INTERNATIONAL TELECOMMUNICATION UNION

CCITT

G.763

THE INTERNATIONAL
TELEGRAPH AND TELEPHONE
CONSULTATIVE COMMITTEE

GENERAL ASPECTS OF DIGITAL TRANSMISSION SYSTEMS; TERMINAL EQUIPMENTS

DIGITAL CIRCUIT MULTIPLICATION EQUIPMENT USING 32 kbit/s ADPCM AND DIGITAL SPEECH INTERPOLATION

Recommendation G.763

Geneva, 1991

FOREWORD

The CCITT (the International Telegraph and Telephone Consultative Committee) is the permanent organ of the International Telecommunication Union (ITU). CCITT is responsible for studying technical, operating and tariff questions and issuing Recommendations on them with a view to standardizing telecommunications on a worldwide basis.

The Plenary Assembly of CCITT which meets every four years, establishes the topics for study and approves Recommendations prepared by its Study Groups. The approval of Recommendations by the members of CCITT between Plenary Assemblies is covered by the procedure laid down in CCITT Resolution No. 2 (Melbourne, 1988).

Recommendation G.763 was prepared by Study Group XV and was approved under the Resolution No. 2 procedure on the 14 of December 1990.

CCITT NOTE

In this Recommendation, the expression "Administration" is used for conciseness to indicate both a telecommunication Administration and a recognized private operating agency.

Recommendation G.763

DIGITAL CIRCUIT MULTIPLICATION EQUIPMENT USING 32 kbit/s ADPCM AND DIGITAL SPEECH INTERPOLATION

(revised 1990)

1 General

1.1 *Scope*

This Recommendation is intended as an introduction to digital circuit multiplication equipment and systems, and as a base document for the specification of digital circuit multiplication equipment (DCME) and digital circuit multiplication systems.

DCME is utilized as a means of augmenting the capacity of digital transmission systems operating between several ISCs. DCME has all of the following attributes:

- digital speech interpolation;
- low rate encoding;
- dynamic load control arrangement in association with interfacing;
- capability to accommodate the following types of bearer service requirements:
  - i) speech,
  - ii) 3.1 kHz audio (data and speech),
  - iii) 64 kbit/s unrestricted (transparent),
  - iv) alternate speech/64 kbit/s unrestricted.

The link between two DCMEs is generally one where a highly efficient traffic carrying capability is required, e.g. a long-distance link.

Compression is accomplished by active 64 kbit/s trunk channel assignment and ADPCM encoding thereby reducing the nominal transmission channel requirements.

This Recommendation applies to digital circuit multiplication equipment telecommunications systems and specifies the following major aspects of DCME system design:

a) *network interface requirements:*
   - traffic capacities;
   - trunk and bearer facility interface;
   - signalling systems;
   - voice-band data modem support
   - echo control.

b) *functional requirements:*
   - operational modes;
   - system capacity;
   - overload strategy;
   - noise level matching;
   - PCM encoding standards conversion;
   - time slot interchange;
   - 64 kbit/s circuit handling;

– ADPCM encoders and decoders;
– timing and synchronization;
– dynamic load control;
– maintenance and alarm functions;
– facsimile compression (under study);
– tandem operation (under study).

c) *performance criteria of DCME system elements such as:*

– speech detector;
– control channel;
– voice-band data detector;
– signalling detector;
– facsimile compression (under study).

This Recommendation specifies these elements to achieve interworking.

1.2 *Purpose*

Speech signals occurring on telecommunications links are generally the product of two-way conversations. It is customary for one talker to pause while the other speaks; thus, an active speech signal is present on each direction of the trunk channel for only a fraction of the available time. In addition, even when only one talker is speaking, pauses occur between utterances, so there are times when the circuit is idle. Measurements show that speech is present on each direction of the trunk channel approximately 30 to 40% of the time, averaged over a large number of busy trunks. DCME reduces the transmission capacity needed to handle a multiplicity of telephone trunk channels by exploiting the low average channel activity and by transmitting active speech using ADPCM techniques.

The DCME provides a nominal 5 : 1 reduction in the transmission capacity required to carry various mixtures of speech, voice-band data and 64 kbit/s unrestricted channels. An overload strategy consisting of variable bit rate encoding and dynamic load control techniques is utilized to limit speech clipping. The DCME data detector discriminates between voice-band data and speech in order to assign the voice-band data signal to a bearer channel protected against the formation of overload channels which degrade the voice-band data performance.

1.3 *Application*

This Recommendation is applicable to the design of digital circuit multiplication equipment intended for, but not limited to, use in an international digital circuit. Freedom is permitted in design details which are not covered in this Recommendation.

2 Definitions relating to digital circuit multiplication equipment (DCME)

2.1 **digital circuit multiplication equipment (DCME)**

A general class of equipment which permits concentration of a number of 64 kbit/s PCM encoded input trunk channels on a reduced number of transmission channels (see § 2.7).

2.2 **digital circuit multiplication system (DCMS)**

A telecommunications network comprised of two or more DCME terminals where each DCME terminal contains a transmit unit and a receive unit.

2.3 **low rate encoding (LRE)**

A voice-band signal encoding method, e.g. ADPCM, which results in a bit rate less than 64 kbit/s, e.g. either 40 kbit/s, 32 kbit/s, 24 kbit/s, or optionally 16 kbit/s. Conversion between speech signals encoded in PCM at 64 kbit/s and those encoded in ADPCM must be carried out by means of transcoding processes given in Recommendation G.726.

2.4 **variable bit rate (VBR)**

The capability of the encoding algorithm to dynamically switch between either 32 and 24 kbit/s or also optionally between 24 and 16 kbit/s for speech traffic under control of the DCME.

2.5 **digital speech interpolation (DSI)**

A process which, when used in the transmit unit of a DCME, causes a trunk channel (see § 2.9) to be connected to a bearer channel (see § 2.8) only when activity is actually present on the trunk channel. This, by exploiting the probability of the speech activity factor (see § 2.15) of trunk channels being less than 1.0, enables the traffic from a number of trunk channels to be concentrated and carried by a lesser number of time shared bearer channels. The signals carried by a bearer channel therefore represent interleaved bursts of speech signals derived from a number of different trunk channels.

Note – A process complementary to DSI is required in the receive unit of a DCME, i.e. assignment of the interleaved bursts to their appropriate trunk channels.

2.6 **DCME frame**

A time interval, the beginning of which is identified by a unique word in the control channel. The DCME frame need not coincide with the multi-frames defined in Recommendation G.704. The format specification of the DCME frame includes channel boundaries and bit position significance.

2.7 **transmission channel**

A 64 kbit/s time slot within a DCME frame.

2.8 **bearer channel (BC)**

A bearer channel is a unidirectional, digital, transmission path from the transmit unit of one DCME to the receive unit of a second associated DCME used to carry concentrated traffic between the two DCMEs.

Note 1 – A number of bearer channels in each direction of transmission form the both-way link required between two DCMEs. This link may be, for example, a 2048 kbit/s system.

Note 2 – A bearer channel may have any of the following instantaneous bit rates: either 64, 40, 32, 24, or optionally 16 kbit/s.

2.9 **trunk channel (TC)**

A unidirectional, digital transmission path (generally short distance) used for carrying traffic and which connects a DCME to other equipment, e.g. an ISC. Two such trunk channels (transmit and receive) are needed by 4 wire telephone circuits and constitute a trunk circuit.

Note 1 – Signals carried by a trunk channel will be transmitted at a bit rate of 64 kbit/s.

Note 2 – A number of trunk channels in each direction of transmission are required between a DCME and, for instance, an ISC. These trunk channels may be carried by a number of 2048 or 1544 kbit/s systems.

2.10 **intermediate trunk (IT)**

A channel mapping designation which ranges between 1 and 216 which relates each trunk channel to an internal numbering designation used within the DCME for conveying trunk channel to bearer channel connectivity via the control channel (see § 2.13).

2.11 **assignment message**

The message specifying the interconnections required between trunk channels and bearer channels.

2.12 **assignment map**

A record, held in a memory of a DCME, of the interconnections required between trunk channels and bearer channels. This record is dynamically updated in real time in accordance with the traffic demands made on the DCME.

2.13 **control channel (CC)**

A unidirectional transmission path from the transmit unit of one DCME to the receive unit of one or more associated DCMEs which is dedicated primarily to carrying channel assignment messages. In addition, the control channel transmits other messages such as idle noise levels, dynamic load control, alarm messages and optionally line signalling information.

Note – An alternative name for control channel is assignment channel.

2.14 **ensemble activity**

The ratio of the time active signals and their corresponding hangover time and front end delay occupy the trunk channels to the total measuring time, averaged over the total number of trunk channels included in the measurement.

2.15 **speech activity factor**

The ratio of the time active speech signals with their corresponding hangover time and front end delay occupy a trunk channel to the total measuring time, averaged over the total number of trunk channels carrying speech signals.

2.16 **voice-band data ratio**

The ratio of the number of trunk channels carrying voice-band data signals to the total number of trunk channels averaged over a fixed interval of time.

2.17 **64 kbit/s unrestricted digital data ratio**

The ratio of the number of trunk channels carrying 64 kbit/s unrestricted digital data signals to the total number of trunk channels averaged over a fixed interval of time.

2.18 **DCME overload (mode)**

The condition when the number of input trunk channels instantaneously active carrying speech exceeds the number of 32 kbit/s channels available for interpolation.

2.19 **overload channels**

The additional bearer channel capacity which is generated using VBR encoding to minimize or eliminate DSI competitive clipping.

2.20 **average bits per sample**

The average number of encoding bits per sample computed over a given time window for the ensemble of active interpolated bearer channels within a given interpolation pool. Only bearer channels carrying speech are included in this calculation.

2.21 **transmission overload**

The condition when the average bits per sample goes below the value set in accordance with speech quality requirements.

2.22 **freeze-out**

The condition when a trunk channel becomes active and cannot immediately be assigned to a bearer channel, due to lack of available transmission capacity.

2.23 **freeze-out fraction (FOF)**

The ratio of the total time that the individual channels experience the freeze-out condition to the total time of the active intervals and their corresponding hangover times and front end delays, for all trunks over a fixed interval of time.

2.24 **interpolation gain (IG)**

The trunk channel multiplication ratio which is achieved through DSI. The IG is the ratio of the number of trunk channels to the number of DCME bearer channels where the same signal encoding rate is used for trunk and bearer channels. The achievable gain depends on the ensemble activity and the system size.

2.25 **transcoding gain (TG)**

The transmission channel multiplication ratio which is achieved through LRE, which effectively creates a number of low rate encoded bearer channels which is greater than the number of available transmission channels. When only a transcoding process conforming to the 32 kbit/s portion of Recommendation G.726 is used, the TG will equal 2. When no transcoding is used the TG will equal 1. When overload channels are created the TG will be greater than 2.

2.26 **DCME gain (DCMG)**

The trunk channel transmission multiplication ratio, which is achieved through application of DCME, including LRE and DSI. Hence DCMG = TG · IG.

2.27 **clique**

A set of bearer channels which are associated with a set of trunk channels which are independent in operation and control from other bearer channels. The set of trunk channels is directed to a single destination.

Note – An alternative term for clique is bundle.

2.28 **multi-clique mode**

A DCME operational mode in which more than one clique is used when each clique is associated with a different destination.

2.29 **multi-destination mode**

A DCME operational mode where traffic is exchanged between more than two (2) corresponding DCMEs simultaneously and trunk channel traffic is interpolated over a pool of available bearer channels for all destinations having traffic in the pool. The transmit trunk channels are designated to receive trunk channels at corresponding locations.

2.30 **silence elimination**

When voice-band data traffic is recognized on a trunk channel, the DCME sets a long hangover time to ensure that no clipping will occur in case of half-duplex transmission.

In many cases (e.g. facsimile group 3 transmission), the backward direction is mainly used for the transmission of acknowledgements, and the return trunk channel has therefore a very low rate of activity. If the long hangover time was still in operation, there would be a significant waste of bearer capacity.

The use of a second hangover time, shorter than the initial one, will allow making the bearer capacity on the backward direction available to the interpolation pool, and is called silence elimination.

3 DCME functions

3.1 *General*

This Recommendation defines DCME which provides circuit multiplication by means of ADPCM and DSI.

For operation between Administrations using 2048 kbit/s interfaces, the channel side (bearer) interface to/from the DCME shall be based on the 2048 kbit/s interface.

For operation between Administrations using 2048 kbit/s interfaces and Administrations using 1544 kbit/s interfaces, the channel side (bearer) interface to/from the DCME will be based on the 2048 kbit/s interface.

For operation between Administrations using 1544 kbit/s interfaces, the channel side (bearer) interface to/from the DCME may be based on either the 1544 kbit/s or the 2048 kbit/s interface dependent on bilateral agreement.

There may be operational difficulties with ISC/DCME interworking depending on whether the DCME is type 1, where the DCME cannot communicate with the ISC, or type 2, where it can, as defined in Recommendation Q.50.

3.2 *Purpose*

The purpose of DCME is to provide maximum effective use of transmission facilities in the digital operating environment, using DSI and LRE techniques. At a minimum, the DCME functions shall include:

– interpolation of speech signals (DSI);

– transcoding of 64 kbit/s PCM to ADPCM when applicable;

– the means to carry the ISDN bearer services given in § 1.1:

  i) speech,

  ii) 3.1 kHz audio (data and speech),

  iii) 64 kbit/s unrestricted;

- one or more of the following operating modes:
  - i) point-to-point,
  - ii) multi-clique,
  - iii) multi-destination;
- speech detection;
- voice-band data detection;
- facsimile compression (under study);
- a means for transmit detection and receive injection of background noise;
- the means to accommodate non-interpolated pre-assigned traffic;
- a means for interterminal communication (control channel);
- a means for exchanging signals with an ISC for purposes of ISDN bearer services involving 64 kbit/s unrestricted traffic, DLC, and alarms;
- time slot interchange;
- the ability to transport the following signalling systems:
  - i) CCITT No. 5,
  - ii) CCITT No. 6 (both analogue and digital versions),
  - iii) CCITT No. 7,
  - iv) R1 (Note 1 of § 3.2),
  - v) R2 (Note 1 of § 3.2).

Note 1 of § 3.2 – CCITT Signalling Systems R1 and R2 may be transported, but each will require its own special interface. It is recommended that the transmission of line signals is performed using special messages in the control channel.

The DCME will perform processing on traffic between the trunk interface and bearer interface as defined in Table 1/G.763 and explained below:

a) Speech traffic is ADPCM encoded and subject to DSI. The bit rate of individual bearer channels provided for speech is instantaneously either 32, 24, or optionally 16 kbit/s dependent on traffic loading. If the optional 16 kbit/s overload feature is activated and being used, the bit rate of the bearer channels provided for speech is 24 kbit/s or 16 kbit/s dependent on traffic loading.

b) Voice-band data traffic is initially subject to DSI. Bearer channels provided for traffic recognized as voice-band data are ADPCM encoded at 40 kbit/s and are protected against bit reduction and clipping.

c) 64 kbit/s unrestricted traffic may be connected on demand to bearer channels transparently (not subjected to DSI and ADPCM) if an out-of-band control system to/from the ISC is provided to identify the relevant trunk channel.

d) Alternate speech/64 kbit/s unrestricted traffic may be accommodated subject to the provision of an out-of-band control facility and in-call modification signals from the ISC.

e) 64, 40 and 32 kbit/s channels may be pre-assigned for leased line services which are not to be subjected to DSI. Optionally 24 kbit/s or 16 kbit/s pre-assigned channels may be used for maintenance purposes only.

f) CCITT Signalling System No. 5 will be passed transparently through the DCME. CCITT Signalling Systems Nos. 6 and 7 can be accommodated through 64 kbit/s pre-assigned channels.

g) If provided with optional user signalling modules (USM), the DCME will convey line signalling information within the control channel. Two USM modules are presently considered, R1 USM and R2 USM (see Note 1 of § 3.2 above). Requirements have been defined for an R2 USM.

TABLE 1/G.763

DCME traffic handling

Bearer service	Dynamic assignment	Pre-assignment
Speech	32 kbit/s ADPCM with DSI 24 kbit/s ADPCM with DSI optionally 16 kbit/s ADPCM with DSI	32 kbit/s ADPCM (Note 4)
3.1 kHz Audio (Voice-band data (Note 1)	40 kbit/s ADPCM	40 kbit/s ADPCM or 32 kbit/s ADPCM
64 kbit/s unrestricted	64 kbit/s on-demand (Note 2)	64 kbit/s Pre-assigned
Alternate speech 64 kbit/s	64 kbit/s on-demand and 32/34 kbit/s (and optionally 16 kbit/s) ADPCM with DSI (Note 3)	64 kbit/s Pre-assigned

Note 1 – 40 kbit/s ADPCM will accommodate voice-band data at rates ≤ 9.6 kbit/s. 32 kbit/s ADPCM will accommodate voice-band data at rates ≤ 4.8 kbit/s.

Note 2 – Subject to the provision of a dedicated control system to/from the ISC.

Note 3 – Subject to the provisions for in-call modification from the ISC.

Note 4 – 24 or 16 kbit/s ADPCM pre-assignment may be used for maintenance purposes only.

Note 5 – Special arrangements for prevention of quantization distortion unit (QDU) accumulation when DCMEs are operated in tandem are under study.

The actual circuit multiplication gain achieved will be dependent upon traffic loading, speech activity, the percentage and type of voice-band data (e.g. facsimile traffic), the number of on-demand unrestricted 64 kbit/s channels, the number of pre-assigned channels, and the size of the interpolation pools.

The total delay associated with the establishment of dynamically assigned ADPCM encoded bearer channels by the transmit DCME shall not exceed 30 ms. The total delay associated with the establishment of dynamically assigned ADPCM encoded bearer channels by the receive DCME shall not exceed 15 ms. The delay values exclude the effects of Doppler and plesiochronous buffers and exclude the delays associated with the establishment and disestablishment of demand assigned 64 kbit/s unrestricted circuits.

4 Operational modes

4.1 *General*

The following modes of operation are described:

a) point-to-point;

b) multi-clique;

c) multi-destination; and

d) interoperation.

The DCME multiple destination capability for multi-clique and multi-destination modes is summarized in Table 2/G.763.

TABLE 2/G.763

**DCME multiple destination capability
for multi-clique and multi-destination modes**

a) Transmit

	Total No. destinations	No. of pools in bearer	No. of destinations in pool
Multi-clique	1 2	1 2	1 1, 1
Multi-destination	4 max.	1 2	1 to 4 1 to 3, 1

b) Receive

	Total No. of origins	No. of received bearers	No. of pools in each bearer
Multi-clique	2 max.	1	1 or 2
Multi-destination	4 max.	4 max.	1

4.1.1 *Point-to-point mode*

See Figure 1/G.763.

4.1.1.1 *Point-to-point*

Using Figure 1/G.763 for reference, the transmit side DCME concentrates N trunk channels at 64 kbit/s into N/G transmission channels. The transmission channels represent a number of time shared variable bit rate (bearer) channels which are grouped into a primary rate multiplex format.

At the receive side, the receiving DCME simply demultiplexes the primary-rate format and reconstitutes the N trunk channels from the N/G transmission channels.

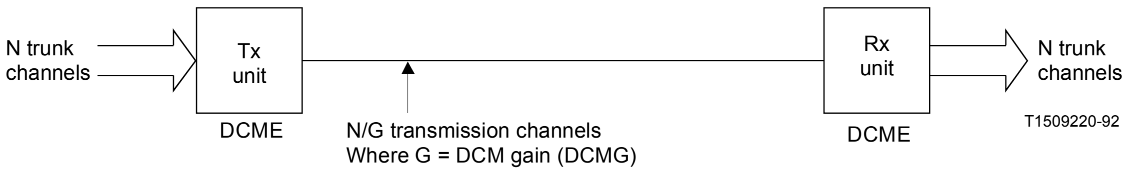


FIGURE 1/G.763

Point-to-point mode
(only one direction shown)

4.1.2 *Multi-clique mode*

See Figure 2/G.763.

4.1.2.1 *Multi-clique mode*

In this mode the pool of bearer channels is subdivided into two independent pools (cliques) of fixed capacity, each being for an individual destination. While the aggregate bearer bit rates for both the transmit side and the receive side are equal, the DCMG of each clique may be different since this gain is a function of the number of input channels to be routed within each clique. It is desirable to limit the number of cliques within a primary rate bearer to two. Figure 2/G.763 indicates one form of this approach in which the primary rate bearer circuit is assumed to be available to each of the DCM nodes, but each node has the capability of preselecting the traffic that is intended for it. The multi-clique mode may be useful to prevent qdu accumulation when DCME terminals are operated in tandem. This subject is under study.

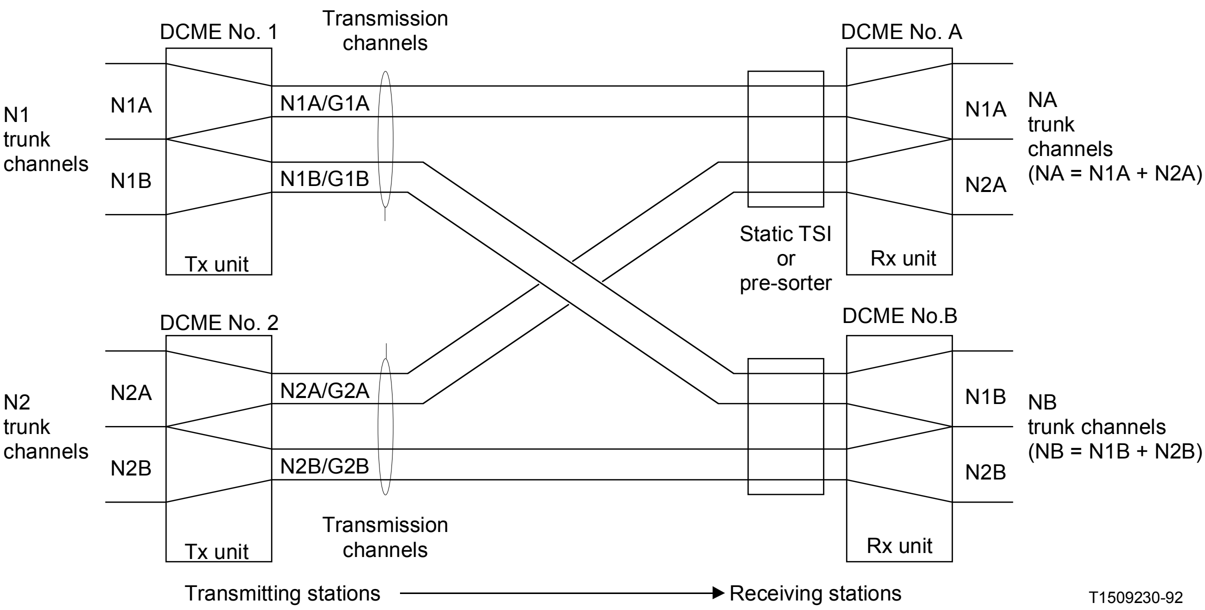


FIGURE 2/G.763

Multi-clique mode
(only one direction shown)

4.1.3 *Multi-destination mode*

See Figure 3/G.763.

4.1.3.1 *Multi-destination mode*

In this mode, the input trunk channels are interpolated over a common pool of bearer channels, regardless of their destination. The input trunk channels are destination pre-assigned so that they may be routed to the appropriate destination in accordance with the control channel messages. This operational mode permits higher DCMG than the multi-clique mode but its usefulness is limited if the DCME is located at the ISC.

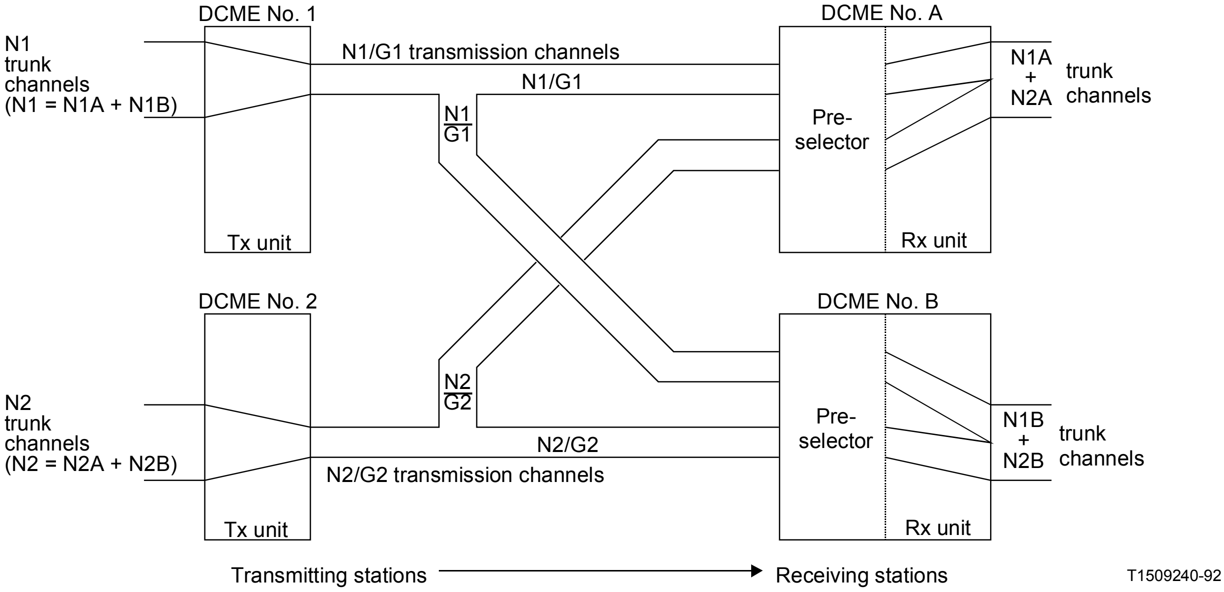


FIGURE 3/G.763

Multi-destination mode
(only one direction shown)

4.1.4 *Interoperation*

DCME with the multi-destination option shall interwork with DCME of the point-to-point option when the multi-destination DCME is configured with a single destination pool. The single destination pool shall be used for interworking.

