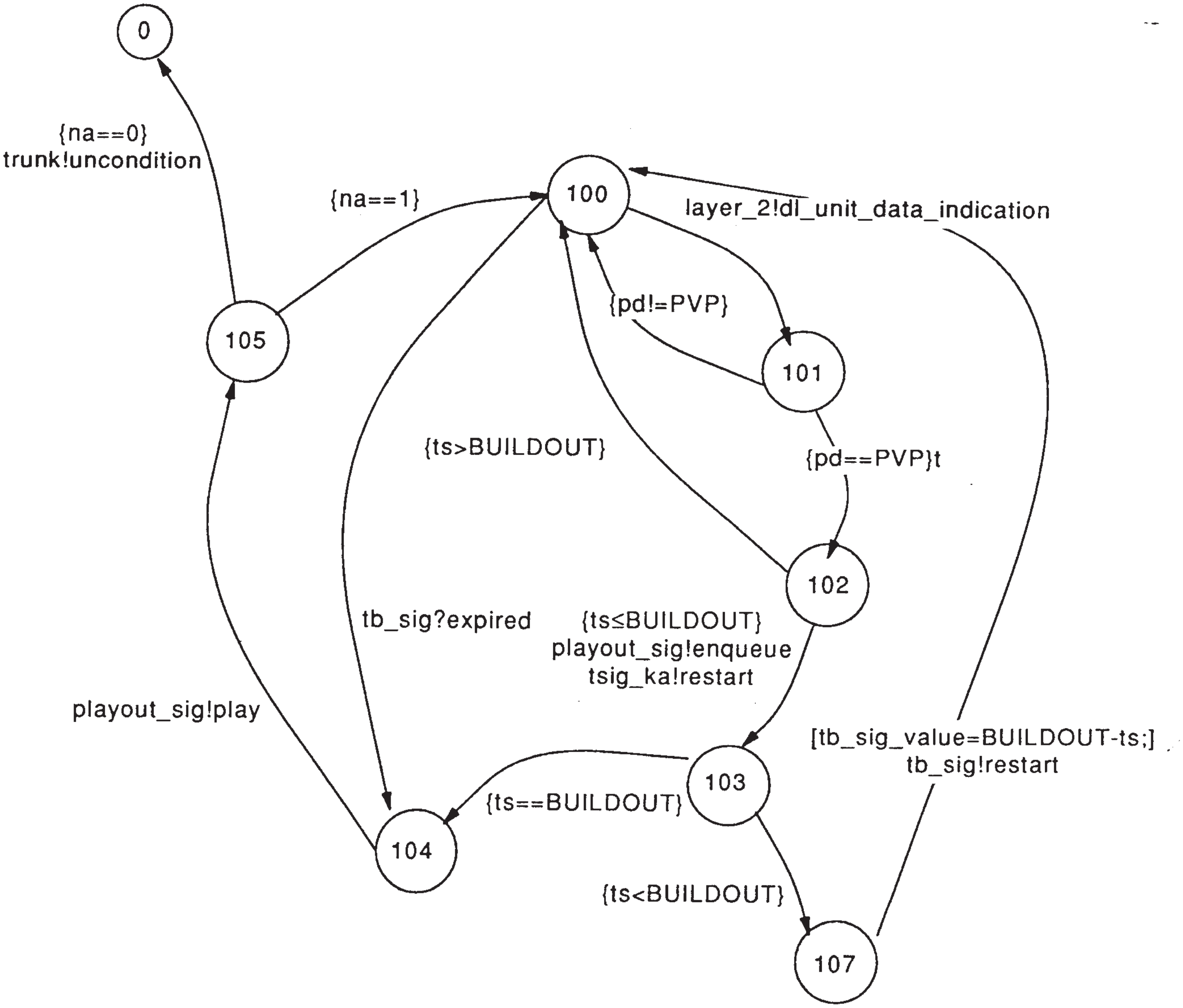


**Figure A.10 rcv_sig (l_alarm)**

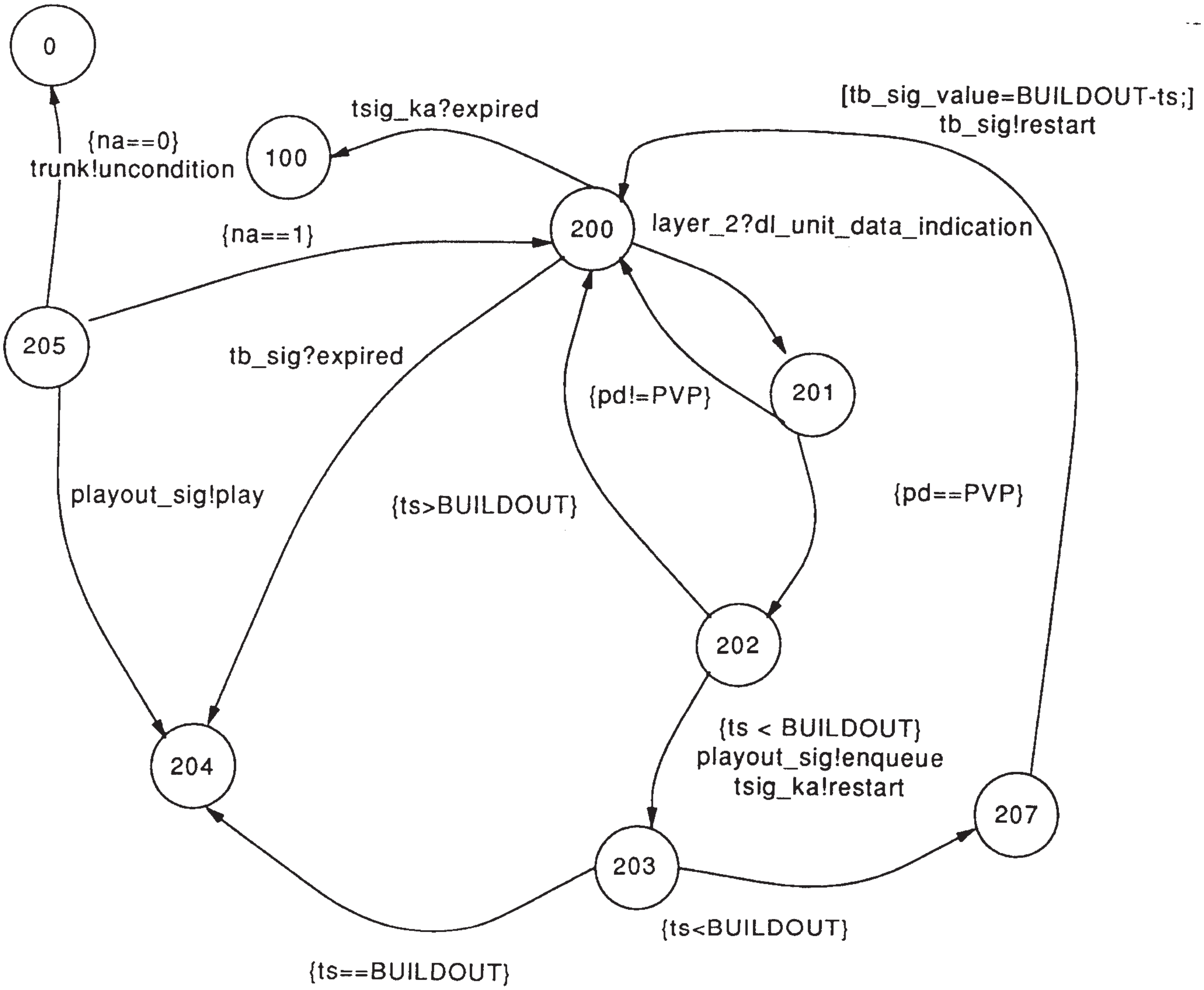


**Figure A.11 rcv_sig (r_alarm)**

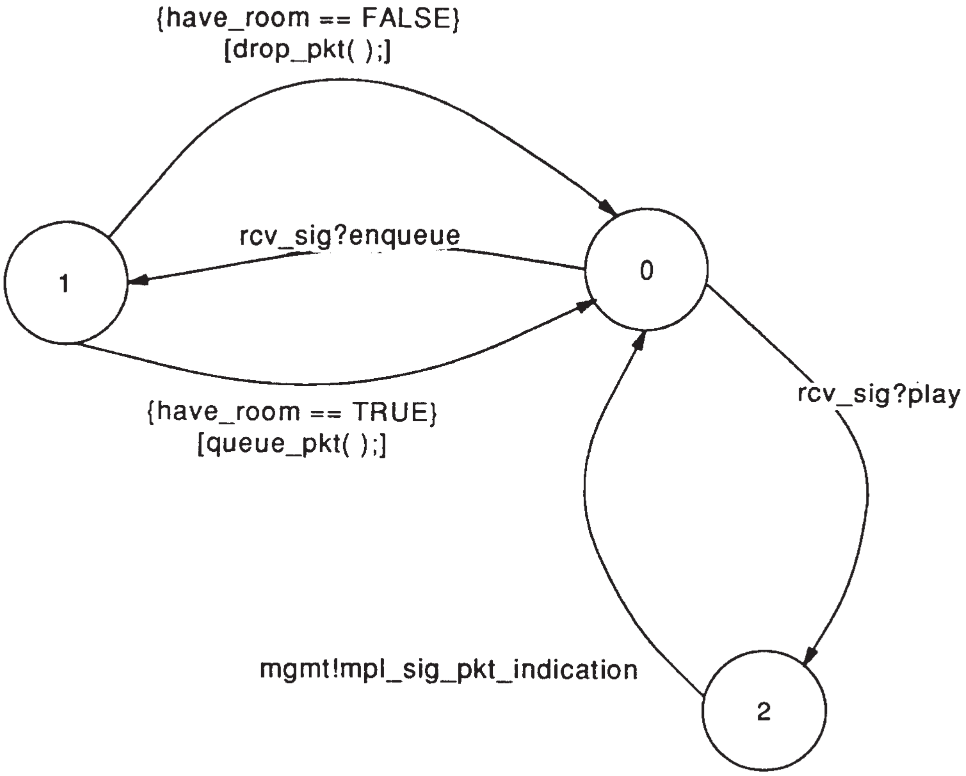


**Figure A.12 playout_sig**

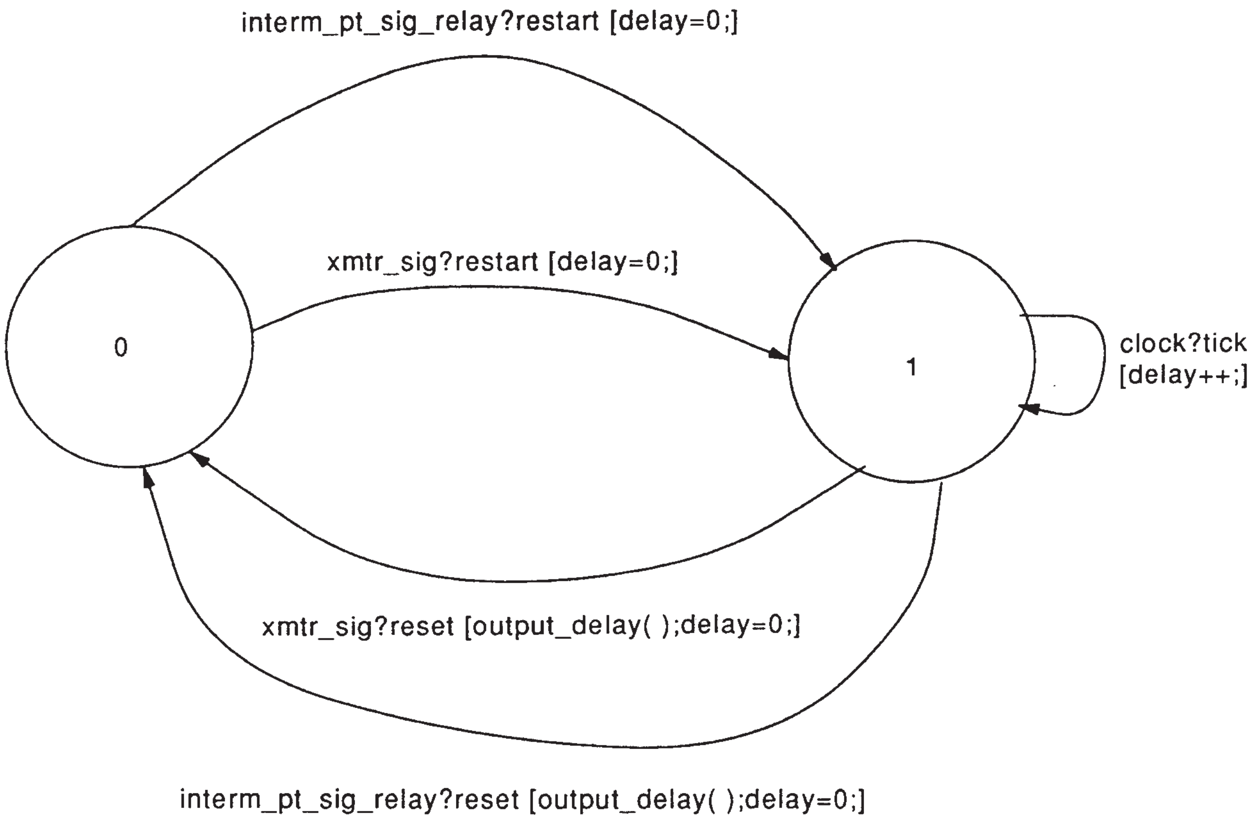

interm_pt_sig_relay?restart [delay=0;]
xmtr_sig?restart [delay=0;]
0
1
clock?tick
[delay++;]
xmtr_sig?reset [output_delay( );delay=0;]
interm_pt_sig_relay?reset [output_delay( );delay=0;]

**Figure A.13 tvdelay_sig**

# Annex B
(informative)

## Bibliography

Hoare, C. A. R. *Communicating sequential processes,* Englewood Cliffs, NJ; Prentice-Hall, 1985.

Sabnani, K. and Lapone, A. *Protocol analyzer and verifier in protocol specification*, *testing and verification VI,* B. Sarikaya and G. V. Bochmann, eds., Amsterdam; Elsevier (North-Holland), 1987, pp. 29–34.