DCME with the multi-destination option shall interwork with DCME of the multi-clique option when the multi-destination DCME includes a single destination pool. The single destination pool shall be used for interworking.

4.2 *Modes of assignment of channels to the bearer structure*

4.2.1 *Pre-assignment*

It shall be possible to pre-assign 64 kbit/s trunk channels to 64 kbit/s bearer channels in the pool (8 bits within the bearer frame). The number of pre-assigned 64 kbit/s trunk channels shall be preset under operator control from zero to the maximum number of complete 8 bit time slots within the pool in increments of one 64 kbit/s channel.

It shall be possible to pre-assign 64 kbit/s trunk channels to 40 kbit/s bearer channels in the pool (5 bits within the bearer frame structure). The number of pre-assigned 40 kbit/s bearer channels shall be preset under operator control from zero to the maximum determined by the pool size in increments of one 40 kbit/s channel.

It shall be possible to pre-assign 64 kbit/s trunk channels to 32 kbit/s bearer channels in the pool (4 bits within the bearer frame). The number of pre-assigned 32 kbit/s bearer channels shall be preset under operator control from zero to a maximum determined by the pool size in increments of one 32 kbit/s channel.

As an option it shall be possible to pre-assign 64 kbit/s trunk channels to 24 kbit/s or 16 kbit/s bearer channels in the pool. Each 24 kbit/s or 16 kbit/s bearer channel will occupy the most significant three bits or two bits respectively of a 32 kbit/s pre-assigned bearer channel and will be used for maintenance purposes only. The number of pre-assigned 24 kbit/s or 16 kbit/s bearer channels shall be preset under operator control from zero to a maximum determined by the pool size in increments of one 32 kbit/s bearer channel.

4.2.2 *Dynamic assignment*

The DCME shall be capable of assigning on-demand 64 kbit/s unrestricted traffic to 64 kbit/s bearer channels in the pool (8 bits within each bearer frame) using an out-of-band control facility between the ISC and the DCME for seizure/release of 64 kbit/s bearer channels as defined in § 5. The provision of the ISC control facility is at the users' option. The transmit and receive assignment processes are described in §§ 6, 7 and 8.

The DCME shall be capable of dynamically assigning voice traffic within 64 kbit/s trunk channels to bearer channels at bit rates of 32, 24, and optionally 16 kbit/s in each pool (4 bits, 3 bits or 2 bits within each bearer frame). The transit and receive assignment processes are described in §§ 6 and 7.

The DCME shall be capable of dynamically assigning voice-band data traffic within a 64 kbit/s trunk channel to 40 kbit/s bearer channels (5 bits within each bearer frame). The transmit and receive assignment processes are described in §§ 6 and 7.

5 Interface requirements

The DCME shall be interconnected with a local or remote ISC(s) by means of the trunk interface equipment and a DCME ISC signalling system. There is a maximum channel capacity of 216 trunk channels as a consequence of fundamental limitations in the assignment message numbering scheme. Therefore, the trunk interface equipment shall be capable of accommodating seven 2048 kbit/s primary multiplex streams or nine 1544 kbit/s primary multiplex streams.

Trunk channels (TCs) are related one-to-one to the intermediate trunks (ITs) by a TC to IT mapping facility within the DCME to permit configuration control of the trunk time slots and to adopt a channel numbering convention for DCME-to-DCME operation.

Local ITs are used by the transmit DCME and are identified within the DCME-to-DCME control channel messages. Remote ITs are received in the control channel messages from corresponding DCMEs.

In the case of interworking between the 1544 kbit/s and 2048 kbit/s hierarchies on the same DCME, it is recommended in Recommendation G.802 that the bearer system be 2048 kbit/s.

There may be operational difficulties with ISC/DCME interworking depending on whether the DCME is type 1, where the DCME cannot communicate with the ISC, or type 2, where it can, as defined in Recommendation Q.50.

5.1 *Transmission interface: trunk side*

5.1.1 *Trunk side interface at 2048 kbit/s*

a) The electrical characteristics shall comply with Recommendation G.703. The test load impedance shall be either 75 Ω unbalanced or 120 Ω balanced depending on the user requirement.

b) The frame structure shall comply with Recommendation G.704.

c) The encoding law for voice frequency signals shall conform to the A-law system described in Recommendation G.711.

5.1.2 *Trunk side interface at 1544 kbit/s*

a) The electrical characteristics shall comply with Recommendation G.703. The line code adopted shall be either AMI or B8ZS depending on the user selection.

b) The frame structure shall comply with Recommendation G.704. The multi-frame size shall be either 24 frames or 12 frames depending on the user selection.

c) The encoding law for voice frequency signals shall conform to the μ-law system described in Recommendation G.711.

5.2 *Transmission interface: bearer side*

5.2.1 *Bearer side interface at 2048 kbit/s*

5.2.1.1 *General*

For the point-to-point and multi-clique modes, the bearer interface shall consist of one 2048 kbit/s interface on the transmit side and one 2048 kbit/s interface on the receive side.

For the multi-destination mode, the bearer interface shall consist of one 2048 kbit/s interface on the transmit side and one to four 2048 kbit/s interfaces on the receive side.

5.2.1.2 *Electrical characteristics*

The electrical characteristics shall comply with Recommendation G.703. An optional non-return-to-zero (NRZ) electrical interface may be provided for special applications. The test load impedance shall be either 75 Ω unbalanced or 120 Ω balanced depending on the user requirement.

5.2.1.3 *Bearer frame structure*

The bearer frame structure shall comply with Recommendation G.704. Time slot 0 shall be used as recommended in G.704 and time slots 1 to 31 shall carry control channels and traffic according to the DCME frame structure.

5.2.2 *Bearer side interface at 1544 kbit/s*

5.2.2.1 *General*

For the point-to-point and multi-clique modes, the bearer interface shall consist of one 1544 kbit/s interface on the transmit side and one 1544 kbit/s interface on the receive side.

For the multi-destination mode, the bearer interface shall consist of one 1544 kbit/s interface on the transmit side and one to four 1544 kbit/s interfaces on the receive side.

5.2.2.2 *Electrical characteristics*

The electrical characteristics shall comply with Recommendation G.703. An optional non-return-to-zero (NRZ) electrical interface may be provided for special applications.

Due to the compressed nature of the DCME bearer interface and the necessity for 64 kbit/s unrestricted channel transmission, robbed-bit zero code suppression (ZCS) line coding techniques are prohibited on the 1544 kbit/s bearer channel interface. Only bipolar eight zero substitution (B8ZS) or zero byte time slot interchange (ZBTSI) line coding techniques are permitted.

5.2.2.3 *Bearer frame structure*

The bearer frame structure shall comply with Recommendation G.704.

Provisions shall be included in the bearer frame structure to accommodate control channels and traffic according to the DCME frame structure.

The 193rd bit shall be used for frame synchronization as recommended in Recommendation G.704.

5.3 *Signalling interfaces to switching equipment (ISC)*

The choice of interface is left for each administration to define within the constraints of their transmission facilities and ISCs.

The signalling interface to the switching equipment is dependent on the capability of the ISC and the facilities between the ISC and the DCME (see Recommendation Q.50).

5.3.1 *DCME-ISC signalling interface functions*

The following groups of functions are defined in Recommendation Q.50.

5.3.1.1 *Transmission resource management*

Facilitates the dynamic load control process within the ISC and the DCME concurrently, based on the status of the traffic loading on the DCME system. The requirements for this function are given in § 9.

5.3.1.2 *Seizure/release of 64 kbit/s circuits (see Note)*

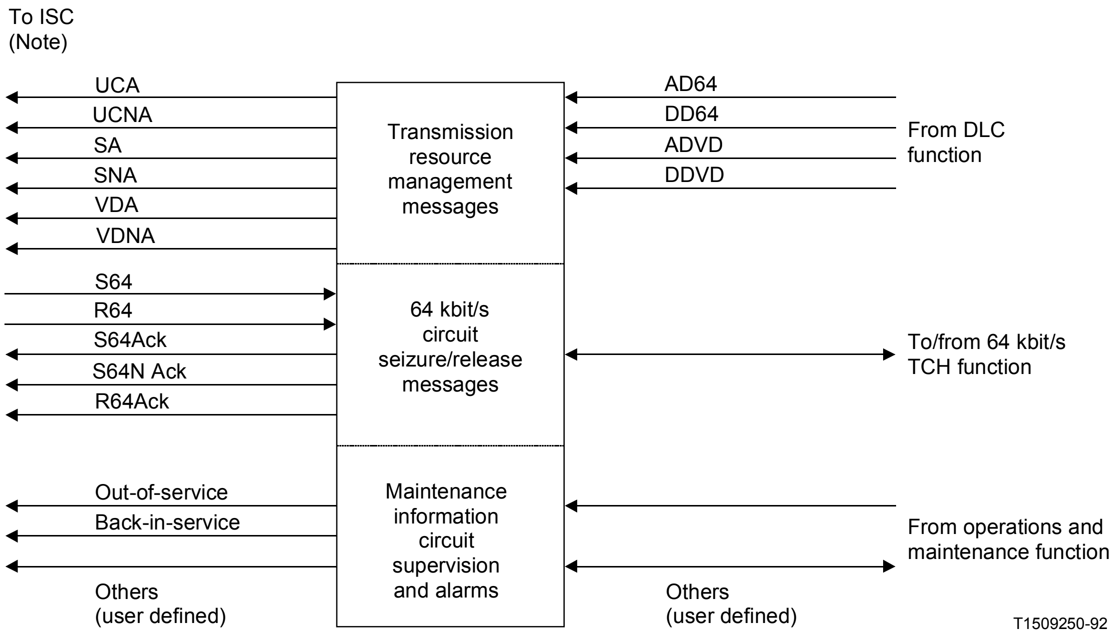
Used in the DCME for the generation of internal assignment and disconnection messages and in the ISCs for the validation of circuit seizure selection/release based on acknowledgement from the DCME. The requirements for this function are given in § 8.

Note – If the implementation of the ISC does not permit seizure/release of 64 kbit/s circuits, the provision of 64 kbit/s circuits may be accomplished under bilateral agreement by pre-assignment.

5.3.1.3 *Maintenance information*

Facilitates information exchange between the DCME and the ISCs pertaining to the maintenance status. Maintenance status information may be exchanged between the DCME and the ISC. This function may include the transfer of circuit supervision and alarm conditions referred to in § 15.

The DCME-ISC signalling system consists of one or more control links, a DCME control interface within the ISC and a switching centre interface (SCI) within the DCME. The selection of the DCME-ISC signalling system, including the physical and electrical interface characteristics are at the users' option. To permit this the SCI is defined with minimum functional requirements. Refer to Figure 4/G.763.



Note – These signal indications are available at this interface but may not all be used.

FIGURE 4/G.763

Switching centre interface unit

5.3.2 *External and internal messages/indications*

The SCI shall process the following external information elements (between the DCME and the ISC) and the following internal messages/indications (within the DCME). Depending on the characteristics of the chosen DCME-ISC signalling system, all of the following required external information elements may not be used.

External information element (Recommendation Q.50)	*Acronym*
Capacity for speech available	SA
Capacity for speech not available	SNA
Capacity for 3.1 kHz data available	VDA
Capacity for 3.1 kHz data not available	VDNA
Capacity for 64 kbit/s unrestricted available	UCA
Capacity for 64 kbit/s unrestricted not available	UCNA
Seizure/select 64 kbit/s circuit	S64
Seizure/select 64 kbit/s positive acknowledged	S64Ack
Seizure/select 64 kbit/s negative acknowledged	S64NAck
Release 64 kbit/s circuit	R64
Release 64 kbit/s circuit acknowledged	R64Ack
Circuit out-of-service	Out-of-service
Circuit back-in-service	Back-in-service

Internal messages/indications	*Acronym*
Activate DLC for voice/voice-band data traffic	ADVD
De-activate DLC for voice/voice-band data traffic	DDVD
Activate DLC for 64 kbit/s traffic	AD64
De-activate DLC for 64 kbit/s traffic	DD64

The interaction of these external information elements with the on-demand 64 kbit/s circuit handler (TCH), the dynamic load control (DLC) function and the operations and maintenance function is described in §§ 8, 9 and 15, respectively.

The format of all signals and messages is dependent upon the internal DCME design implementation and the chosen signalling interface, and is therefore not specified herein.

5.3.3 *Circuit numbering translation*

The SCI will perform the translation between the internal DCME IT numbering and the trunk channel identification used for the selected DCME ISC signalling system. This translation will be performed for any signalling function that requires individual trunk channels to be identified.

5.3.4 *Transmission resource circuit mapping*

The SCI will perform the mapping between each destination to which internal DLC messages apply and the primary multiplex streams, trunk channels, or time slots (depending on the selected signalling system) to which the associated external information elements apply. This mapping will make use of the TC-IT map information resident in the DCME described in § 15.1.

5.4 *Man-machine interface*

The DCME shall contain a system command structure which serves as a menu-driven interface between internal functions and the system operator. Typically two V24 ports are necessary to provide operator access to the equipment, one for a display terminal and one for a printer.

5.5 *Operations function interface*

5.5.1 *Trunk side operation at 2048 kbit/s or 1544 kbit/s*

The utilization of spare bits for monitoring and error protection shall be in accordance with Recommendations G.704 and G.706.

5.5.2 *Bearer side*

5.5.2.1 *Single destination mode*

The utilization of spare bits for monitoring and error protection is under study.

5.5.2.2 *Multi-clique or multi-destination mode*

The utilization of spare bits for monitoring and error protection is under study.

5.6 *Local alarms interface*

The DCME must present alarms to the local entity according to the user's requirement. The choice of the physical/electrical interface is to be decided by the individual Administration. In the case of individual voltage-free loop alarms, the categories of alarm in Recommendation G.803 should be included. In the case of a serial alarm interface it is recommended to provide as a minimum the following signals:

a) initial occurrence of an alarm in the monitored DCME;

b) initial occurrence of a clear in the monitored DCME;

c) receipt of a data request from the local entity;

d) initial system power-on.

Note – It is intended to study the inclusion of telecommunications management network (TMN) protocols and interface requirements in future DCME Recommendations.

5.7 *External clock interface*

5.7.1 *DCME working with 2048 kbit/s transmission interfaces*

The external clock interface shall comply with Recommendation G.703, § 10.3. The test load impedance shall be either 75 ohms unbalanced or 120 ohms balanced depending on the user requirement.

5.7.2 *DCME working with 1544 kbit/s transmission interfaces*

The timing is normally derived from an incoming digital link at 1544 kbit/s complying with Recommendation G.703, § 2. Where required an external clock interface may be provided.

5.8 *DCME frame structure*

5.8.1 *2048 kbit/s structure*

The bearer structure shall be compatible with the format specified in Recommendation G.704. The bearer structure shall contain 32 consecutively numbered 8 bit time slots, from 0 to 31. Time slot 0 shall be used for frame synchronization and special functions in accordance with Recommendation G.704. Time slots 1 through 31 shall be used to carry the DCME control channel(s) and traffic. The control channels are comprised of a unique word and a control channel message as described in § 11. In all figures used to illustrate the bearer frame structure, the left-most bit shall be transmitted first.

Sixteen 125 μsec bearer frames constitute one 2.0 ms DCME frame. The DCME frame is not required to be aligned with the multiframes defined in Recommendation G.704. The beginning of the DCME frame is identified by a unique word in the control channel(s).

5.8.2 *1544 kbit/s structure*

The bearer structure shall be compatible with the format specified in Recommendation G.704. Alternatives for 24-frame multiframe structure or 12-frame multiframe structure will be as agreed upon by the users. The bearer structure shall contain a framing bit (F-bit) and 24 consecutively numbered 8 bit time slots, from 1 to 24. The F-bit shall be used in accordance with Recommendation G.704. Time slots 1 through 24 shall be used to carry the DCME control channel(s) and traffic. The control channel(s) are comprised of a unique word, and a control message as described in § 11. In all figures used to illustrate the bearer frame structure the left-most bit shall be transmitted first.

Sixteen 125 μsec bearer frames constitute one 2.0 msec DCME frame. The DCME frame is not required to be aligned with the multiframes defined in Recommendation G.704. The beginning of the DCME frame is identified by a unique word in the control channel(s).

5.9 *Bearer BC numbering and use of the bearer frame*

As shown in Figure 5/G.763 for the 2048 kbit/s structure and Figure 6/G.763 for the 1544 kbit/s structure, one or two pools may be formed. Each pool shall contain an integer number of 8-bit time slots. The first 8-bit time slot of the first pool shall be TS1. The last 8-bit time slot of the second pool shall be TS31 (2048 kbit/s structure) or TS24 (1544 kbit/s structure). The upper boundary of the first pool and the lower boundary of the second pool shall be independently programmable (pre-set) at 8-bit time slot boundaries (see Note). Each pool will contain an even number of contiguous 4-bit time slots. The left-most 4-bit time slot shall carry the control channel as specified in § 11. The remaining 4-bit time slots of the pool are bearer channels (BCs) and are used to carry traffic.

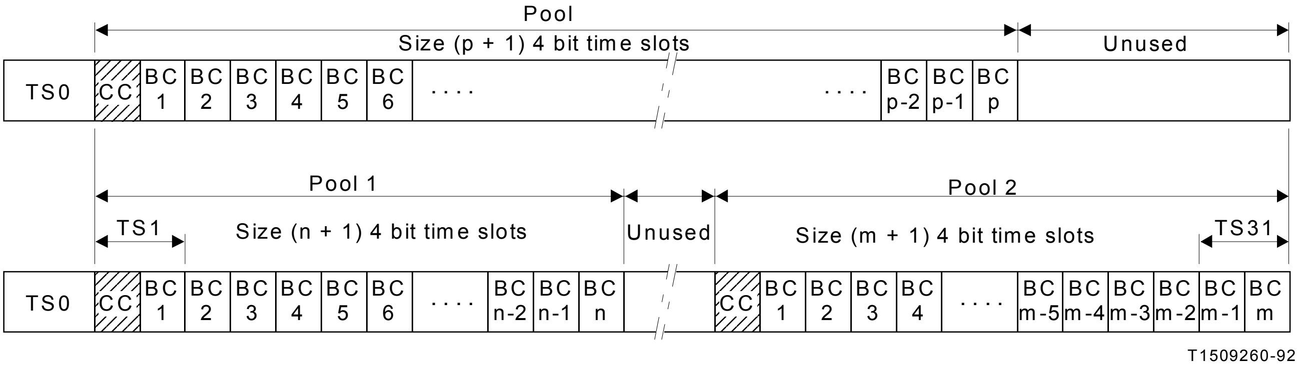
Note – The entire bearer structure need not be utilized by the pool(s). The unused portion of the bearer will contain an integer number of 8-bit time slots. This flexibility facilitates received pool sorting by a PCM cross-connect.

In the case where a bearer structure contains two pools (two control channels), transmit pools shall be mutually DCME frame aligned. Receive pools may not be mutually DCME frame aligned.

The normal range BCs of a pool are consecutively numbered from 1 to *p*, with BC No.1 being the 4-bit slot next to the control channel and *p* the total number of 4-bit slots in the pool, excluding the control channel. This numbering scheme is shown in Figures 5/G.763 and 6/G.763. The BC number contained in the assignment message can either be in the range 1 through 61 (normal BC numbering range) or in the range 64 through 83 (overload BC numbering range). If the optional 2 bit encoding mode is available and enabled, the overload BC numbering range is from 64 to 124. BCs in the normal range may consist of either 8, 5, 4, 3, or optionally 2 bits. These bits are obtained from bits of the bearer frame as described below.

BCs in the overload range may be disconnected or connected. If disconnected, they will not be associated with any bit of the bearer structure. If connected, they may consist of either 4, 3, or optionally 2 bit channels and will be associated with bits of the bearer frame as discussed later.

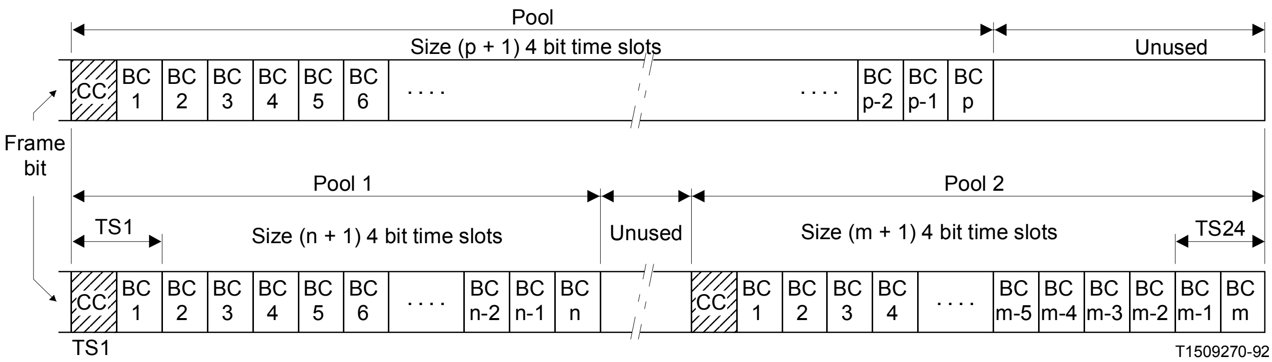
The criteria for associating the BC contained in the assignment message to bits within the bearer structure are as follows:



Note 1 – The boundary of pool 1 and the lower boundary of pool 2 are at the boundaries of 8-bit time slots within the Recommendation G.704 bearer frame structure. Unused portions of the bearer frame will also be integers of 8-bit time slots.
Note 2 – Left-most bits are transmitted first. Bit 1 of TS0, as defined in Recommendation G.704, is transmitted first.
Note 3 – For the special case of tandem operation the arrangement of the channels for pool 2 may be modified. This is under study.

FIGURE 5/G.763

DCME bearer frame structure and BC numbering scheme
2048 kbit/s bearer interface



Note 1 – The boundary of pool 1 and the lower boundary of pool 2 are at the boundaries of 8-bit time slots within the Recommendation G.704 bearer frame structure. Unused portions of the bearer frame will also be integers of 8-bit time slots.

Note 2 – Left-most bits are transmitted first. The F-bit, as defined in Recommendation G.704, is transmitted first.

Note 3 – For the special case of tandem operation the arrangement of the channels for pool 2 may be modified. This is under study.

FIGURE 6/G.763

DCME bearer frame structure and BC numbering scheme
1544 kbit/s bearer interface

5.9.1 *8 bit (64 kbit/s) BCs*

These are used for the assignment of unrestricted 64 kbit/s ITs. The BC number in the assignment message indicates the bearer BC (even number) which carries the first 4 bits (nibble) of the 8 bit sample. The second nibble is carried by the next higher BC. Refer to Figure 7/G.763.

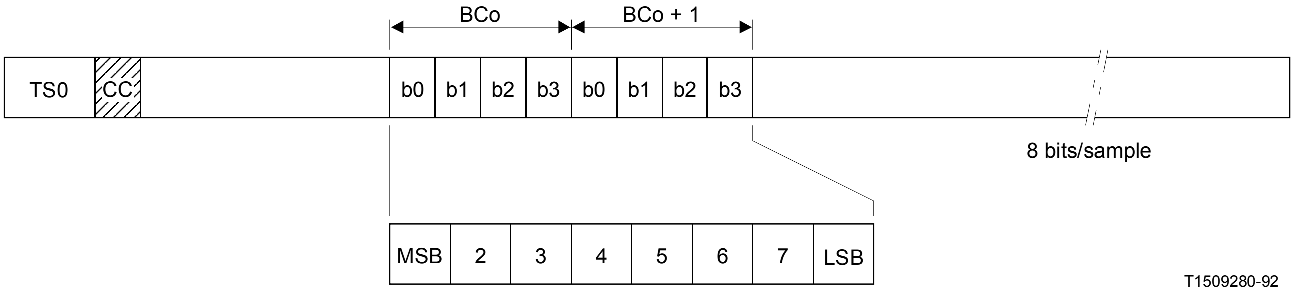


FIGURE 7/G.763
64 kbit/s bearer channel (8 bits/sample)

5.9.2 *5-bit (40 kbit/s) BCs*

These are used for the assignment of voice-band data ITs. The BC number in the assignment message indicates the bearer which carries the first 4 bits of the 5-bit sample. The 5th bit (LSB) is obtained from a different bearer which is independently assigned as a Bit Bank. The Bit Bank constitutes a bit pool providing one bit for up to four data channels. The selection of the 5th bit for a data IT is regulated by the DCME processes. Refer to Figure 8/G.763.

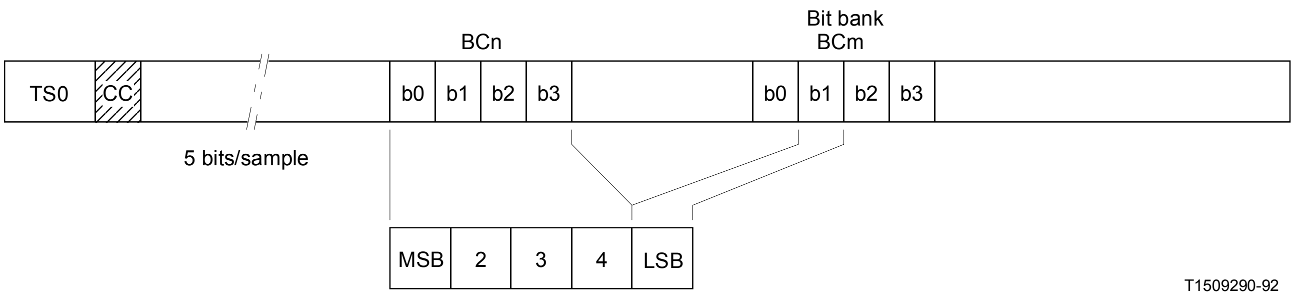


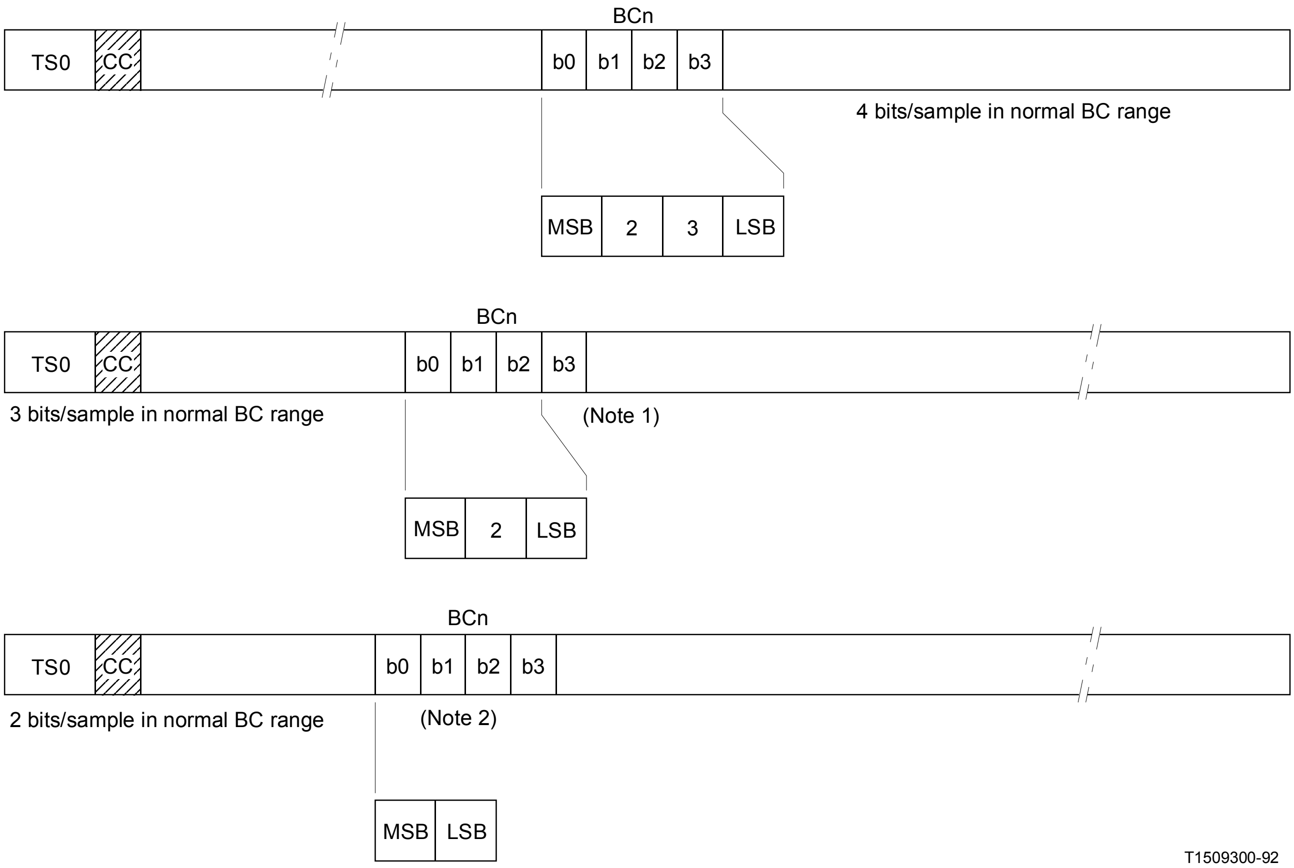
FIGURE 8/G.763
40 kbit/s bearer channel (5 bits/sample)

5.9.3 *Normal range 4-bit BCs*

These BCs are used for the assignment of bit banks. The BC number in the assignment message indicates the bearer BC performing the bit bank function.

5.9.4 *Normal range 4/3-bit (32/24 kbit/s) BCs*

These BCs are used for the assignment of voice ITs. The BC number in the assignment message indicates the bearer BC which carries the IT bits. These can be either three or four depending on whether the bearer BC LSB is used for the creation of an overload BC during high load conditions. Bit stealing will occur randomly for the purpose of voice quality equalization over the ensemble of voice ITs. The bit stealing function is controlled by the DCME processes. Refer to Figure 9/G.763.



Note 1 – b3 has been used in the creation of an overload channel.
Note 2 – b2 and b3 have been used in the creation of an overload channel(s).

FIGURE 9/G.763
32, 24 and 16 kbit/s normal range bearer channel
(4, 3 and 2 bits/sample)

5.9.5 *Overload range 4/3-bit (32/24 kbit/s) BCs*

These BCs are used, during heavy load conditions, for the assignment of voice ITs. These BCs can be either 3 or 4 bits as determined by the DCME processes. Changes from 3 to 4 bit encoding (and *vice versa)* will occur randomly for the purpose of voice quality equalization over the ensemble of voice ITs. The BC number in the assignment message has no direct correspondence to any bearer BC. Refer to Figure 10/G.763.

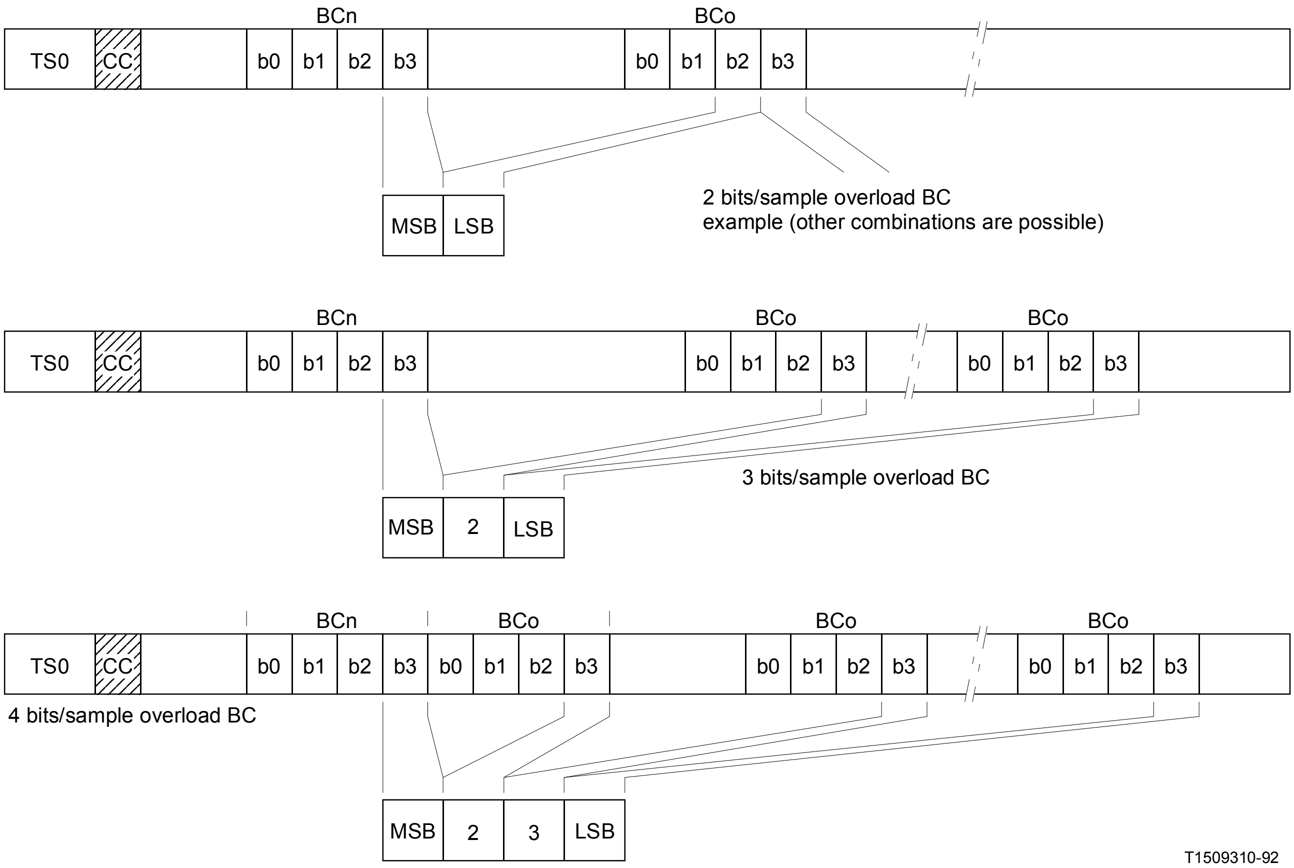


FIGURE 10/G.763

An example of 32, 24 and 16 kbit/s overload bearer channels (4, 3 and 2 bits/sample)

5.9.6 *Normal and overload range 3/2 bit (24/16 kbit/s) BCs resulting from the optional 3/2 bit overload procedure*

If the optional 2 bit ADPCM encoding feature is available and enabled, normal and overload channels may operate in 3/2 bit mode as required by the 3/2 bit overload procedure (§ 6.1.7.2). This procedure parallels the 4/3 bit procedure and it is invoked when more overload channels are required than provided by the 4/3 bit procedure.

5.9.7 *Pre-assigned BCs*

It is possible to select certain ITs for a semi-permanent connection to the bearer resources (pre-assigned ITs), therefore bypassing the dynamic assignment function (DAF) of the DCME. 64, 40 and 32 kbit/s ITs can be pre-assigned.

The optional 24 kbit/s or 16 kbit/s pre-assigned channels intended for maintenance purposes will occupy the most significant 3 or 2 bits respectively of the 32 kbit/s pre-assigned normal range BCs.

The allocation of pre-assigned ITs to portions of the bearer frame is specified by entering the appropriate information into the DCME configuration data (see § 6.1).

The BC specified in the configuration data indicates the bearer BC carrying the first four bits of the IT. For a 64 kbit/s IT, the BC must be even numbered (the use of the next higher BC for this 64 kbit/s IT is implied). A sufficient number of Bit Banks must be pre-assigned to provide the 5th bit for the pre-assigned 40 kbit/s channels.

The bearer BCs carrying 40 kbit/s pre-assigned ITs shall be contiguous starting from bearer BC No.1. The lowest bearer BC numbers shall be used for the Bit Banks required for 40 kbit/s pre-assigned ITs, followed by the pre-assigned 40 kbit/s BCs.

It is recommended that the pre-assigned 32 kbit/s ITs (other than bit banks) and the pre-assigned 64 kbit/s ITs also utilize the lower portion of the bearer BC numbers.

6 DCME transmit unit

The function of the DCME transmit unit structure is to provide connections between the ITs, the ADPCM encoders and the BCs, and to generate assignment messages for correspondent DCMEs. At initialization, the various transmit side functions are loaded with the appropriate configuration data, and pre-assigned ITs are connected to ADPCM encoders and BCs, as required. Each dynamically assigned IT is continuously monitored for detection of activity and classification as voice, data or signalling. Active ITs are then dynamically assigned to available BCs (and ADPCM encoders). At the request of the ISC, 64 kbit/s IT are assigned to available BCs and kept connected until the ISC releases the circuit. Assignment information is sent to the remote DCME via an assignment message which is generated once during every 2 ms DCME frame. The actual connection is implemented three DCME frames following transmission of the message. This delay is necessary in order to provide adequate time for processing the message before the associated ADPCM BC samples arrive at the DCME receive unit.

This section specifies those aspects of the DCME transmit side structure which are necessary to accurately define the DCME transmit unit/receive unit interaction. An example of a DCME transmit side structure which satisfies the interaction requirements specified in this section is given in Annex A.1.

6.1 *Transmit channel processing function*

The transmit channel processing (TCP) function examines the input ITs and classifies them according to their signal type and status. Service requests are placed in assignment queues as a consequence of IT status and type transitions. The assignment queues are serviced and the associated assignment messages are then generated. In servicing the assignment queues, the ADPCM encoders are directed to operate at various bit rates under the control of the overload channel creation function, the data/speech discriminator and the transparent circuit handler (TCH). The resulting ADPCM samples are dynamically placed in the DCME output bearer frame at specific BC locations under the control of the TCP function.

6.1.1 *DCME transmit unit initialization*

At initialization, the DCME transmit unit channel connectivity map shall be set to a known state (BCs and ITs disconnected) and the TCP parameters shall be loaded. This includes the information necessary for the allocation of pre-assigned channels and bit banks. The pre-assigned channel allocation (determined by the configuration data) shall be in accordance with the bearer structure requirements (§§ 5.2 and 5.9).

6.1.2 *Intermediate trunk classification*

The intermediate trunk signals must be continually examined to determine their activity and their type (i.e. speech, signalling or data). The activity and type characteristics of the intermediate trunk signals are determined by the activity detector, the data/speech discriminator and the signalling detector. Transitions from one activity/type state to another are used by the input pre-processor function to generate service requests.

The intermediate trunk classification process shall classify input signals as specified below.

a) Initially, this function shall classify an IT as either pre-assigned, if so designated by the configuration data, or as voice-inactive, if subject to dynamic assignment.

b) Whenever a 64 kbit/s assignment request transpreq indication is received from the TCH, the IT shall be classified as transparent and shall remain in this condition until a 64 kbit/s disconnection request transprel indication is received from the TCH, in which case the signal classification shall change to voice-inactive.

c) If an IT is active and of the voice/signalling type and a data-detect indication is received from the data/speech discriminator, the IT shall be classified as data-active. The same applies to the case of reception of a Rxdata indication from the DCME receive side. The Rxdata indication is generated by the DCME receive unit when an assignment message is received which establishes a voice-band data connection.

   If an IT is inactive and of the data type and an act indication is received from the activity detector, the IT shall be classified as data-active.

d) If an IT is inactive and of the voice/signalling type and a Rxdata indication is received, the IT shall be classified as data-inactive.

   If an IT is of the data type and the hangover expires, the IT shall be classified as data-inactive.

e) If an IT is of the voice/signalling type and the hangover expires, the IT shall be classified as voice-inactive.

f) If an IT is inactive and of the voice type and an act indication is received from the activity detector, the IT shall be classified as voice-active.

g) If an IT is active and of the data type and a voice-detect indication is received from the data/speech discriminator, the IT shall be classified as voice-active.

   If an IT is inactive and of the voice type and an act message is received, the IT shall be declared voice-active.

h) If an IT is active and of the voice type and a signaldetect indication is received from the signalling detector, the IT shall be classified as signalling-active.

   If an IT is of the signalling type and an act indication is received, the IT shall be classified as signalling-active.

i) If an IT is active and of the data type and a signaldetect indication is received, the IT shall be classified as signalling-active.

   If an IT is active and of the voice type and a signaldetect indication is received, the IT shall be classified as signalling-active.

When activity terminates on an IT classified as voice-active, the voice hangover value shall be used. When activity terminates on an IT classified as signalling-active, the signalling hangover value shall be used. The voice and signalling hangover values shall be in accordance with the hangover masks specified in § 12.

Provisions shall be provided to maintain channel connectivity between page changes in the forward direction of a facsimile transmission and to release the reverse channel connection between procedural signal transmissions so as to achieve a higher return link utilization for facsimile transmissions (this feature is also referred to as silence elimination).

The Rxdata indication which is generated by the DCME receive unit upon receiving a voice-band data related assignment message shall be used to initiate a voice-band data connection at the DCME transmit unit.

6.1.3 *Input preprocessing*

The input preprocessing function examines the activity/type state transitions originating from the intermediate trunk classification process and generates assignment service requests which are placed in service request queues. The input preprocessing function shall process the intermediate trunk state transitions as specified below.

When a data-inact(IT) indication is received from the intermediate trunk classification process, the BC connection to this IT shall be checked. If the type of the BC is data or voice, the BC type shall be maintained and the BC shall be placed back into the pool of available BCs.

When a voice-inact (IT) indication is received from the intermediate trunk classification process, the BC connection to this IT shall be checked. If the type of the BC is data or voice, the BC type shall be maintained and the BC shall be placed back into the pool of available BCs. If the BC is in the overload range, a disconnect request shall be stored in the overload disconnect queue.

When a Voice(IT) indication is received from the intermediate trunk classification process, the type of the BC connected to this IT shall be checked. If the type of BC is voice, the BC type shall be maintained and no request shall be generated. If the BC type is other than voice, a request shall be stored in the voice assignment queue.

When a data(IT) indication is received from the intermediate trunk classification process, the type of the BC connected to this IT shall be checked. If the type of BC is data, the BC type shall be maintained and no request shall be generated. If the BC type is other than data, a request shall be stored in the data assignment queue.

When a transp(IT) indication is received from the intermediate trunk classification process, a request shall be stored in the 64 kbit/s assignment queue.

6.1.4 *Service request implementation*

The service request implementation function examines the service request queues and generates assignment messages in response to service requests as specified below. The priorities associated with servicing the various queues have not been specified since they are implementation dependent and do not affect the equipment interworking capability.

6.1.4.1 If the optional USM is used, the USM queue shall be addressed in DCME frames 0, n, 2n, etc. (i.e. every nth DCME frame) of the DCME multiframe where n is a variable which may be selected by the user. For optional R2 USM, DCME frames 0, 8, 16, 24, 32, 40, 48 and 56 of the DCME multiframe shall include 20 bits comprising the synchronous part of the CC, which shall be used to convey two IT numbers and their respective line signalling information. Upon servicing the request, the message shall be deleted from the USM queue. The other service request queues shall be addressed in the remaining DCME frames.

6.1.4.2 *64 kbit/s disconnect request*

The request at the top of the 64 kbit/s disconnect queue shall be addressed. An assignment shall be generated which disconnects the IT. At implementation, the service request shall be deleted from the 64 kbit/s disconnect queue.

6.1.4.3 *Overload disconnect request*

The request at the top of the overload disconnect queue shall be addressed. An assignment shall be generated which disconnects the IT. At implementation, the serviced request shall be deleted from the overload disconnect queue.

6.1.4.4 *64 kbit/s assignment request*

The request at the top of the 64 kbit/s assignment queue shall be addressed. The IT for which the request was generated shall be checked to determine whether it is connected or disconnected. If the IT is connected, a count of the usable bits in the pool shall be taken to determine whether enough capacity exists to accommodate the additional bits required. If no capacity exists, an assignment which disconnects the available BC (and the associated IT) shall be generated.

If the IT is connected and enough capacity exists to accommodate the additional bits, the BC number of the connected bearer channel (number k) shall be checked to determine whether it is even or odd. If k is even, the next higher BC (number $k+1$) shall be examined. If k is odd, the next lower BC (number $k-1$) shall be examined. The objective is to allocate the first nibble (containing the MSB) of the 64 kbit/s channel to an even numbered normal range BC. If the IT is disconnected, the number of usable bits in the pool shall be counted to determine whether the request can be accommodated (8 bits are required). If sufficient capacity is available, the IT shall be assigned to the available normal range BC pair. If sufficient capacity is not available, a refresh message shall be generated in accordance with § 6.1.5.

If the assignment of a 64 kbit/s channel requires that existing ITs be reassigned to different BCs in order to make room in the DCME bearer frame for the 64 kbit/s BC, then the reassignments shall be done with highest assignment priority and in an orderly manner so that the connections between assigned ADPCM encoders and ADPCM decoders are not broken. At implementation, the service request shall be deleted from the 64 kbit/s assignment queue.

6.1.4.5 *Data request*

5 bit encoding is required for the transmission of a data channel. Four bits are obtained by assigning the data IT to a 4 bit BC in the normal BC range. The fifth bit (LSB) is obtained from a pool of four bits referred to as the bit bank. Special 4 bit BC channels of the bank type are created in the bearer frame for this purpose. The criteria for creation of bank channels are specified in Table 3/G.763.

TABLE 3/G.763

Criteria for bit bank creation/deletion

a) Creation of bit bank (At new data channel assignment)		
Data BC Available	Yes	Bank not required
	No	Bank required if $nb < \frac{nd}{4}$ (Note 1) (Note 2)
b) Deletion of bit bank (Note 3)		
$nb \geq \frac{nd}{4} + 1$		Delete highest numbered bank

Note 1 – nd = No. of data channels in use and requested (pre-assigned and dynamic assigned).

Note 2 – nb = No. of banks in use.

Note 3 – If bank was just created, wait two DCME frames before applying deletion criterion. If during the 2 DCME frame wait period a USM signalling message is sent, wait one additional DCME frame.

The request at the top of the data assignment queue shall be addressed. First, it shall be determined whether a new bit bank is required. Then, the IT for which the request was generated shall be checked to determine whether it is connected to a BC.

If the IT is connected, a bit count shall be made to determine whether enough bearer capacity exists for the allocation of the data IT (including the creation of an additional bank channel if required).

If sufficient capacity exists and a new bit bank is not required, an assignment of the data IT to the connected BC shall be made.

If a bit bank is required, an assignment message connecting the selected BC to IT No. 250 shall be generated. At the next DCME frame, the data IT shall be assigned to the connected BC.

If sufficient capacity is not available and the IT is connected, a disconnect message shall be generated.

If the IT is disconnected, a count of the usable bits in the pool shall be made to determine whether enough capacity exists to allocate the data call. If sufficient capacity is not available, a refreshment message shall be generated in accordance with § 6.1.5. If sufficient capacity is available and a new bit bank is not required, the data IT shall be assigned to an available BC. If the bit bank is required, it shall be assigned first, and then the data IT shall be assigned to an available BC. If the assignment of a voice-band data channel required that existing ITs be reassigned to different BCs in order to make room in the DCME bearer frame for the 40 kbit/s BC, then the reassignments shall be done with highest assignment priority and in an orderly manner so that the connections between assigned ADPCM encoders and ADPCM decoders are not broken. At implementation, the service request shall be deleted from the data assignment queue.

6.1.4.6 *Voice request*

The request at the top of the voice assignment queue shall be addressed. The IT for which the request was generated shall be checked to determine whether it is connected to a BC.

If connected and the BC type is available, the IT shall be assigned to the available BC.

If the IT is disconnected, the usable bits in the pool shall be counted to determine whether enough capacity exists to accommodate the voice call. If sufficient capacity does not exist, a refreshment message shall be generated in accordance with § 6.1.5.

If sufficient capacity exists, the voice IT shall be assigned to an available BC.

At implementation, the service request shall be deleted from the voice assignment queue.

6.1.5 *Refreshment message generation*

In DCME frames when the USM queue is not to be addressed and there are no messages in the remaining queues, a refreshment message shall be generated. A refreshment message shall also be generated when a service request queue cannot be serviced due to unavailable bearer capacity unless a disconnect message is generated.

The refreshment shall be done by scanning the range of normal BCs (from BC No. 1 to the upper pool boundary) and the range of overload BCs (from BC No. 64 to the highest number permitted). Pre-assigned BCs shall not be refreshed. Each dynamically assigned 64 kbit/s connection shall be refreshed but only limited to the even numbered BC (the next higher BC shall not be refreshed). Every other refreshment message shall alternate between the normal and the overload BC range. In each range, the BCs shall be refreshed sequentially from the lowest to the highest BC, restarting the cycle after refreshment of the highest BC in the range. Whenever a BC is refreshed, all the required information elements shall be inserted in the assignment message. The refreshment of a bit bank BC shall be transmitted with IT No. 250.

6.1.6 *ADPCM encoder control*

Whenever a new assignment of a previously disconnected IT is made, an ADPCM encoder shall be selected from the available encoders of the encoder pool. Whenever a reassignment of a previously connected IT is made, the ADPCM encoder currently associated with the IT shall be maintained.

Whenever an IT is disconnected, the associated ADPCM encoder shall be returned to the pool of available ADPCM encoders.

Resetting of the ADPCM encoder shall be done when the IT connection to an ADPCM encoder is changed. The ADPCM encoder shall be reset before establishing a new connection.

6.1.7 *Bit bank handling and overload channel creation*

Inherent to the TCP function are the bit bank handling and the overload channel creation processes. The bit bank handling process derives the LSB of each data channel from one of the existing bit banks according to predetermined rules.

When the overload mode is required, the overload channel creation process distributes the 3 bit encoding across the entire set of speech channels. The objective is to make the average encoding rate almost equal for the normal voice BC (subject to bit stealing) and the overload channels. This is achieved by stealing bits from eligible BCs in a pseudo-random fashion and by making each overload BC alternate between 3 bit and 4 bit encoding (also in pseudo-random fashion).

When the 4/3 bit overload channel creation procedure, described above, cannot provide the required number of overload channels, the optional 3/2 bit overload channel creation procedure may be used. This procedure, similarly to the 4/3 bit procedure, makes the average encoding rate almost equal for the normal voice BCs (subject to bit stealing) and the overload channels. Pseudo-random bit rotation and switching between two alternate overload channel sizes (3 bit and 2 bit channels) are also used in this case.

6.1.7.1 *Bit bank handling process*

The bit bank handling process maintains two lists of BCs which are used in allocating the LSB of each data channel. All lists contain, in ascending order, the BC numbers that fall into the categories defined below. BCs are extracted from, or inserted into, these lists always keeping the BC numbers in ascending order. A BC can appear only in one list at the same instant.

Data list – This list contains all BC numbers which are connected to data channels. At initialization this list contains no entries.

Bit bank list – This list contains all BC numbers which are currently being used to form bit banks. At initialization, this list contains no entries.

Pre-assigned 40 list – This list contains all BC numbers that are pre-assigned as 40 kbit/s channels. At initialization, this list contains no entries.

A bit bank BC number shall be placed into the bank list at the generation of an assignment message containing IT No. 250, if the associated BC number does not already exist in the bit bank list. The BC number in question shall be removed from the voice list when this occurs.

A bit bank BC number is removed from the bank list when the bit bank BC is no longer needed. The bit bank deletion shall be in accordance with the criterion specified in Table 3/G.763. When the conditions for the deletion of a bit bank exist, the highest numbered bit bank BC shall be deleted. The BC number shall then be put back into the voice list.

The bit position of the LSBs of the handled data channels (pre-assigned 40 or dynamically assigned channels declared as data) shall be assigned in the following way:

a) LSB of BC number in pre-assigned 40 list position 1: MSB of BC number in bank list position 1;

b) LSBs of BC numbers in pre-assigned 40 list positions 2 to 4: second, third and fourth bits, respectively, at BC number in bank list position 1;

c) LSB of BC number in pre-assigned 40 list position 5: MSB of BC number in bank list position 2.

This procedure is followed until the BC numbers in the pre-assigned 40 list have been handled, after which the BC numbers in the data list are handled in the same manner.

6.1.7.2 *Overload channel creation process*

The overload channel creation process maintains two lists of BCs which are used in forming overload channels. All lists contain, in ascending order, the BC numbers that fall into the categories defined below.

BCs are extracted from, or inserted into, these lists always keeping the BC numbers in ascending order. A BC can appear only in one list at the same instant.

Voice list – This list contains all BC numbers that are within the normal BC range and can contribute bits for the creation of overload channels. At initialization, this list contains all normal BC numbers subject to DSI.

Overload list – This list contains all BC numbers that are within the overload BC range. At initialization, this list contains no entries.

When an assignment message is generated, the voice list or overload list is updated and the list lengths Nv (voice list) and Nov (overload list) are computed. If Nov is 0, overload channel creation is not required.

6.1.7.2.1 *4/3 bit overload channel creation procedure*

If Nov is greater than 0, but not greater than Nv/3 the number (N4) of channels in the overload range that will be carried at four bits per sample shall be computed as follows:

$$N4 = \text{Integer}\left[\frac{Nv \times 4 \times Nov}{Nv + Nov} + \frac{1}{2}\right] - Nov \times 3$$

In addition, when Nov is greater than 0, the integer variables Pv and Pov shall be computed as follows:

$$Pv = (\text{IT modulo Nv})$$
$$Pov = (\text{IT modulo Nov})$$

where, IT is the IT number contained in the assignment message (see note). Pv and Pov represent voice and overload list positions. These positions are numbered from 0 to Nv − 1 and from 0 to Nov − 1, respectively.

Note – If an optional USM is being used, the IT number information in DCME frames 0, n, 2n, etc. (i.e. every nth DCME frame) of the DCME multiframe shall also be used to create overload channels.

The BCs stored in the overload list at the first N4 locations (modulo Nov) starting from the list position Pov shall be marked as 4 bit channels. The remaining BCs of the overload list shall be marked as 3 bit channels.

The 4/3 bit mode of the first BC in the overload list shall be checked to determine whether the 3 bit or 4 bit mode is used. If the mode is 3 bit, the BCs stored in the voice list at the first three locations, starting from the list position, Pv, shall be set to 3 bit mode. The following bit associations shall be entered into the BC bit map:

a) MSB of BC in overload list position 0: LSB of BC in voice list position Pv;

b) second bit of BC in overload list position 0: LSB of BC in voice list position (Pv + 1 modulo Nv);

c) LSB of BC in overload list position 0: LSB of BC in voice list position (Pv + 2 modulo Nv).

If the first BC in the overload list is a 4 bit channel, items a) and b) above remain applicable, c) changes and d) is added as shown below:

c) third bit of BC in overload position 0: LSB of BC in voice list position (Pv + 2 modulo Nv);

d) LSB of BC in overload list position: LSB of BC in voice list position (Pv + 3 modulo Nv).

The same procedure shall be applied to the second BC in the overload list, starting at either position Pv + 4 or Pv + 3, depending on the 4/3 bit mode of the first BC.

The same procedure shall be applied to the remaining BCs of the overload list. The voice list BCs subject to bit stealing will be marked as 3 bit channels, the remaining ones as 4 bit channels. An example of the procedure is illustrated in Figure 11/G.763.

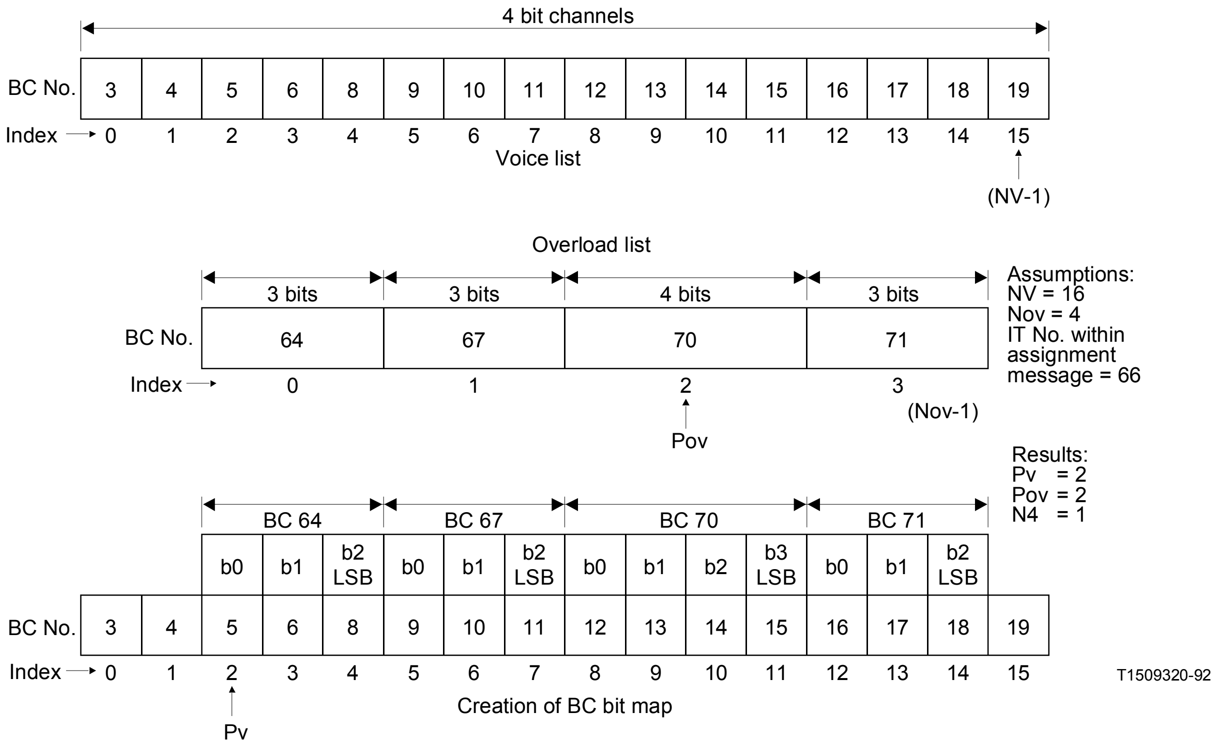


FIGURE 11/G.763

Example of 4/3 bit overload channel creation procedure

6.1.7.2.2 *3/2 bit overload channel creation procedure*

If Nov is greater than Nv/3 and the optional 2 bit overload feature is available and enabled, the number (N3) of channels in the overload range that will be carried at three bits per sample shall be computed as follows:

$$N3 = \text{Integer}\left[\frac{\text{Nv} \times 4 \times \text{Nov}}{\text{Nv} + \text{Nov}} + \frac{1}{2}\right] - \text{Nov} \times 2$$

The number (n2) of bearer channels in the voice list that will operate at two bits per sample (i.e. these channels donate two bits) shall be computed as follows:

$$n2 = N3 - \text{Nv} + \text{Nov} \times 2$$

In addition, the integer variables Pv and Pov shall be computed as follows:

$$\text{Pv} = (\text{IT modulo Nv})$$
$$\text{Pov} = (\text{IT modulo Nov})$$

where, IT is the IT number contained in the assignment message (note). Pv and Pov represent voice and overload list positions. These positions are numbered 0 to Nv-1 and from 0 to Nov-1, respectively.

Note – If an optional USM is being used the IT number information in DCME frames 0, n, 2n, etc. (i.e. every nth frame) shall also be used to create overload channels.

The BCs stored in the overload list at the first N3 locations (modulo Nov), starting from the list position Pov, shall be marked as 3 bit channels. The remaining BCs of the overload list shall be marked as 2 bit channels.

The BC stored in the voice list at the first n2 locations (modulo Nv), starting from the list position Pv, shall be marked as 2 bit channels. The remaining BCs of the voice list shall be marked as 3 bit channels (i.e. these channels donate one bit).

In order to obtain the bit associations for the BC bit map, the donated bits from the channels of the voice list shall be arranged in the following ordered sequence:

Place in the sequence	*Donated Bit in voice List* (see Note)
1st	2nd bit of BC at position Pv
2nd	LSB of BC at position Pv
3rd	2nd bit of BC at position Pv + 1
4th	LSB of BC at position Pv + 1
5th	2nd bit of BC at position Pv + 2
6th	LSB of BC at position Pv + 2, etc.

The 3/2 bit mode of the first BC in the overload list shall be checked to determine whether the 2 bit or 3 bit mode is used.

If the first BC in overload list is a 2 bit channel, the following bit associations shall be entered into the BC bit map:

a) MSB of BC in overload list position 0: 1st bit of the sequence;

b) LSB of BC in overload list position 0: 2nd bit of the sequence.

If the first BC in the overload list is a 3 bit channel, the same procedure shall be applied to item a), but item b) changes and item c) is added as follows:

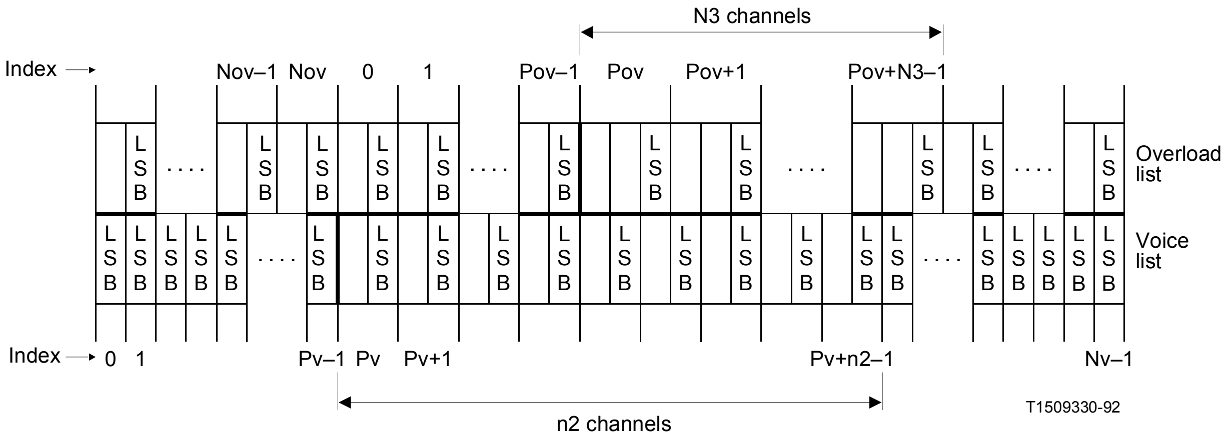
b) second bit of BC in overload list position 0: 2nd bit of the sequence;

c) LSB of BC in overload list position 0: 3rd bit of the sequence.

The same bit association procedure shall be applied to each remaining BC in the overload list: for these BCs, the association will start from the first available bit in the donated bit sequence.

This procedure is illustrated in Figure 12/G.763. A particular example for a pool of nine BCs, No. 1 to BC No. 9, is shown in Figure 13/G.763.

Note – In some cases the second bit of BC in voice list, indicated above, may not be available depending on the value of N2, in which case the sequence is compacted upwards.



Note – The overload channel bits are matched with the corresponding stolen bits of the voice channels.

FIGURE 12/G.763

3/2 bit overload channel creation procedure

Assumptions	Computed parameters
Nv = 9	N3 = 3
IC = 2	N2 = 1
Nov = 4	n2 = 2

Note – The overload channel bits are matched with the corresponding stolen bits of the voice channels.

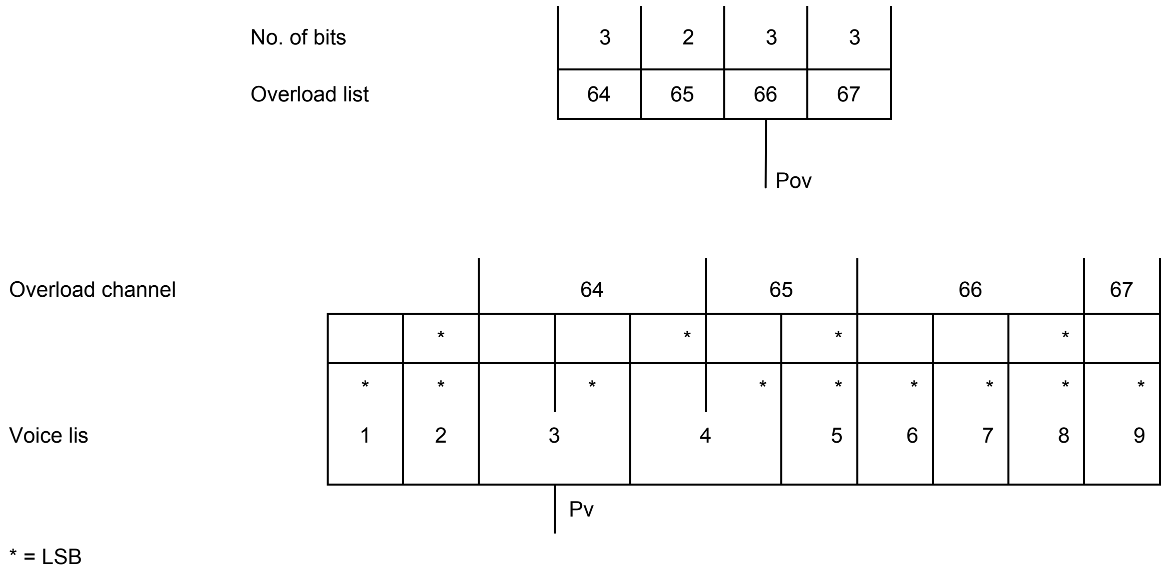


FIGURE 13/G.763

Example of 3/2 bit overload channel creation

6.1.8 *Connectivity implementation delay*

The IT-ADPCM Encoder-BC connection is implemented at the beginning of the DCME frame which occurs three frames after the start of the DCME frame containing the applicable assignment message. Refer to Figure 14/G.763.

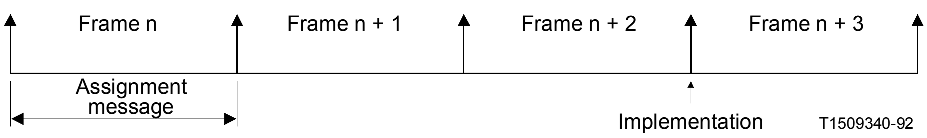


FIGURE 14/G.763

Implementation timing

7 DCME receive unit structure

The function of the DCME receive unit is to provide connections between the BCs, the ADPCM decoders and the ITs. At start-up, the various processes are loaded with the appropriate configuration data, and pre-assigned BCs are connected to ADPCM decoders and ITs, as required. Assignment messages are decoded to obtain information required for dynamic assignments of non-pre-assigned BCs to ITs (and ADPCM decoders). A sufficient number of ADPCM decoders shall be provided to guarantee that freeze-out due to unavailability of ADPCM decoders *cannot occur* (see Note). The actual connection is implemented at the beginning of the DCME frame which occurs three frames after the start of the DCME frame containing the applicable assignment message. This delay is necessary in order to provide adequate time for processing the assignment message before the associated ADPCM BC samples arrive at the DCME receive unit.

Note – For the point-to-point mode this condition is met if the number of ADPCM decoders is equal to the number of ADPCM encoders. For the multi-destination mode this condition is met if the number of ADPCM decoders provided equals the number of trunk channels receiving traffic.

This section specifies those aspects of the DCME receive unit structure which are necessary to accurately define the DCME transmit unit/receive unit interaction. An example of a DCME receive unit structure which satisfies the interaction requirements specified in this section is given in Annex A.2.

7.1 *Receive channel processing function*

The receive channel processing (RCP) function decodes the received assignment messages, dynamically establishes BC-decoder-IT connectivity relationships and provides information to the transparent circuit handler and the transmit channel processing functions.

7.1.1 *DCME receiver unit initialization*

At initialization, the receive side channel connectivity map shall be set to a known state (BCs and ITs disconnected) and the RCP parameters shall be loaded. These parameters include the information necessary for the allocation of pre-assigned channels and bit banks. The pre-assigned channel allocation (determined by the configuration data) shall be in accordance with the bearer structure requirements (§§ 5.2 and 5.9). A map which identifies the remote IT numbers intended for the DCME and associates them with the local IT numbers (making up the circuit), is included in information loaded at initialization. The local IT numbers are used by the DCME in its transmitted assignment message. The remote IT numbers are those associated with the individual channels received by the local DCME receive unit from its corresponding DCME transmit units.

7.1.2 *Input pre-processing*

Upon reception of an assignment message, a validity check shall be performed to ensure that the message is consistent with the transmit side assignment rules and with the DCME configuration data. A minimum list of conditions to be verified shall be as specified below:

a) If the BC is in the overload range, or if the IT number is 250, the MSB of the BC identification word in the assignment message must be 0.

b) If the synchronous data word indicates a 64 kbit/s transparent channel, the MSB of the BC identification word must be 0, and the BC number must be even.

c) The BC number must not already be used for a pre-assigned channel.

d) The IT number must be contained in the range which the corresponding DCME encoder can address regardless of destination.

e) The BC number must be in the normal range if the BC type is data or transparent, or if the IT number is 250.

f) If the optional USM is used special checks are required for DCME frames 0, n, 2n, etc., of the DCME multiframe.

   For the optional R2 USM line signalling, messages will be delivered in DCME frames 0, 8, 16, etc. (i.e. every eighth frame of the DCME multiframe). The validity of the two IT numbers conveyed should be checked against the permissible range.

If any of the above conditions are not satisfied, or if DCME frame alignment is lost, no further processing of the assignment message shall be performed. The receive IT number shall be assumed to be 0 for the purpose of providing the Pv and Pov pointer values for the overload channel derivation process.

7.1.3 *Connectivity map update*

If the assignment message validity check is successful, the received control channel message shall be processed as follows:

a) The IT-to-BC connection shall be logged in the channel connectivity map.

b) If the IT number is 0, the BC shall be disconnected from the previously connected IT, defined as ITP.

c) If the IT number is 250, the BC shall be interpreted as a bit bank channel.

d) If a BC of the transparent type is indicated by the assignment message, BC + 1 shall be designated as disconnected in the channel connectivity map.

e) If the connection of a BC changes to a different IT, the previously connected IT, defined at ITP, shall be disconnected. This is an implicit disconnection of ITP.

f) If the connection of an IT changes to a different BC, the previously connected BC shall be designated as disconnected in the channel connectivity map.

g) If a BC of the transparent type changes to a different type, the other BC of the transparent BC pair shall be designated as disconnected in the channel connectivity map. Its associated IT shall also be designated as disconnected in the channel connectivity map.

If, as a result of the above actions, the conditions exist for the deletion of a bit bank (as per Table 3/G.763), the bit bank BC shall be disconnected.

If the optional USM is used, the channel connectivity map shall not be updated. However, ITn shall be used as a pointer in the overload channel derivation process.

It should be noted that some connection/type changes are not strictly permissible under the assignment rules specified for the DCME transmit unit structure (see § 6). These transitions, although abnormal, may occur at the DCME receive unit as the result of loss of assignment messages. Note that abnormal transitions are different from erroneous assignment messages (rejected by the input pre-processing task). The DCME receive unit shall recover from an abnormal assignment transition within a maximum of one assignment channel refresh cycle.

7.1.4 *ADPCM decoder connection control*

The following ADPCM decoder control rules shall apply only if the remote IT number is destined for the DCME receive unit:

a) When a new assignment of a previously disconnected IT is made, an ADPCM decoder shall be connected to the corresponding local IT, thus completing the BC to ADPCM decoder to IT path through the DCME receive unit.

b) When a reassignment of a previously connected IT is made to a different BC, the ADPCM decoder currently associated with the corresponding local IT shall be maintained.

c) Whenever an IT connection changes to BC number 0 (explicit disconnection) or the IT is implicitly disconnected, the associated ADPCM decoder shall be disconnected.

d) The ADPCM decoder shall be reset when a new IT connection is established.

7.1.5 *Bit bank handling and overload channel derivation*

The rules for bit bank handling and overload channel derivation shall be the same as those defined for the TCP function. The only differences are that when an assignment message is erroneous (or lost):

1) the pointer variables Pv and Pov shall be set to 0;

2) if there is not enough bit capacity available to service the corrupted assignment message then the affected channels shall receive dummy bits set to 0;

3) the variable N4 (number of 4 bit overload channels) shall be set to 0 if its calculated value is negative;
4) the variable N3 (number of 3 bit overload channels) if used, shall be set to 0 if its calculated value is negative.

7.1.6 *Connectivity implementation delay*

The BC to ADPCM decoder to IT connection is implemented at the beginning of the DCME frame which occurs three frames after the start of the DCME frame containing the applicable assignment message. Refer to Figure 14/G.763.

7.1.7 *TCP and TCH interactions*

The following indications are sent to the TCP and the TCH in response to certain transitions indicated by the assignment message.

Rxdata(IT): This indication shall be sent to the TCP function when an assignment message is received which indicates a transition from a previous BC type to a data BC type. (IT is the transmit IT number.)

RxTranspreq(IT): This indication shall be sent to the TCH when an assignment message is received which indicates a transition from another BC type to a transparent BC type.

RxTransprel(IT): This indication shall be sent to the TCH when an assignment message is received which indicates a transition from a transparent BC type to some other BC type.

8 On-demand 64 kbit/s circuit handling

8.1 *Overview of establishment and disestablishment of 64 kbit/s unrestricted class connections (transparent circuits)*

The DCME shall establish/disestablish 64 kbit/s unrestricted duplex connections under control of the seizing/releasing ISC as part of the call setup/clearing process between exchanges. Dedicated seizure/select and release messages and the associated acknowledgement messages are exchanged between the DCME and the ISC as defined in Recommendation Q.50. The DCME-to-ISC control interface is defined in § 5.3.

Subject to the capability of the ISC, this process is usable for performing the in-call modifications between the DCMEs during alternate speech/64 kbit/s unrestricted type calls.

Upon reception of a seizure/select message from the ISC for a trunk, the DCME shall perform the necessary internal checks, including the 64 kbit/s unrestricted dynamic load control status, for the accommodation of this call and an acknowledgement (positive or negative) message shall be returned as soon as possible to the calling ISC. The calling end DCME shall initiate the establishment of the unrestricted 64 kbit/s forward connection to the called end DCME using a special identifier in the assignment message. The called end DCME, upon receipt of this message, shall automatically initiate the establishment of the unrestricted 64 kbit/s return connection. Failure to complete the establishment of a 64 kbit/s circuit between DCMEs shall be reported to the ISC as soon as this has been detected internally. This reporting shall be in the form of an out-of-service message.

Upon receipt of a release message from the calling ISC the releasing end DCME shall initiate the disestablishment of the unrestricted 64 kbit/s forward connection, and the opposite end DCME shall automatically initiate the disestablishment of the unrestricted 64 kbit/s return connection. Upon completion of this process, a positive release acknowledgement message shall be returned to the releasing ISC. Failure to complete the disestablishment shall be reported to the releasing ISC using the out-of-service message and the DCME shall put the trunk in a blocked condition.

After manual or automatic removal of any failure condition, the DCME shall put the trunk in an idle condition and send a back-in-service message to the ISC.

A calling end DCME shall detect a release initiated by the opposite end (non-controlling) ISC by the reception of a disconnect message in the control channel. This abnormal release shall be recognized as a dual seizure situation being resolved between the ISCs. The detecting DCME shall first complete the release normally and immediately attempt to re-establish the released 64 kbit/s unrestricted duplex connection between the DCMEs.

8.2 *Transparent circuit handler (TCH)*

The TCH function interacts with the switching centre interface (SCI), the DLC, the TCP and the RCP functions. The TCH function is invoked for the execution of peer procedures in correspondent DCMEs for 64 kbit/s circuit handling.

A facility by which the operator may enable or disable the interaction between the TCH and DLC functions shall be provided (see Note).

Note — Disabling of the TCH/DLC interaction may degrade voice performance under high load conditions.

The functional partitioning of processing functions is intended to add clarity to the requirements of the TCH function and not to specify processing partitions within a DCME implementation.

The end-to-end on-demand circuit establishment and on-demand circuit disestablishment procedures have the following salient features:

a) The generation of a positive acknowledge for a seizure/select request which is sent to the ISC when the 64 kbit/s circuit establishment process between DCMEs has been properly initiated.

b) Circuit handling protocols for dual seizures between DCMEs. These protocols are compatible with the procedures for dual seizures in ISCs as defined in Recommendation Q.764 (signalling procedures of Signalling System No. 7 ISUP).

c) Automatic recovery of the circuit handling process following unsuccessful or delayed completion of circuit establishment.

d) Automatic circuit blocking (for 64 kbit/s calls) after unsuccessful two-way disconnection.

The block interaction diagram for the TCH function is shown in Figure 15/G.763.

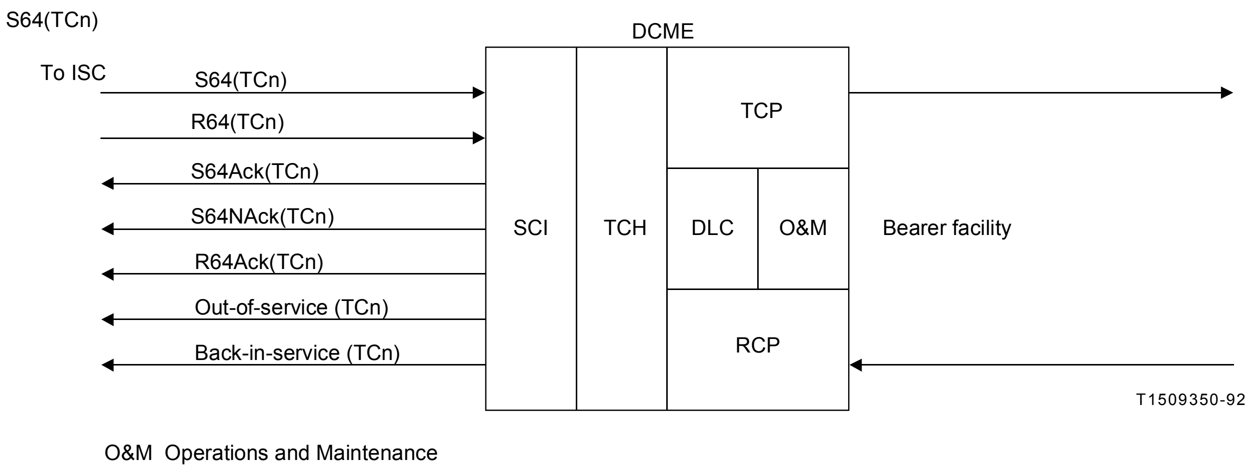


FIGURE 15/G.763

Block interaction diagram for 64 kbit/s circuit handling

The TCH receives 64 kbit/s seizure/select messages and 64 kbit/s release messages from the local ISC (through the SCI), receive transparent request and receive transparent release indications from the RCP function and 64 kbit/s dynamic load control indications from the local DLC function.

The TCH send 64 kbit/s acknowledged and not acknowledged messages, out-of-service and back-in-service messages to the local ISC (through the SCI) and sends transparent request and transparent release indications to the TCP. Table 4/G.763 gives nine different TCH states.

TABLE 4/G.763

List of states of the transparent circuit handler

00	Not-64
01	Established-forward-64
02	Disestablished-forward-64
03	Disestablished-backward-64
04	Auto-recovery-64
05	Connect-calling-64
06	Connect-called-64
07	Out-of-service
08	Blocked (DLC64)
09	Spurious recovery

Four timers in the TCH define time intervals within which circuit establishment, disestablishment, and auto-recovery procedures are to be completed successfully.

T1: Maximum time allowed for successful completion of 64 kbit/s circuit establishment, 1.0 s.

T2: Maximum time allowed for successful completion of 64 kbit/s circuit disestablishment, 1.5 s.

T3: Time assumed for completion of 64 kbit/s circuit disestablishment remotely initiated, 1.0 s.

T4: Maximum time allowed for successful completion of 64 kbit/s circuit auto-recovery, 1.5 s.

8.2.1 *External information elements*

The provision of the signalling system between the DCME and the local ISC, specified in Recommendation Q.50, will ensure the availability of the following external information elements for on-demand 64 kbit/s circuit handling. Depending on the characteristics of the chosen DCME-ISC control system, all of the required external information elements may not be used.

8.2.1.1 *S64(TCn)*

Request for the establishment of a 64 kbit/s circuit on local TCn received from the ISC.

8.2.1.2 *R64(TCn)*

Request for the disestablishment of a 64 kbit/s circuit on local TCn received from the ISC.

8.2.1.3 *S64Ack(TCn)*

Acknowledgement sent to the ISC that the establishment of a 64 kbit/s circuit for TCn has been initiated.

8.2.1.4 *S64NAck(TCn)*

Negative acknowledgement sent to the ISC that the request for establishment of a 64 kbit/s circuit for TCn has been rejected.

8.2.1.5 *R64Ack(TCn)*

Acknowledgement sent to the ISC that the disestablishment of a 64 kbit/s circuit for TCn has been completed.

8.2.1.6 *Out-of-service (TCn)*

Indication sent to the ISC that TCn is out-of-service.

8.2.1.7 *Back-in-service (TCn)*

Indication sent to the ISC that TCn is back-in-service.

8.2.2 *DLC information elements*

The indications received from the DLC function are as follows:

8.2.2.1 *DD64*

Indication received from the DLC function when 64 kbit/s capacity is available locally and at the correspondent DCME. Refer to § 9.4.

8.2.2.2 *AD64*

Indication received from the DLC function when 64 kbit/s capacity is not available locally or at the correspondent DCME. Refer to § 9.4.

8.2.3 *Other information elements*

The indications sent to the TCP function and received from the RCP function are as follows:

8.2.3.1 *Transpreq(ITn)*

Indication sent to the TCP to initiate the assignment of a 64 kbit/s forward channel for ITn.

8.2.3.2 *Transprel(ITn)*

Indication sent to the TCP to initiate the disconnection of 64 kbit/s forward channel for ITn.

8.2.3.3 *RxTranspreq(ITn)*

Indication received from the RCP to indicate that a 64 kbit/s connection has been established.

8.2.3.4 *RxTransprel(ITn)*

Indication received from the RCP to indicate that a 64 kbit/s connection has been released.

8.3 *On-demand circuit establishment*

All procedures described for the on-demand circuit establishment pertain to a single trunk (circuit) denoted by TCn, and to the associated forward and return intermediate trunks ITn and ITn′, respectively.

8.3.1 *Normal circuit establishment*

The sequence chart for a normal 64 kbit/s circuit establishment cycle is shown in Figure 16/G.763.

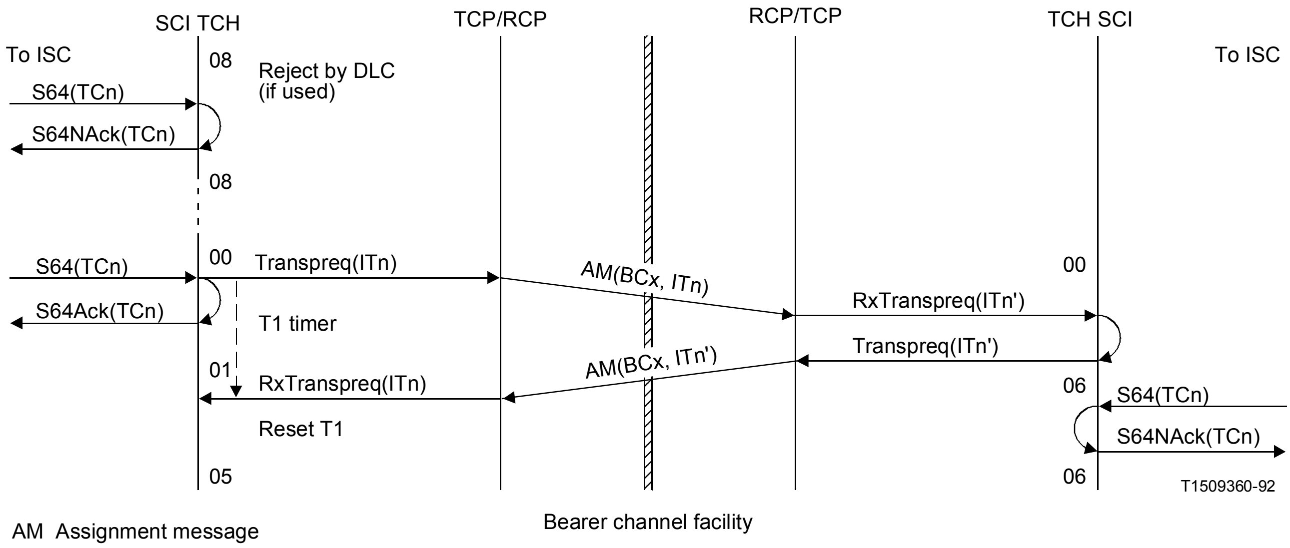


FIGURE 16/G.763

Sequence chart for normal 64 kbit/s circuit establishment

8.3.1.1 *Actions required at the calling end SCI/TCH*

Upon reception of the external information element S64 for TCn from the ISC, the SCI shall send S64NAck(TCn) to the ISC if the TCH is in the process of disestablishing TCn from a previous call (see § 8.4.1.4), or if the DLC ON condition (AD64 has been received) is in effect (provided that the internal interaction with the DLC process is enabled), or if the TCH is in the connect-called-64 state. No further action shall be taken after sending S64NAck(TCn).

If the internal DLC condition (if used) is OFF (DD64 has been received), and if the TCH is not in the process of disestablishing ITn from a previous call, the SCI shall send S64Ack(TCn) to the ISC and the TCH shall:

a) send Transpreq(ITn) to the TCP;

b) start timer T1. A subsequent reception of the RxTranspreq(ITn) indication shall signify the completion of the circuit establishment and shall cause the TCH to reset timer T1 and to enter the connect-calling-64 state for the circuit using ITn. The expiration of timer T1 is described in § 8.3.2.1.

8.3.1.2 *Actions required at the calling end TCP/RCP*

The reception of the Transpreq(ITn) indication from the local TCH shall cause the TCP to perform a transmit assignment (BCx, ITn) in accordance with § 6 for the forward link of the circuit being established.

Reception of the new assignment message (BCx, ITn′) shall cause the RCP to establish the receive connection for the return in accordance with § 7 and to send the RxTranspreq(ITn) indication to the TCH.

Actions required on failure to receive the expected RxTranspreq(ITn) indication for the return link are described in § 8.3.2.2.

8.3.1.3 *Actions required at the called end RCP/TCP*

Reception of a new assignment message (BCx, ITn) from the distant (calling) DCME shall cause the RCP to establish the receive side connection in accordance with § 7 and send the RxTranspreq(ITn) indication to the TCH.

Reception of the Transpreq(ITn) indication shall cause the TCP to perform a transmit assignment (BCx, ITn′) in accordance with § 6 for the return link of the circuit being established.

8.3.1.4 *Actions required at the called end TCH*

Reception of the RxTranspreq(ITn′) indication shall cause the TCH to initiate the 64 kbit/s transparent channel establishment in the return direction by sending a Transpreq(ITn′) indication to the TCP and to enter the connect-called-64 state for the circuit using ITn ′.

8.3.2 *Unsuccessful circuit establishment*

The sequence chart for an automatic recovery after an unsuccessful circuit establishment is shown in Figure 17/G.763.

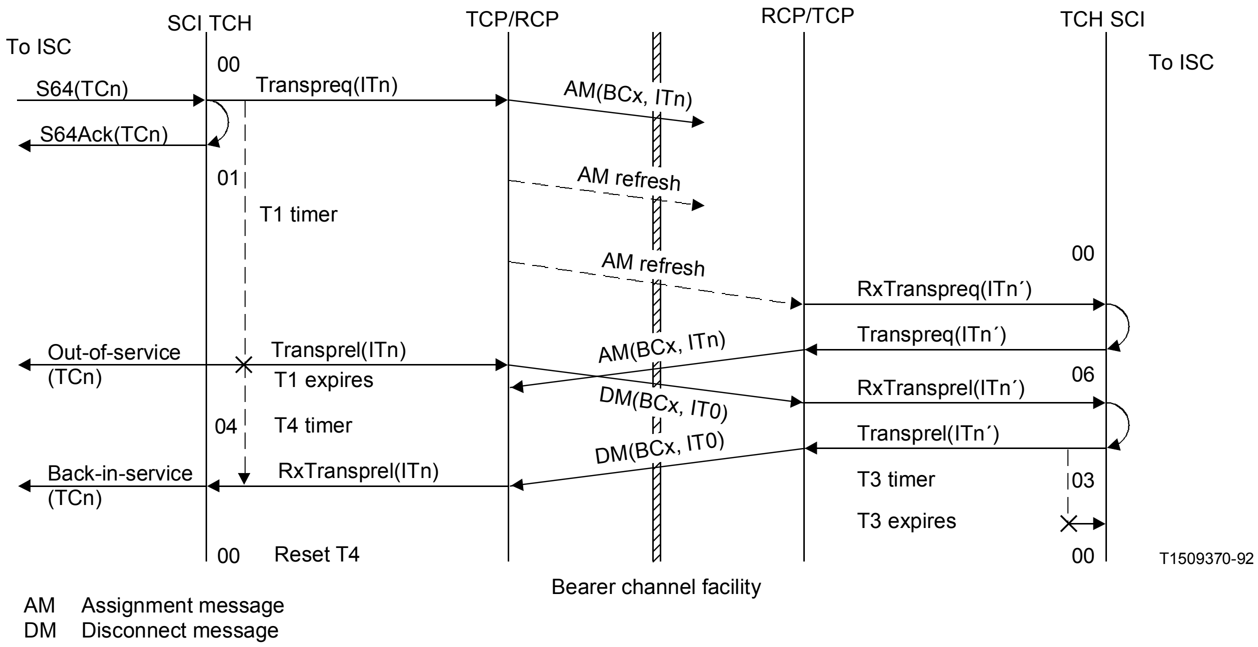


FIGURE 17/G.763

Sequence chart for automatic recovery after unsuccessful (delayed) 64 kbit/s circuit establishment

8.3.2.1 *Actions required at the calling end SCI/TCH*

If the RxTranspreq(ITn) indication is not received before the expiration of timer T1, the following automatic recovery procedure shall be initiated.

The TCH shall:

a) send a Transprel(ITn) indication to the TCP;

b) start timer T4. The subsequent reception of a RxTransprel(ITn) indication shall signify the completion of the circuit disestablishment and shall cause the TCH to reset timer T4 and to enter the appropriate state for the circuit using ITn. The expiration of timer T4 is described in § 8.3.2.3.

The SCI shall:

a) send an out-of-service (TCn) indication to the ISC;

b) send a back-in-service (TCn) message to the ISC when the TCH has indicated the reception of RxTransprel(ITn) from the local RCP.

8.3.2.2 *Actions required at the calling end TCP/RCP*

Reception of the Transprel(ITn) indication shall cause the TCP to perform a disconnection (BCx, IT0) in accordance with § 6 for the forward link of the unsuccessfully (or delayed) established circuit.

If the expected new assignment message (BCx, ITn′) for the return direction is received while the above circuit disconnection is in progress, the RCP in the calling DCME shall first establish the receive side connection normally by executing the actions described in § 8.3.1.2, § 2, and then complete the normal disconnection process by executing the actions described in § 8.4.1.2, § 2.

8.3.2.3 *Unsuccessful automatic recovery*

If the RxTransprel(ITn) from the RCP is not received before the expiration of timer T4, the SCI shall block circuit TCn and raise a blocking alarm for this circuit. The SCI shall be reset to the appropriate state for the circuit using TCn only after the operator's attendance to the blocking alarm. Upon reset of the SCI, a back-in-service (TCn) shall be sent to the ISC.

8.4 *On-demand circuit disestablishment*

All procedures described for the on-demand circuit disestablishment pertain to a single trunk (circuit) denoted by TCn, and to the associated forward and return intermediate trunks ITn and ITn′, respectively.

8.4.1 *Normal circuit disestablishment*

The sequence chart for a normal 64 kbit/s circuit disestablishment cycle is shown in Figure 18/G.763.

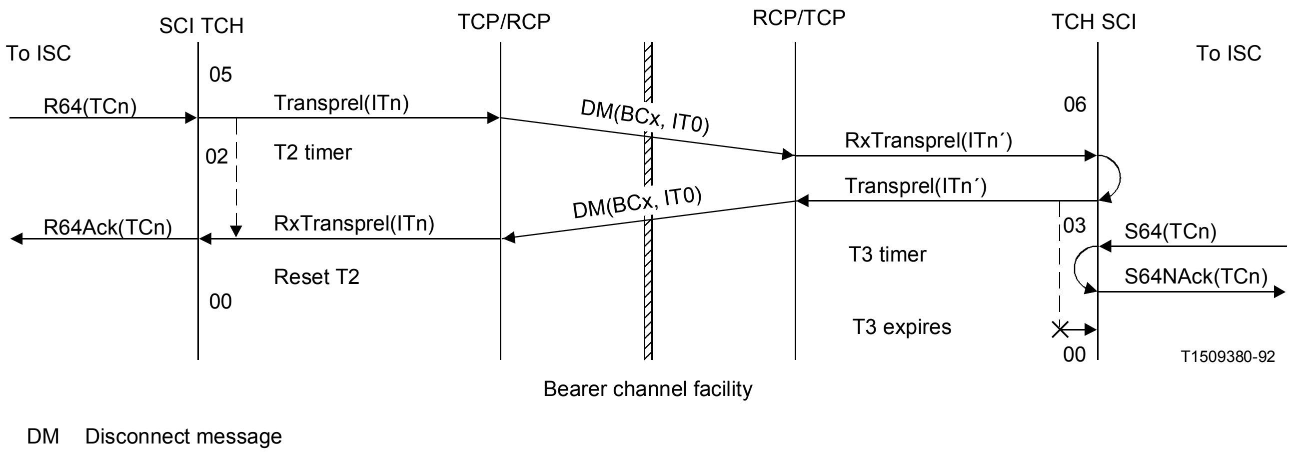


FIGURE 18/G.763

Sequence chart for normal 64 kbit/s circuit disestablishment

8.4.1.1 *Actions required at the releasing end SCI*

Upon reception of the external information element R64 for TCn at the releasing end SCI, the TCH shall:

a) send Transprel(ITn) to the TCP;

b) start timer T2. A subsequent reception of RxTransprel(ITn) indicates completion of the circuit disestablishment and shall cause the TCH to reset timer T2 and enter the appropriate state for the circuit using ITn. The expiration of timer T2 is described in § 8.4.2.1,

and the SCI shall send an R64Ack(TCn) indication to the ISC when the TCH has indicated the reception of RxTransprel(ITn) from the local RCP.

8.4.1.2 *Actions required at the releasing end TCP/RCP*

The reception of Transprel(ITn) from the local TCH shall cause the TCP to perform a disconnection (BCx, IT0) in accordance with § 6 for the forward link of the circuit.

Reception of the disconnection message (BCx, IT0) or an implied disconnection shall cause the RCP to perform a receive disconnection for the return ITn′ in accordance with § 7 and to send the RxTransprel(ITn) signal to the TCH.

8.4.1.3 *Actions required at the released end RCP/TCP*

Reception of the disconnect message (BCx, IT0), or alternatively an implied disconnect from the distant (releasing) DCME shall cause the RCP to disconnect the receive side connection in accordance with § 7 and send the internal signal RxTransprel(ITn') to the TCH.

Reception of the Transprel(ITn′) signal shall cause the TCP to perform a disconnection (BCx, IT0) in accordance with § 6 for the return link of the circuit.

8.4.1.4 *Actions required at the released end TCH/SCI*

Reception of RxTransprel(ITn′) from the RCP shall cause the TCH to initiate the release of the 64 kbit/s transparent channel in the return direction by sending Transprel(ITn′) to the TCP, and to start a timer T3 (expiration of timer T3 assumes normal completion of the disconnection process initiated by the distant end).

Prior to timer T3 expiration, any reception of S64 for the same TCn from the local ISC shall be negatively acknowledged by forwarding S64NAck(TCn) to the ISC.

If the RxTranspreq(ITn′) signal followed by a RxTransprel(ITn′) signal are received prior to timer T3 expiration, a spurious disconnection condition shall be declared and the actions described in § 8.6.2 shall be taken.

Upon expiration of T3, the TCH shall enter the appropriate state for circuit ITn′.

8.4.2 *Unsuccessful circuit disestablishment*

The sequence chart for an unsuccessful 64 kbit/s circuit disestablishment cycle is shown in Figure 19/G.763.

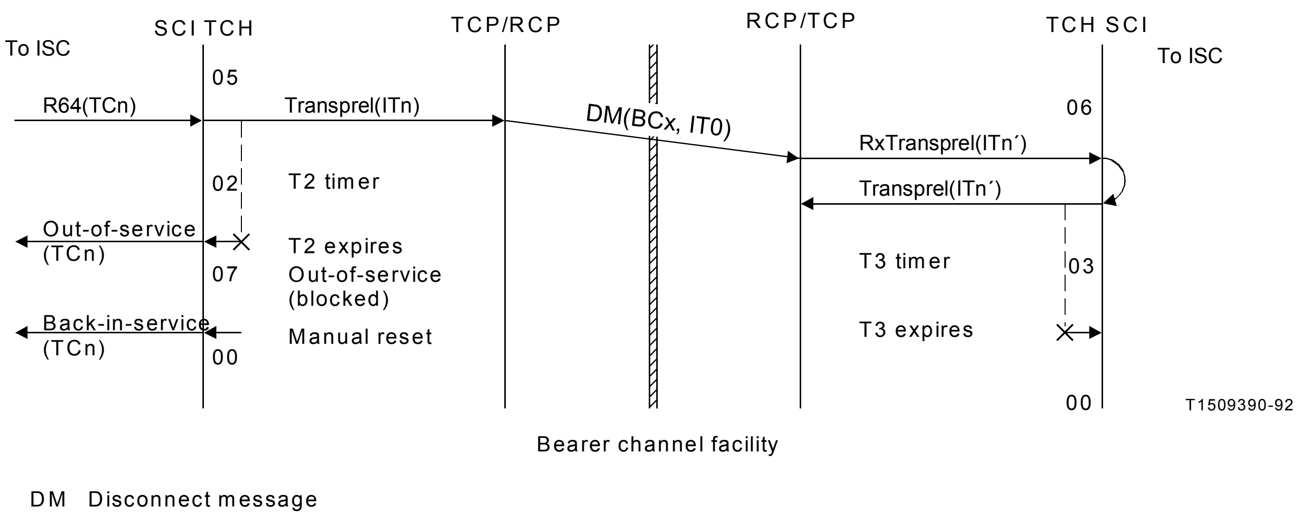


FIGURE 19/G.763

Sequence chart for unsuccessful circuit disestablishment
(one-sided circuit blocking)

8.4.2.1 *Actions required at the releasing end TCH*

If the RxTransprel(ITn) from the RCP is not received before the expiration of timer T2, the TCH shall block circuit ITn and raise a blocking alarm for this circuit. The SCI shall send the out-of-service (TCn) message to the ISC. The TCH shall be reset to the appropriate state for the circuit using ITn only after the operator's attendance to the blocking alarm. Upon reset of the TCH, the SCI shall send a back-in-service (TCn) message to the ISC.

8.5 *Dual seizure handling*

All procedures described for the dual seizure handling pertain to a single trunk (circuit) denoted by TCn, and to the associated forward and return intermediate trunks ITn and ITn′, respectively.

8.5.1 *Dual seizure condition*

Simultaneous reception of seizure requests S64 for TCn from both ISCs will cause the procedures described in §§ 8.3.1.1 and 8.3.1.2 to be invoked from each end of the circuit. The condition after execution of those procedures would be the connect-calling-64 state of the TCHs at both ends for the same circuit. Refer to Figure 20/G.763.

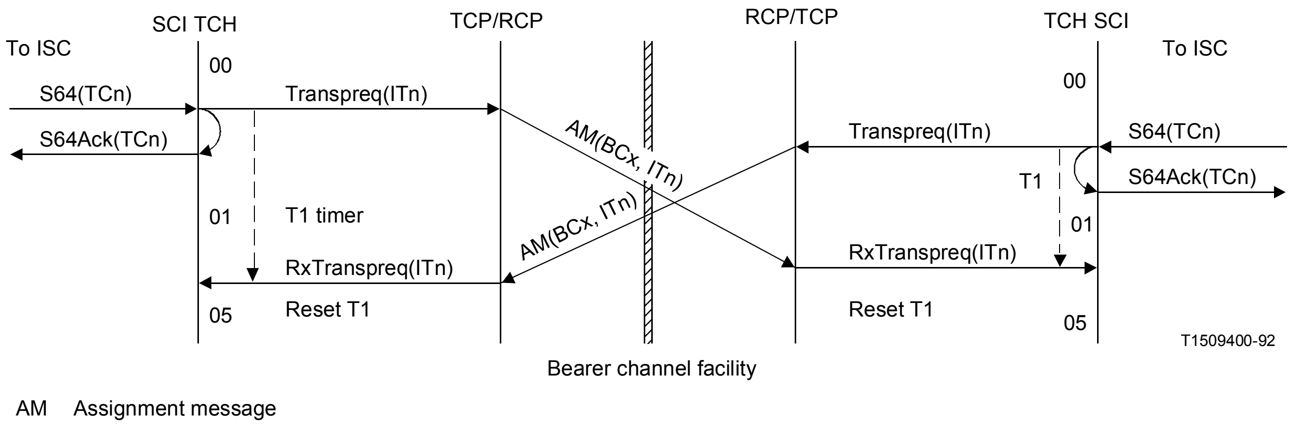


FIGURE 20/G.763

Sequence chart for 64 kbit/s circuit establishment in dual-seizure situation

8.5.2 *Dual seizure resolution*

For explanation in this section, the non-controlling ISC is assumed to be at the ITn′ side. Refer to Figure 21/G.763.

8.5.2.1 *Actions required at the TCH (*non-controlling switching centre end)

Upon reception of the external information element R64 (TCn) from the non-controlling ISC, the TCH shall initiate the normal circuit disestablishment procedures described in §§ 8.4.1.1, 8.4.1.2 and 8.4.1.3.

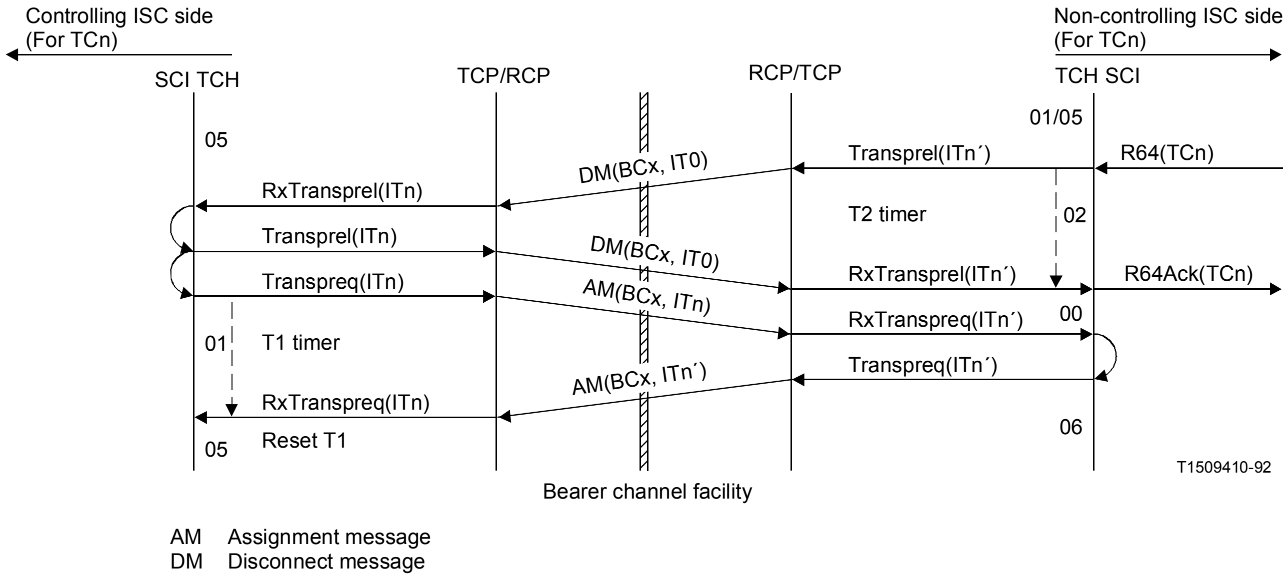


FIGURE 21/G.763

Sequence chart for 64 kbit/s circuit re-establishment after dual-seizure detection/resolution

8.5.2.2 *Actions required at the TCH* (controlling switching centre end)

Upon reception of RxTransprel(ITn) the TCH shall respond by sending Transprel(ITn) to the TCP. The TCP shall thereafter immediately initiate the automatic re-establishment of the circuit by sending Transpreq(ITn) to the TCP and to start timer T1. All subsequent procedures described in §§ 8.3.1.2, 8.3.1.3 and 8.3.1.4 shall proceed normally (including procedures for auto-recovery described in § 8.3.2 in case of unsuccessful circuit re-establishment).

8.6 *Spurious disconnect handling*

All procedures described for the spurious disconnect handling pertain to a single trunk (circuit) denoted by TCn, and to the associated forward and return intermediate trunks ITn and ITn′, respectively.

8.6.1 *Spurious disconnect conditions*

Condition I – A spurious disconnect message or a spurious implied disconnect detected by the called end RCP while the called end TCH is in the connect-called-64 state will cause the procedures described in §§ 8.4.1.3 and 8.4.1.4 to be invoked. A subsequent assignment message refresh will result in the generation of a RxTranspreq(ITn′) signal to the called end TCH after timer T3 has started. Refer to Figure 22/G.763.

Condition II – A spurious disconnect message or a spurious implied disconnect detected by the calling end RCP while the calling end TCH is in the connect-calling-64 state will cause the procedures described in § 8.5.2.2 to be invoked. The resulting disconnect message and the subsequent re-establishing assignment message will be recognized as spurious disconnect condition I. Refer to Figure 23/G.763.

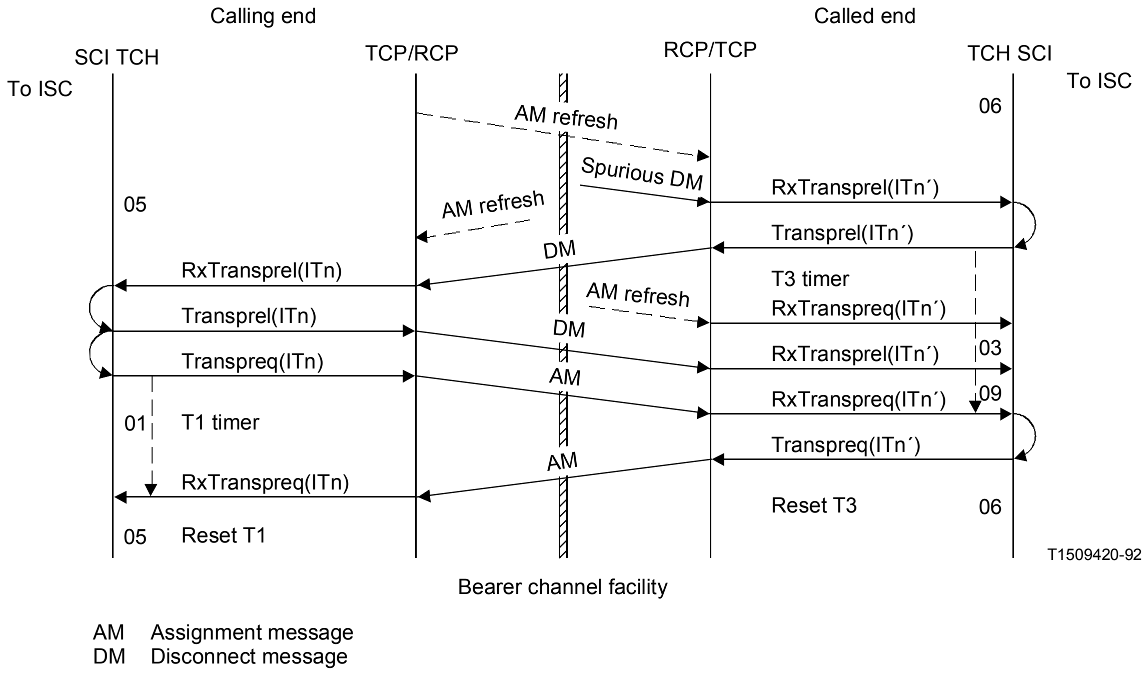


FIGURE 22/G.763

Sequence chart for 64 kbit/s circuit re-establishment after a spurious disconnect into called end

8.6.2 *Spurious disconnect recovery*

8.6.2.1 *Actions required at the called-end TCH*

After timer T3 has started, upon reception of a RxTranspreq(ITn′) signal, followed by a RxTransprel(ITn′) signal, the TCH shall enter the spurious-recovery state for the circuit using ITn′. A subsequent reception of the internal message RxTranspreq(ITn′) shall cause the TCH to reset timer T3, to initiate the re-establishment of the 64 kbit/s transparent channel in the return direction by sending Transpreq(ITn′) to the TCP, and to re-enter the connect-called-64 state for the circuit using ITn′.

If timer T3 expires before the reception of the RxTranspreq(ITn′) signal, the TCH shall enter the appropriate state for circuit ITn′.

8.6.2.2 *Actions required at the calling end TCH*

For condition I, the actions described in § 8.5.2.2 shall be taken once.

For condition II, the actions described in § 8.5.2.2 shall be taken twice.

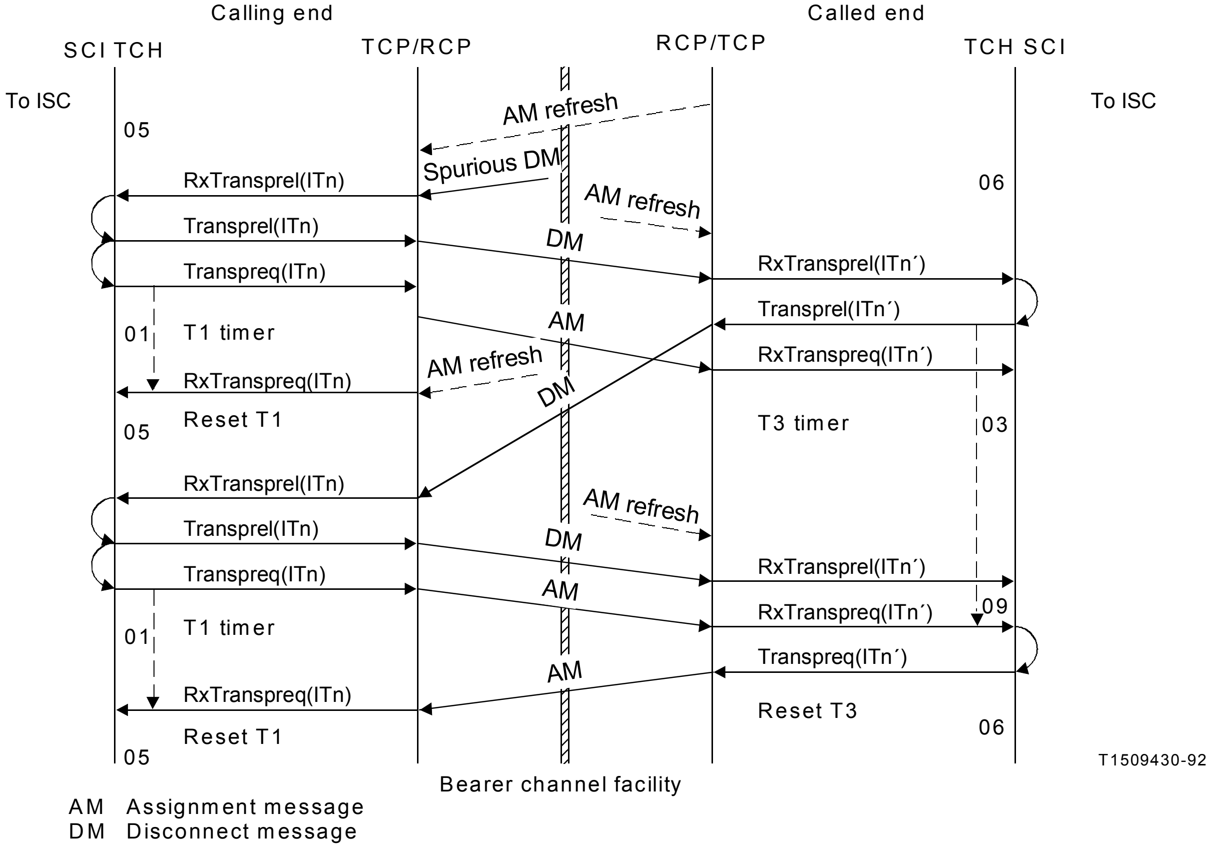


FIGURE 23/G.763

Sequence chart for 64 kbit/s circuit re-establishment after a spurious disconnect into calling end

9 Dynamic load control

9.1 *Overview*

To reduce the probability of excessive freeze-out percentages, the DCME shall have a facility for DLC. This facility signals towards the local and distant-end ISCs that new calls should not be established. The DCME shall communicate with the ISC in accordance with Recommendation Q.50. A DLC facility shall be provided for voice/voice-band data and on-demand 64 kbit/s traffic separately. DLC shall not be applied to pre-assigned traffic.

The DCME shall provide a DLC signal which may be sent to the local and distant ISCs to limit the traffic load presented to the DCME during overload conditions. Overload conditions should be indicated by average number of bits per sample calculated for each pool. When the value falls below a particular previously set threshold level, the DLC message should be generated at the DCME. DLC messages shall be sent back to the local ISC(s), and the distant DCME shall be informed through the control channel. The distant DCME shall interpret and appropriately convey the DLC information to its associated ISC(s).

9.1.1 *DLC activation/deactivation criteria*

Voice/voice-band data DLC activation messages shall be generated when the number of bits per sample, averaged over the voice channels of the pool, drops below a presettable threshold.

The 64 kbit/s unrestricted (transparent) DLC activation messages shall be generated when:

a) the measured number of assigned 64 kbit/s unrestricted channels exceeds a presettable threshold; or

b) the voice/voice-band data DLC has been activated; or

c) the voice/voice-band data DLC is expected to be activated due to an increase of one additional channel in the 64 kbit/s unrestricted traffic loading.

Activation of DLC shall occur immediately upon transgressing the threshold criteria. Deactivation of DLC shall occur when the average number of bits per sample exceeds a presettable threshold or the number of 64 kbit/s unrestricted channels falls below a presettable threshold. If the 64 kbit/s DLC is not active, 64 kbit/s unrestricted channel requests shall not be denied. Deactivation of DLC shall not occur earlier than a programmable interval which has a minimum of 10 s in order to prevent an oscillating condition.

9.1.2 *Message processing and routing*

Internal DLC indications are sent on a destination selective basis to the local SCI and the TCH for further processing and subsequent forwarding of the associated bearer service related external information elements to the ISC(s) according to §§ 9.3.2 and 9.4.2. The list of DLC related external and internal messages used by the SCI is given in § 5.3.2. The external message exchange between the SCI and the ISC is as defined in Recommendation Q.50.

Assuming that the ISC is responding to messages originating from the DCME SCI, it is recommended that once a DLC signal has been active (i.e. new calls blocked) and then returns to an inactive state (new calls may be established), the affected circuits be unblocked in a gradual manner for the relevant bearer service type. Correspondent DCMEs shall exchange their respective load conditions by means of the DLC support messages within the control channel asynchronous data word. Refer to § 11.3.3.2.

9.2 *Load condition calculation* (see Note)

The local loading condition shall be determined using the average number of encoding bits per sample as a measure. An example of a DLC double averaging technique is given in Annex B.1.

Note – The load calculation may be used for provision of special tandeming facilities (under study).

9.3 *Voice/voice-band data DLC*

9.3.1 *DCME function*

Two load conditions are defined:

a) *High Load (HL)* – In this condition, the measured average number of encoding bits is less than the high load threshold (e.g. 3.6 bits per sample).

b) *Low Load (LL)* – In this condition, the measured average number of encoding bits is greater than the low load threshold (e.g. 3.9 bits per sample).

The HL and LL thresholds shall be operator programmable options settable between 3 and 4 in 0.05 bits per sample steps.

When the average number of encoding bits is between the two thresholds, the last load condition shall be maintained.

A local HL condition shall be signalled to the corresponding DCME(s) by setting the voice/voice-band data DLC support message (bit p) to state 1. Refer to § 11.3.

A local LL condition shall be signalled to the corresponding DCME(s) by setting the voice/voice-band data DLC support message (bit p) to state 0. Refer to § 11.3.

The DLC ON condition for voice/voice-band data traffic shall be declared when:

a) the HL condition is detected locally; or

b) the bit p received from a corresponding DCME is in state 1 (DLC is applicable only to those circuits which are destined to this corresponding DCME).

The DLC OFF condition for voice/voice-band data traffic shall be declared for each destination when:

a) the LL condition is detected locally; and

b) the bit p received from the relevant destination is in state 0.

The DLC ON condition shall be declared during a system reconfiguration.

The ADVD indication (see § 5.3.2) shall be sent to the local SCI at the transition from DLC OFF to DLC ON. The DDVD indication shall be sent to the local SCI at the transition from DLC ON to DLC OFF.

9.3.2 *SCI function*

The SCI shall send the information elements SNA and VDNA to the ISCs when the TCH receives an ADVD indication from the DLC function.

When the TCH receives a DDVD indication from the DLC function, the SCI shall send the information elements SA and VDA to the ISC(s), unless an ADVD indication recurs within Ta seconds after the last detected ADVD indication.

The Ta timer shall be operator selectable with a minimum of ten seconds.

Depending on the characteristics of the chosen DCME - ISC control system, the SNA, VDNA, SA and VDA information elements may not all be used.

9.4 *On-demand 64 kbit/s DLC*

9.4.1 *DCME function*

The availability of capacity for on-demand 64 kbit/s traffic is based on the predicted average number of encoding bits for voice traffic if a pair of 4 bit bearer time slots currently used for voice traffic were to be used to accommodate one additional 64 kbit/s channel.

Two capacity availability conditions are defined:

a) *Capacity Available (UCA)* – In this condition, the predicted average number of encoding bits is greater than the LL threshold defined in § 9.3.1.

b) *Capacity Not Available (UCNA)* – In this condition, the predicted average number of encoding bits is less than the HL threshold defined in § 9.3.1.

When the predicted average number of encoding bits is between the two thresholds, the last load condition shall be maintained.

A local UCNA condition shall be signalled to the corresponding DCME(s) by setting the on-demand 64 kbit/s DLC support message (bit q) to state 1. Refer to § 11.3.

A local UCA condition shall be signalled to the corresponding DCME(s) by setting the on-demand 64 kbit/s DLC support message (bit q) to state 0. Refer to § 11.3.

The DLC ON condition for on-demand 64 kbit/s traffic shall be declared when:

a) the UCNA condition is detected locally; or

b) the bit q received from a corresponding DCME is in state 1 (DLC is applicable to those circuits which are destined to this corresponding DCME).

The DLC OFF condition for on-demand 64 kbit/s traffic shall be declared for each destination when:

a) the UCA condition is detected locally; and

b) the bit q received from the relevant destination is in state 0.

The DLC ON condition shall be declared during a system reconfiguration.

The AD64 message shall be sent to the local SCI and to the TCH at the transition from DLC OFF to DLC ON.

The DD64 message shall be sent to the local SCI and to the TCH at the transition from DLC ON to DLC OFF. A facility to enable or disable the DLC/TCH interaction shall be provided. Refer to § 8.2.

9.4.2 *SCI function*

The SCI shall send the information element UCNA to the ISC(s) when the TCH receives an AD64 indication.

When the TCH receives a DD64 indication, the SCI shall send the information element UCA to the ISC(s), unless the TCH receives an AD64 indication within Tb seconds after the last detected AD64 indication.

The Tb timer shall be operator selectable with a minimum of ten seconds.

Depending on the characteristics of the chosen ISC control system, the UCNA and UCA information elements may not be used.

10 Test procedures

A means of verifying end-to-end continuity and correct assignment of channels must be provided. If an automatic procedure is implemented then it should conform to the following:

Note – The channel check procedure is applied independently to each pool.

10.1 *Channel check procedure*

10.1.1 *Test procedure*

A repetitive 20 second Test time frame (TTF) shall be established. At the start of each TTF, unless the procedure is inhibited, a test vector bit pattern sequence shall be originated on IT 239 for pool 1 and IT 240 for pool 2. This test vector sequence shall compete for assignment to a bearer channel in accordance with § 6. The ADPCM encoder for (BCn, IT 239/240) shall be selected normally in accordance with § 6, except that any ADPCM encoders selected for IT 239/240 shall always operate in A-law mode. The characteristics of the test vector sequence shall be in accordance with § 10.1.4. The test vector sequence shall remain ON for approximately one second. A DCME transmit unit shall generate one channel check test vector which shall be processed by all corresponding DCME receive units. For this reason, IT 239/240 shall be assumed to be destination directed to all destinations corresponding with the DCME transmit unit.

To inform the corresponding DCME receiver units that a channel check has commenced, a code 1111 is transmitted in the synchronous data word in the same DCME frame as the first associated channel check assignment message (BCn, IT 239/240) for each TTF. The synchronous data word code 1111 shall be transmitted once for each channel check sequence.

If the channel check procedure has been manually inhibited at the DCME transmit unit, bit 1 in DCME frame 62 of the Asynchronous data word shall be set to 1, otherwise the bit shall be set to 0. The manual inhibit shall become effective at the next TTF boundary.

All corresponding DCME receive units shall assign IT 239/240 to a special test port. A special test port is assigned for each received bearer. The special test ports are identified by the local IT numbers 241 through 244 receiving bearer numbers 1 through 4, respectively. A test ADPCM decoder associated with (BCn, IT 239/240) shall be selected in accordance with § 7. However, any ADPCM decoder selected for IT 239/240 shall operate in A-law mode. Continuous correlation shall be performed to identify the presence of the test vector. When the test vector is identified, a test pattern receiver shall determine the accuracy of the match between the received test vector and a locally stored version of this pattern in accordance with § 10.1.4. For each bearer, the result from the test pattern receiver shall be disregarded if the continuous BER measurement declares a high BER condition. Refer to §15.10.1.

10.1.2 *Reporting test results (remote DCME)*

The remote DCME shall generate a local alarm when the test vector pattern is not correctly received in accordance with § 10.1.4, or if the code 1111 is received from synchronous data word and a test vector has not been synchronized on the corresponding test port.

The remote DCME shall construct and maintain a table of test results of each BC. Separate test result tables shall be maintained for each incoming bearer and/or pool. For each BC, an entry in the table shall contain a 0 if the test result is pass, otherwise the test result table shall contain a 1. If the result of the test pattern receiver has been disregarded, a 1 shall be entered in the test result table for the high BER yes/no condition and a 1 shall be assumed for the pass/fail entry. The test result table shall also include the identity of the ADPCM decoder currently assigned to the test port.

It is recommended that the test result table also contain a real-time clock and date entry showing the time and date that the last test result was obtained for each BC. It is further recommended that the result tables be made accessible by the local operations and maintenance function or an equivalent facility.

For each bearer, the remote DCME shall send the result of the last channel check to the corresponding local DCME via the Asynchronous Data Word using the format shown in Table 5/G.763. A test result consisting of a BC number, pass/fail condition, high BER yes/no condition and ADPCM decoder number shall be sent once per DCME multiframe in DCME frames 56-61. The test results are sent in ascending numerical order of the incoming bearer number.

If no test result exists, if the automatic procedure has not been implemented, or if more than 60 seconds have elapsed since the last channel check test for that bearer, the BC number and the ADPCM decoder number contained in DCME frames 57, 58, 59 and 60, respectively, shall be set to all 1s (ineffective message). The pass/fail and high BER bits shall be set to 1. The message contents shall remain latched to the last result for that bearer until a new result is available.

TABLE 5/G.763

4 bit asynchronous data word bit allocation

DCME frame	Data word bit No.				Message (s = spare bit set to 0)
	1	2	3	4	
0 1 53	1 5 213	2 6 214	3 7 215	4 8 216	Type: IT-related circuit supervision/alarm condition Designation: The No. represents the IT No. Content: 0 = normal condition 1 = alarm condition
54	A	A	A	A	Type: DCME bearer backward alarm Designation: The data word bit No. represents the Rx bearer No. (see Notes 1 and 2) Content: A: 0 = normal condition 1 = alarm condition
55	p	q	x	x	Type: DLC support message Designation: p = Voice/voice-band data q = Unrestricted 64 kbit/s x = Do not care Content: 0 = LL or UCA 1 = HL or UCNA
56	b_1 (MSB)	b_2	R	x	Type: Identification of Rx bearer to which channel check results apply if channel check is progressing normally Designation and content: b_1 b_2: Represents the Rx bearer No. in binary code (see Note 1) R: 1 = channel check disregarded (high BER) 0 = channel check progressing normally x: Do not care

TABLE 5/G.763 *(cont.)*

DCME frame	Data word bit No.				Message (s = spare bit set to 0)
	1	2	3	4	
57	x	BC	BC	BC (LSB)	Type: BC-related channel check results transmitted one BC per DCME multiframe
58	BC (MSB)	BC	BC	BC	
59	D	D	D	D (LSB)	Designation and content:
60	D	D	D	D	BC: 7-bit code represents theNo. of the BC for which the result applies D: 8-bit code represents the No. of the decoder for which the result applies
61	Y	x	x	x	Y: Channel check alarm 0 normal, 1 alarm x: Do not care
62	T	x	x	x	Type: Transmit channel check inhibit Designation and content: T: 1 channel check interrupted 0 channel check normal x: Do not care
63	x	x	x	x	Spare (Note 3)

Note 1 – There is a fixed association between the Rx bearer number in DCME frame 54, the Rx bearer number in DCME frame 56, the VOW IT number, and the local channel IT number. Refer to Table 12/G.763.

Note 2 – In two pool multi-clique operation there is one Rx bearer number associated with each pool.

Note 3 – The unused codes are reserved for implementation of facsimile compression and special tandeming facilities.

10.1.3 *Reporting test results (local DCME)*

The local DCME shall build a result table for each corresponding DCME by accumulating the incoming channel check result messages. The local DCME shall identify the required result messages by examining the bearer identification number contained in the first message (DCME frame 56) of each table. The traffic plan will contain the bearer identification number(s) pertaining to each DCME.

The local DCME shall generate a local alarm when an incoming bearer channel currently subject to the channel check procedure is reporting an abnormal channel check result condition.

10.1.4 *Test vector sequence characteristics*

The test vector sequence shall consist of the following three contiguous segments:

a) 100 ms of 2400 Hz sinusoidal tone at −10 dBm0;

b) the A-law PCM initializing sequence in accordance with § 10.1.5;

c) 768 ms of 1254 Hz sinusoidal tone in accordance with the test vector sequence described in § 10.1.5.

The test pattern receiver shall continuously search for a 1254 Hz sinusoidal tone pattern at an equivalent level of 0 dBm0 ± 1 dB. The test pattern receiver shall be designed to synchronize to the 1254 Hz sinusoidal tone pattern within 100 ms at a bearer error rate of 1 in 10^{-3} while the bearer is operating in 3 bit mode (note). Following synchronization, the test pattern receiver shall declare test pass if the sum of the measured errors does not exceed 2000 for each of the LSB and LSB + 1 bits, and the sum of the errors does not exceed 1000 for each of the MSB through MSB-5 bits in a 600 ms measurement period measured at the PCM output stream. The determination test pass or test fail shall be made at the end of a 600 ms measurement window. The start of the measurement window shall be located 650 ms from the time of occurrence of the assignment message containing the synchronous data code word (1111). The measurement window test time frame and 1254 Hz test tone sequence are shown in Figure 24/G.763.

Note – Operation in 2 bit mode requires further study.

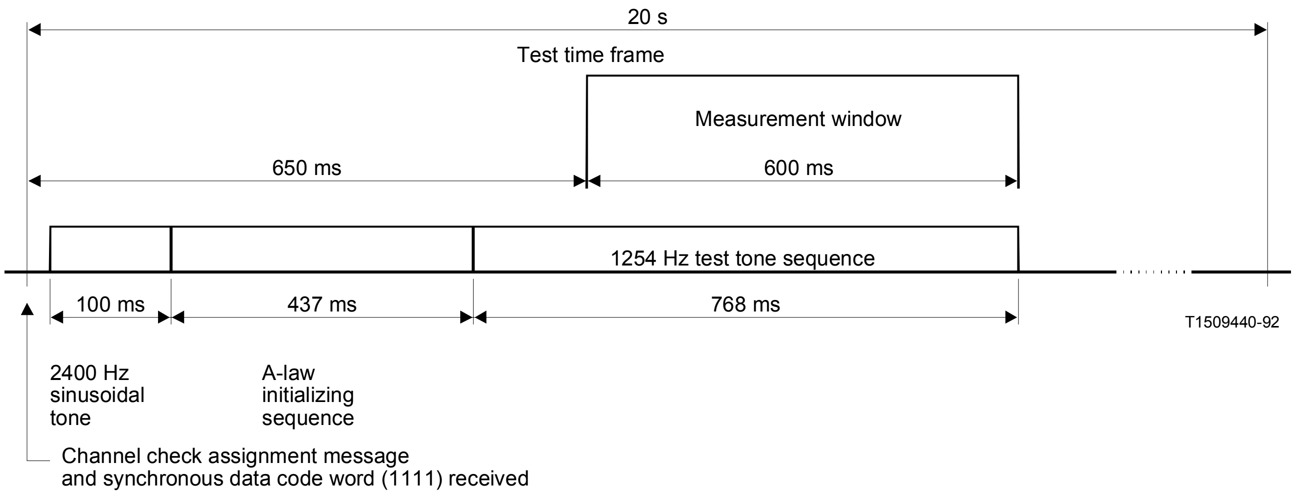


FIGURE 24/G.763

Measurement window and test tone sequence timing relationships

10.1.5 *Channel check test vectors*

The complete test vector sequence comprises a 2400 Hz sinusoidal signal followed by an initializing segment followed by a 1254 Hz sinusoidal signal. All segments are contiguous. The first sequence comprises 834 samples (approximately 100 ms) of a 2 400 Hz sinusoidal sequence encoded in accordance with Recommendation G.711 using A-law encoding. An output sequence is not provided for this input sequence. A reset is assumed prior to the start of the second sequence. The second sequence consists of 3496 samples (approximately 437 ms) of A-law initializing sequence. No output sequence is provided for this input sequence.

The third input test sequence represents a 1254 Hz sinusoidal tone encoded in PCM in accordance with Recommendation G.711 using A-law encoding. The output sequence is the corresponding PCM A-law output obtained when the input test sequence is passed through an ADPCM encoder and ADPCM decoder operated back-to-back.

The output sequence assumes that the decoder ADPCM algorithm has been initialized immediately prior to the reception of the test sequence.

The test sequence format is based on 768 ms of coded signal divided into a series of blocks.

To maintain the accuracy with which the sample sequences will be incorporated in manufacturers' equipment, flexible disks containing these sample sequences may be obtained from the ITU.

10.2 *Internal tests*

It is recommended that an internal test sequence performing a TC-BC-TC loopback test be provided. These tests should, as a minimum, evaluate the activate level of the activity detectors (DCME transmit unit) and the PCM-to-PCM bit integrity (for DCME transmit unit and receive unit). The test sequence should be designed to sequentially evaluate all combinations of channels (TC, IT and BC) and ADPCM codecs.

11 Control channel (CC)

The CC shall be a 32 kbit/s channel, and shall include provisions for accommodating the following categories of inter DCME terminal messages:

- trunk-to-bearer assignment;
- idle noise level;
- dynamic load control;
- alarm information;
- self diagnostic information;
- signal classification.

Each pool of channels within the bearer frame shall contain a CC. The CC shall occupy the lowest numbered 4 bit BC in the pool. The first bit is a sync bit and the remaining 3 bits carry a part of the CC message.

Control channel messages are transmitted at a rate of 3 bits in each 125 µs bearer frame. A complete 48 bit encoded (CC) message shall be transmitted in one DCME frame of 2 ms. Prior to encoding, the CC message shall consist of an 8 bit BC identification word, an 8 bit IT identification word and 8 bits for other DCME to DCME messages (Data Word). The CC message shall be protected by a (24, 12) rate 1/2 Golay code. Figure 25/G.763 illustrates the CC transmission scheme. In the figures describing the CC, the left-hand bits are transmitted first.

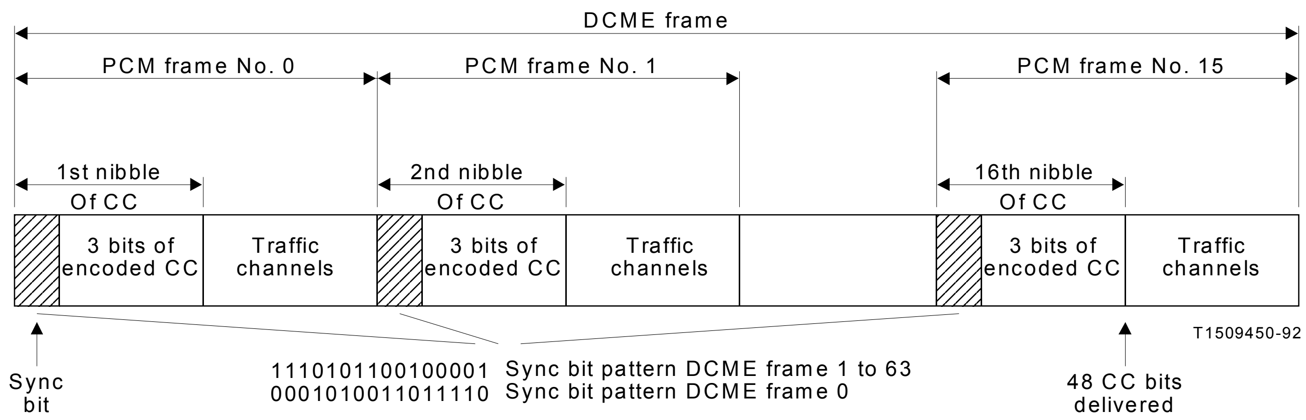


FIGURE 25/G.763

CC message transmission scheme

11.1 *CC error protection*

A (24, 12) rate 1/2 code shall be applied to the CC for error protection. The (24, 12) code is obtained from a (23, 12) Golay code with the addition of a dummy bit and is capable of correcting 1, 2 or 3 bits in error in a block of 24 bits. The code generator polynomial is:

$$g(x) = x^{11} + x^9 + x^7 + x^6 + x^5 + x + 1$$

The 24 information bits comprising 8 bits for the BC number, 8 bits for the IT number and 8 bits for other data are transmitted in two blocks of 12 information bits each. For each information block there is a check block consisting of 11 bits for the Golay code and one dummy bit as shown in Figure 26/G.763. The check bits are obtained by computing the remainder of the polynomial division shown below:

$$x^{11} \cdot I(x) = g(x) \cdot Q(x) + R(x)$$

where,

$I(x) = b11x^{11} + b10x^{10} + \dots b1x + b0$

$g(x) = x^{11} + x^9 + x^7 + x^6 + x^5 + x + 1$

$R(x) = r10x^{10} + r9x^9 + \dots r1x + r0$

$Q(x)$ = quotient of the division

$R(x)$ = remainder of the division

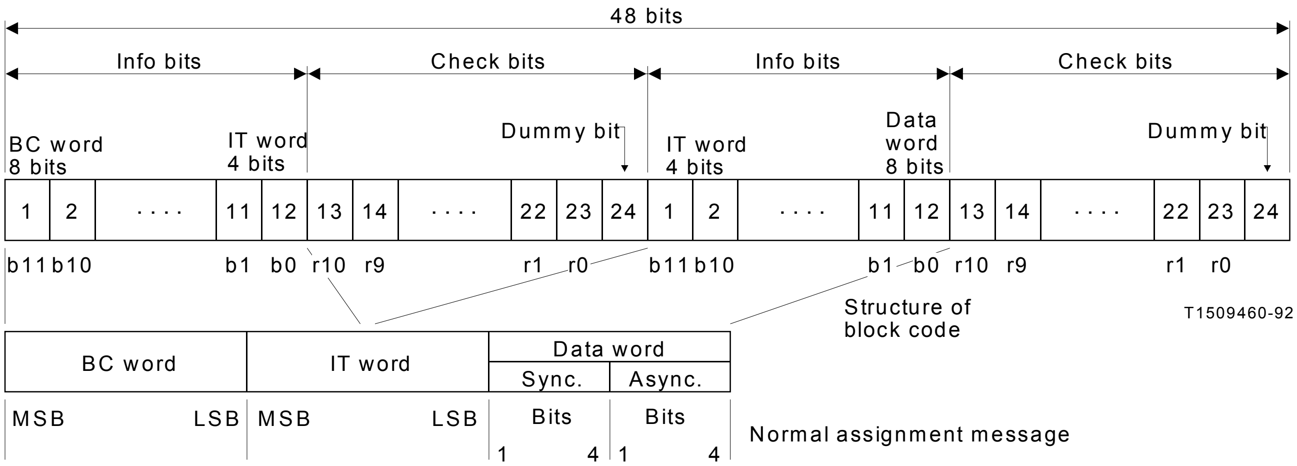


FIGURE 26/G.763

DCME control channel message format

11.2 *CC synchronization*

A 16 bit unique word shall be provided for each individual clique, to identify the beginning of the 2 ms DCME frame over which the encoded CC message of the pool is transmitted. Refer to Figure 25/G.763. The unique word shall be transmitted at the rate of one bit per bearer frame via the sync bit. The sync bit shall occupy the most significant bit position of the CC 4 bit time slot.

The 16 bit unique word shall also provide a means of identifying the beginning of a 128 ms DCME multiframe (64 DCME frames) for use by the asynchronous data word. Refer to § 11.3.3.2.

11.2.1 *Unique word pattern*

The sync bit pattern transmitted in a DCME frame shall constitute the following unique words:

DCME frame 0 0 0 1 0 1 0 0 1 1 0 1 1 1 1 0

DCME frame 1 to 63 1 1 1 0 1 0 1 1 0 0 1 0 0 0 0 1

The order of transmission of the pattern shall be from the extreme left-hand bit first to the extreme right-hand bit last. The first bit of the pattern shall be transmitted in the first nibble of the 16 nibbles constituting a complete CC message.

11.2.2 *Unique word detection*

The unique word detection shall be based on the detection of a correlation match between the accumulated contents of the first bit of the CC nibble and a locally stored unique word pattern. The resulting correlation matches shall be used to attain, maintain and regain the synchronization of the CC message.

In the steady state, a detection threshold of three shall be used to maintain synchronization, and a 3 bit window centred 16 bits after the previous detection of the correlation match shall be used to locate the start of the DCME frame for the proper decoding of the CC message. If the correlation match is not achieved, the CC message bits shall be discarded and a search procedure shall be initiated over a 16 bit window.

11.3 *CC message structure*

11.3.1 *BC identification word*

The MSB of the 8 bit BC identification word shall be used to indicate the BC type. For data, the MSB shall be 1. For all other BC types (bit bank, transparent, voice), the MSB shall be 0.

The seven LSBs in binary code shall identify the BC number in accordance with § 5.9. The normal BC numbering range shall be 1 through 61. The overload BC numbering range shall either be 64 through 83 or if the optional 2 bit encoding mode is available and enabled 64 to 124.

For 64 kbit/s transparent channels, the BC number shall identify the first 4 bit BC of a pair of adjacent 4 bit BCs used to create an 8 bit BC and shall be even numbered in the range 2 through 60. A channel type identifier code in the synchronous DCME to DCME Data Word shall be used, as defined in § 11.3.3.1 to indicate a 64 kbit/s transparent channel.

BC number 0 in binary code shall be used for CC messages transmitted during system start-up or during a DCME transmit unit map change.

BC number 255 in binary code shall be used to indicate an ineffective CC message if all traffic is preassigned.

11.3.2 *IT identification word*

The 8 bits of the IT identification word shall be used to identify the ITs. ITs numbered 1 to 216 in binary code shall be available for traffic. When less than 216 ITs are used, the numbering will not necessarily be numerically consecutive.

IT numbers 232, 233, 234 and 235 in binary code shall be used for DCME to DCME orderwires (up to four correspondents), see § 15.9.

IT numbers 239/240 in binary code shall be used for the automatic end-to-end channel check procedure, see § 10.

IT number 0 in binary code shall be used to indicate an explicit disconnection or shall be transmitted in the CC during system start-up and DCME transmit unit map changes.

IT number 250 in binary code shall be used when the associated BC is to be utilized for the bit bank as described in §§ 6 and 7.

IT number 255 in binary code shall be used to indicate an ineffective CC message if all traffic is preassigned.

11.3.3 *Data word*

The 8 bit data word in the CC message forms two independent data channels. The first data channel consists of the four MSBs of the 8 bit data word, and is transmitted with each assignment message synchronously relative to the BC and IT identification.

The second data channel consists of the remaining 4 bits of the 8 bit data word transmitted in a multi-frame structure asynchronously relative to the BC and IT identifications.

11.3.3.1 *Synchronous data word*

The 4 bit synchronous data word is used:

a) to transmit background noise level information to the DCME receive unit;

b) to indicate that the BC is the first 4 bit nibble of a 64 kbit/s transparent channel;

c) to indicate that the BC is assigned for the channel check procedure;

d) to indicate an ineffective message;

e) to carry user signalling bits when the optional USM is used.

Background Gaussian noise, as determined at the transmit activity detector, will vary between −68 dBm0 and −42 dBm0 (see Note). For channels subject to DSI, the background noise level shall be encoded in accordance with Table 6/G.763. The noise level code shall be transmitted with each new assignment and refreshment message.

Note – For A-law encoding the minimum noise level is −65 dBm0.

For each CC message, the DCME receive unit shall decode the 4 bit data word and update the noise level memory associated with the decoded IT according to Table 6/G.763. At the DCME receive unit, a pseudo-random 8 bit PCM sequence simulating Gaussian noise shall be applied to the disconnected IT. The simulated noise level shall match the last stored value in the noise level memory before the disconnection.

For channels carrying 64 kbit/s transparent calls, the 4 bit data word shall be 1001 and transmitted with each new assignment, refreshment and disconnection message.

If the BC in the assignment message is being subjected to the automatic channel check procedure according to § 10, the 4 bit data word shall be 1111.

11.3.3.2 *Asynchronous data word*

The 4 bit asynchronous data word will convey the following types of DCME-to-DCME information:

a) end-to-end circuit supervision and alarm indications on a per channel basis;

b) bearer related backward alarm indication to the remote DCME;

c) DLC support messages;

d) BC related messages pertaining to channel check procedures.

The data word multiframe shall consist of 64 DCME frames numbered from 0 to 63. Frame No. 0 is the DCME frame in which the CC unique word is inverted. The CC unique word for the remaining 63 frames shall be transmitted normally.

Bit allocations in the data word multiframe for the various applications shall be as shown in Table 5/G.763.

11.3.4 *CC structure when USM option is used*

If the optional USM is used, the BC identification word and the CC synchronous data word can be formatted according to the users' requirements in the DCME frames 0, n, 2n, etc. (i.e. every nth DCME frame) of the DCME multiframe.

For the R2 USM every eighth frame of the DCME multiframe shall be used to transmit a signalling message as follows. The bits 1 to 8 of the signalling message shall identify ITn1. The bits 9 to 16 of the signalling message shall identify ITn2. Bits 17 and 18 shall encode the a and b bits of ITn1. Bits 19 and 20 shall encode the a and b bits of ITn2. The a and b bit signalling information will be either change of signalling states, or the refreshment of existing states. Figure 27/G.763 illustrates the format for this type of message.

TABLE 6/G.763

4 bit synchronous data word encoding

Transmit DCME action Measure noise level n (dBm0) (Notes 1, 2)	Code Word	Receive DCME Reaction Store noise level m (dBm0)
n < –68	0 0 0 1	–68 (μ-law only)
–68 ≤ n < –62	0 0 1 0	–65
–62 ≤ n < –57	0 0 1 1	–60 (default)
–57 ≤ n < –52	0 1 0 0	–55
–52 ≤ n < –47	0 1 0 1	–50
–47 ≤ n < –44	0 1 1 0	–46 (Note 3)
–44 ≤ n < –42	0 1 1 1	–43 (Note 3)
–42 ≤ n	1 0 0 0	–42 (Note 3)
BC identifies 64 kbit/s channel	1 0 0 1	BC indicates first 4 bit nibble of 8 bit channel
BC is under channel check test	1 1 1 1	BC is under channel check test
Ineffective message	0 0 0 0	Ineffective
Unused codes (Note 4)	1 0 1 0 1 0 1 1 1 1 0 0 1 1 0 1 1 1 1 0	No action required

Note 1 – It is suggested that because the noise inserted at the receive unit is broadband, the transmit unit noise measurement should also be broadband.

Note 2 – The DCME transmit unit noise intervals are implementation specific, a tolerance of ± 2 dB is suggested.

Note 3 – When the background noise level is high (–46 dBm0 or greater) some administrations have indicated there may be a subjective benefit in inserting lower values of noise at the receive unit than those measured at the transmit unit. The contrast is most apparent when the noise spectral density at the DCME transmit unit is substantially different from the noise inserted at the receive unit. Since the noise inserted at the receive unit does not affect DCME interoperability the selection of the noise level is left as an option (–50 dBm0 is being considered).

Note 4 – The unused codes are reserved for implementation of facsimile compression and special tandeming facilities (for further study).

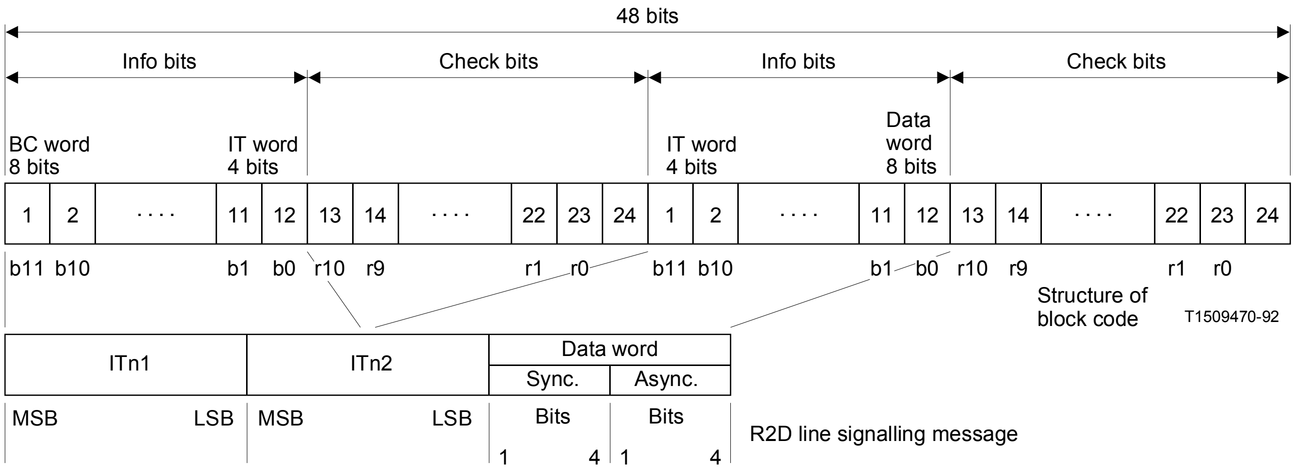


FIGURE 27/G.763

R2D line signalling message format for CC

12 Activity detection and data/speech discrimination

This section describes the functional requirements of the transmit activity detector, data/speech discriminator, signalling detector and receive activity detector.

Compliance with all paragraphs of this section is mandatory with the exception of the transmit activity detector threshold and operate time specification. Compliance with the threshold and operate time specification is not required to achieve interworking between various DCME manufacturers. The performance of the DCME transmit unit activity detector will be assessed by conducting MOS subjective tests on the entire DCME system. DCME testing methodologies have been specified by CCITT Study Group XII in Recommendation P.84.

12.1 *Transmit activity detector*

For each IT, the transmit activity detector characteristics are based upon the assumption that the amplitude frequency response of the transmission channel (up to the input of the activity detector) is ±0.5 dB with respect to 1000 Hz over the frequency band from 300 to 3400 Hz. The Gaussian noise level can typically vary over a range from −68 to −42 dBm0.

Note – For A-law encoding, the minimum noise level is −65 dBm0.

Functionally, the transmit activity detectors shall determine whether or not there is activity on each transmit IT and provide an active/inactive (act/Inact) indication. Upon system start-up or map change, the transmit activity detectors shall be reset to provide an Inact indication.

Functionally, the transmit activity detectors shall determine the transmit idle channel noise level on each non-pre-assigned IT in the DCME transmit unit. The idle channel noise level for each DCME transmit unit IT is encoded and transmitted to the DCME receive unit in the 4 bit synchronous data word. The idle channel noise is regenerated in the DCME receive unit in accordance with § 11.3.3.1 and is applied to the corresponding DCME receive unit ITs when they are disconnected from their assigned BCs.

12.1.1 *Threshold and operate time*

The transmit activity detector threshold shall automatically adjust relative to the average power of Gaussian noise band limited between 300 to 3400 Hz.

The threshold and operate time of the transmit activity detector may be implementation specific. However, for guidance threshold and operate time, characteristics for the transmit activity detector are given in Annex B.2.

12.1.2 *Hangover control*

The permissible hangover time as a function of stimulus signal duration shall be within the mask shown in Figure 28/G.763 for CCITT Signalling System No. 5 and within the mask shown in Figure 29/G.763 for speech and CCITT Signalling Systems Nos. 6, 7 and R2D.

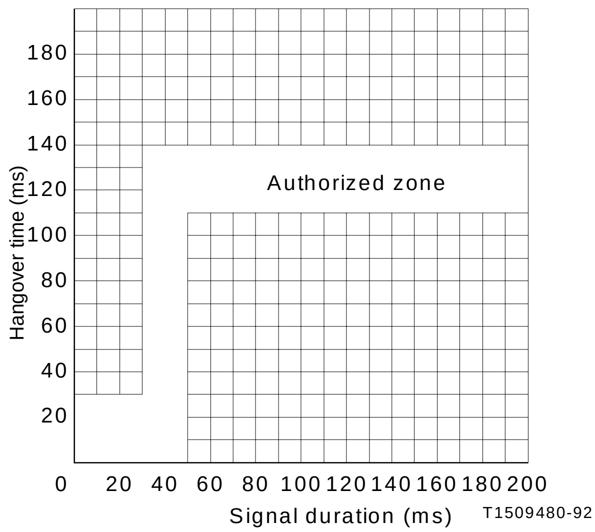


FIGURE 28/G.763

Hangover time mask – CCITT No. 5 Signalling

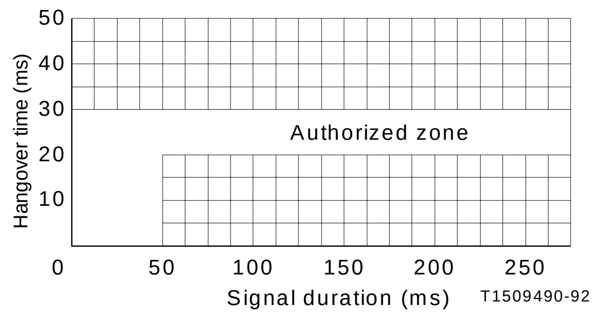


FIGURE 29/G.763

Hangover time mask – Speech

It shall be possible to select the required type of hangover time mask. For voice-band data the hangover time should be extended so that it is sufficiently long to bridge facsimile page changes. This time may be as long as 14 s.

12.1.3 *Interaction of transmit activity detector with echo control devices*

The threshold of the transmit activity detector shall not adapt to Gaussian noise level variations which are due to the actions of echo suppressors or echo cancellers. This might be accomplished, for example, by providing the transmit activity detector with a threshold inhibit signal from a receive activity detector when activity is present in the receive channel (see Annex B.5. Special DCME networking considerations).

12.2 *Data/speech discriminator*

The data/speech (D/S) Discriminator shall determine whether the activity on each IT in the DCME transmit unit is speech or data and provide a speech/data indication to the TCP function. An example of a data/speech Discriminator which satisfies the requirements specified in this section is given in Annex B.3.

The following requirements shall be met with the modem types and bit rates given in Table 7/G.763.

12.2.1 *Output conditions*

The D/S Discriminator shall analyse the activity on each transmit IT and provide the following output conditions.

The D/S Discriminator shall provide a continuous output condition indicating the presence of either speech or data on the IT. The current output condition shall be maintained upon termination of activity on the IT or until the output condition of a subsequent activity is determined. Upon system start-up or map change, the D/S discriminator shall be reset to voice.

12.2.2 *Accuracy*

The missed detection probability of data as speech or speech as data shall be less than 0.5 per cent.

12.2.3 *Response time*

The D/S Discriminator shall update its output condition within 200 ms after any of the following changes in the IT signal characteristics:

– inactive-to-speech;

– inactive-to-data;

– speech-to-data;

– data-to-speech.

12.2.4 *2100 Hz tone detector*

The D/S Discriminator shall detect the presence of the V.25 echo control disabling tone by analysing signals on the transmit ITs. The function may be implemented separately but is here defined as part of the D/S discriminator. Requirements for a 2100 Hz tone detector are given in Annex B.3.

TABLE 7/G.763

Types of modem and bit rates which shall be supported

Modem	Bit rate (Bit/s)	Operating mode
V.21	300 bit/s	FDX
V.22	600, 1200	FDX
V.22*bis*	2400	FDX
V.23	1200	HDX, character mode HDX, continuous
Group 1, 2	Analogue	FAX
V.26	2400	FDX
V.26*bis*	1200, 2400	HDX
V.26*ter*	1200, 2400	FDX
V.27*bis*	2400, 4800	HDX
V.27*ter*	2400, 4800	HDX Group 3, FAX
V.29	4800, 7200, 9600	HDX/FDX Group 3, FAX
V.32	2400, 4800, 9600	FDX
V.33	9600	FDX

Activity	*Output condition*
Speech	Voice
Tones and tone pairs (Note 1)	Voice
Data signal (Note 2)	Data
2100 Hz	Data

Note 1 – Where a signal frequency tone i.e. an unmodulated carrier, is part of a voice-band data modem signal exchange, once the signal is classified as data the classification should not return to voice within the data call. This may be accomplished by either:

a) specific recognition of those tones which are used in the modems specified in Table 7/G.763; or

b) delaying the transition from data to speech for a specified minimum time (0.5 to 1.0 sec.).

Note 2 – V.21 modem signals must be classified as data to ensure that facsimile signals will not be corrupted.

12.3 *Signalling detector*

Functionally, the signalling detector shall detect the presence of CCITT Signalling System No. 5 line signalling (2400 Hz) on each transmit IT, provide a detection indication (signal detect/No detect) to the TCP function and enable the signalling hangover time mask (Figure 28/G.763) for the duration of the signalling interval. Upon system start-up or map change, the Signalling Detector indication shall be reset to no Detect. Requirements for a 2400 Hz tone detector are given in Annex B.4.

R2D inter-register signalling does not need an extended hangover and shall be classified as voice.

12.3.1 *Accuracy*

The probability of speech, voice-band data or noise being detected as CCITT Signalling System No. 5 signalling or the probability of signalling being detected as speech, voice-band data or noise shall be less than 0.5 per cent.

12.4 *Receive activity detector*

A receive activity detector may be used to recognize periods of activity on each received IT and provide an inhibit signal to prevent interaction of the transmit activity detector with echo control devices. Refer to § 12.1.3.

13 DCME synchronization and echo control

13.1 *DCME synchronization*

Timing synchronization of DCME can be achieved in many ways and care should therefore be taken in any implementation to ensure that the configuration adopted is correct.

13.1.1 *Reference clock*

The DCME reference clock shall be derived from a source which meets the requirement of Recommendation G.811. For networks that entail one international destination, loop timing can be used as an alternative at one end of the link. The need for an internal reference clock for use under failure conditions is for further study.

13.1.2 *Plesiochronous slips*

The slip rate shall not exceed the requirement of Recommendation G.822. Controlled slips at 2048 kbit/s, on the trunk side shall be 2 frames, controlled slips at 1544 kbit/s on the trunk side and for 2048 kbit/s and 1544 kbit/s on the bearer side require further study.

13.1.3 *Buffer sizes and locations*

Table 8/G.763 indicates suitable buffer sizes and locations for the 2048 kbit/s hierarchy for the various synchronization options which are detailed in Annex B.6. A table for the 1544 kbit/s hierarchy is under study.

TABLE 8/G.763

Buffer sizes and locations for the 2048 kbit/s hierarchy

Synchronization type (Note 3)	Buffer size (Note 1)	Slip size (Note 2)	Location (Note 4)	Figure No.
1. No buffering				
A Asynch	No buffer	–	–	B-4/G.763
B Synch	No buffer	–	–	B-5/G.763 B-18/G.763 B-15/G.763
C Synch analogue to digital	No buffer	–	–	B-8/G.763
2. Plesiochronous/ buffering				
A Asynch	0.5 ms	2 frames	Trunk side	B-6/G.763
B Synch	0.5 ms	2 frames	Bearer side	B-7/G.763 B-16/G.763 B-19/G.763
3. Plesiochronous/ Doppler buffering				
A Synch	1.7 ms	2 frames	Bearer side	B-9/G.763 B-14/G.763 B-17/G.763 B-20/G.763 B-22/G.763
B Synch	2.4 ms 1.7 ms		Bearer side and trunk side	B-10/G.763
C Asynch	1.7 ms	2 frames	Trunk side	B-12/G.763
D Synch	2.4 ms 1.7 ms		Trunk side	B-11/G.763
E Synch	1.7 ms	2 frames	Trunk side	B-13/G.763 B-21/G.763

Note 1 – Buffer sizes are derived from the following:

– single Doppler with plesiochronous buffer: (0.6 ms × 2) + 0.5 = 1.7 ms;

– double link Doppler buffer: 1.2 ms × 2 = 2.4 ms;

– plesiochronous buffer for 2 PCM (2048 kbit/s) frames: (2 × 0.125 ms) × 2 = 0.5 ms.

The Doppler buffer size used is an example for a specific satellite. These buffer sizes may need to be adjusted taking into account the orbital parameters of the satellite in use.

Note 2 – The slip size of 2 PCM frames is based upon the requirement in the 2048 kbit/s frame to maintain frame alignment.

Note 3 – Asynch refers to the case where the transit unit and receive unit of the same DCME terminal derive their timing from different clock sources.

Note 4 – In general it is preferable to avoid placing the plesiochronous slip buffers on the bearer side of the DCME to minimize disruptions caused by slips. This may not be possible under all circumstances.

13.1.4 *Terminal synchronization*

The DCME terminal shall be capable of deriving its timing from any of the incoming digital links or from an external clock. When the synchronization is derived from the trunk receive side it is recommended that a fallback trunk receive synchronization source be provided. This is for the event of the primary synchronization link entering an alarm condition indicating a received line signal failure, loss of frame alignment, AIS or receive BER $\geq 10^{-3}$. Switching between primary and fallback sources shall be automatic.

Note – Synchronization arrangements for special operation of DCME in tandem are under study.

13.2 *Echo control*

Echo control is not considered to be part of the DCME Recommendation. A network echo control device integrated or external to the DCME and meeting or exceeding the requirements of Recommendations G.165, G.164, or Recommendation G.161 shall be present on all TCs carrying speech serviced by a DCME.

A lack of echo control on the circuits serviced by the DCME will degrade speech performance due to the increased speech activity factor resulting from the echo signal.

Transmit activity detector/echo control device interactions are controlled by freezing the activity detector threshold in the presence of speech on the corresponding receive channel.

14 ADPCM encoders and decoders

ADPCM encoders and decoders shall be capable of operation within the DCME at the following bearer channel transmission rates:

- 64 kbit/s: 8 bit/sample transparent;
- 40 kbit/s: 5 bit/sample ADPCM;
- 32 kbit/s: 4 bit/sample ADPCM;
- 24 kbit/s: 3 bit/sample ADPCM;
- 16 kbit/s: 2 bit/sample ADPCM (optional).

For 64 kbit/s bearer channels (8-bit mode), the ADPCM encoders and decoders shall be bypassed.

For 40 kbit/s bearer channels (5-bit mode), 32 kbit/s bearer channels (4-bit mode), 24 kbit/s bearer channels (3-bit mode) and 16 kbit/s bearer channels (2-bit mode), the ADPCM encoders and decoders shall be in accordance with Recommendation G.726 and shall operate in accordance with §§ 6.1.6 and 7.1.4.

Digital sequences (test vectors) for use in the verification of correct implementation of the ADPCM algorithms are available on flexible disks. Copies of the flexible disks may be obtained from the ITU.

15 Operations and maintenance functions

The following operations and maintenance functions shall be performed at the DCME. Additional operation and maintenance functions are under study:

a) configuration of the DCME for operation in a network;

b) traffic rearrangements under coordinated operator control;

c) voice orderwire (VOW) communication to correspondent DCMEs;

d) attendance to prompt maintenance alarms resulting from the channel check procedure, the continuous BER measurement, and other fault conditions;

e) storage and display of status information pertaining to the freeze-out fraction, DLC operation, channel check procedure, and control channel BER and fault analysis;

f) redundancy switchover facility;

g) display of statistical information and anomaly reports.

The DCME should provide the following maintenance functions:

a) Facilities for disabling (terminal out of service tests):

   – DSI: Digital Speech Interpolation;

   – LRE: Low Rate Encoding (ADPCM);

   – VBR: Variable Bit Rate Coding.

b) Facilities for providing fixed connections of:

   – specific trunk channels to specific bearer channels, at 64 kbit/s without interpolation, 40 kbit/s without interpolation, 32 kbit/s without interpolation and optionally at 24 kbit/s or 16 kbit/s without interpolation (see § 4.2.1).

c) Facilities for protected monitoring points (under study).

15.1 *Configuration of the DCME for operation in a network*

Operation of DCME in a network will require bilateral or multilateral agreement between correspondents on the use of trunk and bearer channels. Table 9/G.763 outlines the operational parameters on which bilateral or multilateral agreements are required.

Operation of the DCME will also require configuration data which is of concern only to the local user. Table 10/G.763 outlines the unilateral operational parameters.

The DCME shall include a capability to permit the entry of configuration data into a background mapping facility without interruption to service which is utilizing configuration data in a foreground map. The configuration data shall permit operator control of:

a) Dynamically assigned transmit and receive trunk time slots by permitting semi-permanent TC to IT associations. TCs may be identified by digital group and time slot; ITs shall be identified by number (1 through 216);

b) Pre-assigned transmit and receive trunk time slots by permitting semi-permanent TC-to-IT-to-BC associations. Pre-assignments of 24 kbit/s bearer channels and optionally 16 kbit/s for maintenance and 64 kbit/s, 40 kbit/s and 32 kbit/s bearer channels for maintenance or traffic shall be possible. The number of pre-assigned channels for traffic need not be symmetrical between transmit and receive sides.

c) The transmit and receive order wires by permitting semi-permanent IT to correspondent associations.

d) The boundaries of the single (multi)-destination pool(s) for transmit and receive bearer frames (upper bound pool 1, lower bound pool 2) shall be selectable in increments of one 8-bit time slot. The system does not require that the pool(s) occupy the entire bearer frame. The bits in unused time slots should not be permitted to indicate an alarm condition in normal operation (note).

   Note – Configuration and operation of DCME for special tandeming arrangement is for further study.

e) Channel check permanent associations (see Table 12/G.763).

TABLE 9/G.763

DCME operational parameters subject to multi– (bi) – lateral agreements

Mode of opération Number of destination	Point-to-point, 1	Multi-destination 2-4	Multi-clique 2
Destination identification		Name/number	
Optional USM activated		Yes/no	
Optional USM repetition interval		R	
a) Dynamically assigned correspondents			
Tx pool boundaries		Pool boundary shall coincide with an 8-bit TS boundary	
Rx pool boundary per Rx bearer		Pool boundary shall coincide with an 8-bit TS boundary	
Tx TC/Local IT mapping		TC (primary group No., TS No.)/Local IT (No.)	
Rx remote IT/local IT mapping per Rx bearer		Remote IT (No.)/local IT (No.) Remote IT (No.)/to other destination	
b) Pre-assigned correspondents			
Tx pre-assigned mapping 64 kbit/s, 40 kbit/s, 32 kbit/s		TC (group, No., TS No.)/ local IT (No.)/BC (No.)	
Rx pre-assigned mapping 64 kbit/s, 40 kbit/s 32 kbit/s		BC No./remote IT (No.)/ local IT (No.)	
c) Clock source			
Provided on trunk group, bearer clock or external			

TABLE 10/G.763

DCME operational parameters unilaterally determined

Parameter	Note
No. of primary trunk groups	No. of 1544 kbit/s or 2048 kbit/s
DLC timers	Adjustment of Ta, Tb
DLC thresholds	Low load, high load
DLC averaging (Note)	See Annex B.1
DCME TC-trunk identification mapping	For DLC and seizure/release in TCH
TCH/DLC interaction	Enabled/disabled
Bearer backward alarm mapping	For the local DCME alarm
Circuit supervision TC-trunk identification mapping	For SCI
Channel check procedure	ON/OFF
Statistic time interval (STI)	See § 15.2.3

Note – Non-mandatory (implementation specific)

15.2 *System management functions*

15.2.1 *Transmission facilities*

Each terminal should monitor each incoming digital link for the following conditions or parameters and store separate cumulative counts of each type of event as required by users:

– AIS, remote alarm indication;

– loss of incoming signal, loss of frame alignment, reframe rate;

– errored seconds, severely errored seconds;

– slips, slip rate.

15.2.2 *Terminal traffic handling performance*

The DCME terminals shall monitor and store records of the various parameters needed to evaluate the traffic handling performance being provided. These shall include the statistics given in Table 11/G.763.

TABLE 11/G.763

DCME management statistics

Service to be measured	Quality of service statistics	Offered traffic statistics
Voice	1) Bits per sample 2) Voice queue freezeout fraction 3) Voice freezeout excess.	4) Voice activity ratio 5) DLC voice on ratio
Data	6) Data queue freezeout fraction	7) Data activity ratio
64 kbit/s on demand	8) 64 failed seizures ratio	9) 64 connected ratio 10) DLC 64 on ratio
All services	11) Average BER 12) BER excess 13) Severely errored seconds	

Note – Statistics 1) to 4) and 6) to 9) shall be calculated separately for each transmit pool.

Statistics 5) and 10) shall be calculated separately for each destination.

Statistics 11) and 12) shall be calculated for each receive control channel.

Statistic 13) shall be calculated separately for each incoming digital link (trunk and bearer).

15.2.3 *System statistics measurement*

Measurements and calculations of traffic statistics shall be done only on non-pre-assigned trunk channels which are defined in the configuration data. The DLC ON ratio for voice/voice-band data and the DLC ON ratio for 64 kbit/s unrestricted traffic parameters shall be obtained separately for each destination. All other parameters shall be obtained separately for each transmit pool. The measurements of each parameter shall be made over an operator settable statistics time interval (STI). Each statistic shall be calculated once every 1.0 minute interval with the accumulated data from every sampled DCME frame (e.g each 10th frame). The average over the STI shall be the average of the values calculated each 1.0 minute interval during the STI within the range from 10 minutes to 60 minutes (in 10 minute steps).

The BC states that need to be considered for the calculation of the system statistics are specified as follows:

– *Voice:* The connected TC carries speech signals or in-band signalling or calling tones (and marginally active voice-band data when not yet recognized as such), extended with their corresponding hangover time (see Note 1).

– *Data:* The connected TC carries active voice-band data signals (including 2100 Hz tone) recognized as such, extended with its corresponding hangover time (and marginally "voice" signals when not yet recognized as such. (See Note 2).

– *Transparent:* The connected TC carries a 64 kbit/s unrestricted traffic call.

– *Disconnected:* No TC is connected to this BC.

– *Pre-assigned:* The BC is permanently assigned to a TC.

Note 1 – Once a TC has been declared voice or data and the corresponding hangover time of the connection has expired during inactivity, the TC is reputed to be declared initially as voice in both cases when activity resumes. Furthermore when the hangover time of a voice call has not expired, new activity in the BC is declared initially as voice.

During low activity periods, after the above-mentioned hangover time expires, inactive voice TCs will still be connected and coded like active ones at the rate of 4 bits/sample as long as overload BCs are not needed. (This is done to avoid front clipping when activity resumes on those TCs.)

As a consequence, the average number of bits/sample for voice is significant only when the result is less than 4 bits/sample.

Note 2 – When the hangover time of a data call has not expired, new activity in the BC is declared initially as data.

The system statistics monitor will deliver the results of calculations relative to the following definitions. In the definitions, N is the number of sampled DCME frames of the averaging period of 1.0 minute.

15.2.3.1 **bits/sample for voice**

Defined as the average number of encoding bits per sample for all connected TCs used for voice. The average should be calculated to two decimal places.

$$\text{bits/sample for voice} = \frac{\sum_{N} \text{No. of bits within the bearer used for voice BC}}{\sum_{N} \text{No. of non-pre-assigned TCs classified differently from transparent, data or inactive}}$$

15.2.3.2 **voice queue freezeout fraction (voice FOF)**

Defined as the ratio of competitive clip duration to voice spurt duration. The fraction may be determined as the ratio of the number of non-pre-assigned TCs classified as voice-active but not connected, to the total number of non-pre-assigned TCs classified as voice-active connected plus not connected. The ratio should be expressed as a percentage to three decimal places.

$$\text{Voice FOF} = \frac{\sum_{N} \text{No. of non-pre-assigned TCs classified as voice-active (not connected)}}{\sum_{N} \text{No. of non-pre-assigned TCs classified as voice-active (not connected + connected)}} \times 100$$

15.2.3.3 **voice freezeout excess**

% of time voice FOF exceeds 0.5% when averaged over 1 minute

$$\text{Voice FOF excess} = \frac{\text{No. of 1 minute periods in STI in which voice FOF} > 0.5\%}{\text{No. of 1 minute periods in STI}} \times 100$$

given to 2 decimal places.

15.2.3.4 **voice activity ratio**

Defined as the ratio of the number of non-pre-assigned TCs which are classified as voice-active, to the total number of non-pre-assigned TCs. The ratio is expressed as a percentage to the nearest integer.

$$\text{Voice activity ratio} = \frac{\sum_{N} \text{No. of non-pre-assigned voice-active TCs}}{\text{No. of non-pre-assigned TCs} \times \text{N}} \times 100$$

15.2.3.5 **DLC voice on ratio**

Defined as the ratio of the number of DCME frames during which DLC for voice/voice-band data (V/VBD) is ON, to the total number of DCME frames N. The ratio is expressed as a percentage to the nearest integer.

$$\text{DLC voice on ratio} = \frac{\text{No. of sampled DCME frames with DLC ON for V/VBD}}{\text{N}} \times 100$$

15.2.3.6 **data queue freezeout fraction (data FOF)**

Defined as the ratio of the number of non-pre-assigned TCs classified as data-active but not connected, to the total number of non-pre-assigned TCs classified as data-active (connected + not connected). The ratio should be expressed as a percentage to three decimal places.

$$\text{Data FOF} = \frac{\sum_{N} \text{No. of non-pre-assigned TCs classified as data-active (not connected)}}{\sum_{N} \text{No. of non-pre-assigned TCs classified as data-active (not connected + connected)}} \times 100$$

15.2.3.7 **data activity ratio**

Defined as the ratio of the number of non-pre-assigned TCs which are classified as data-active, to the total number of non-pre-assigned TCs. The ratio is expressed as a percentage to the nearest integer.

$$\text{Data activity ratio} = \frac{\sum_{N} \text{No. of non-pre-assigned data-active TCs}}{\text{No. of non-pre-assigned TCs} \times \text{N}} \times 100$$

15.2.3.8 **64 kbit/s failed seizures ratio**

Percentage of 64 kbit/s on demand seizure (S64) attempts that receive a 64 kbit/s negative acknowledgement (S64 NACK) from the DCME

$$\text{64 kbit/s FSR} = \frac{\text{count of S64 signals sent in STI}}{\text{count of S64 NAck signals received in STI}} \times 100$$

given as an integer.

15.2.3.9 **64 kbit/s connected ratio**

Defined as the ratio of the number of non-pre-assigned TCs which are classified as 64 kbit/s connect-called plus 64 kbit/s connect-calling, to the total number of non-pre-assigned TCs. The ratio is expressed as a percentage to the nearest integer.

$$\text{64 kbits connected ratio} = \frac{\sum_{N} \text{No. of non-pre-assigned 64 kbit/s TCs connect-called and -calling}}{\text{No. of non-pre-assigned TCs} \times \text{N}} \times 100$$

15.2.3.10 **DLC 64 kbit/s on ratio**

Defined as the ratio of the number of DCME frames during which DLC for 64 kbit/s unrestricted is ON, to the total number of DCME frames N. The ratio is expressed as a percentage to the nearest integer.

$$\text{64 kbit/s DLC-on ratio} = \frac{\text{No. of sampled DCME frames with DLC for 64 kbit/s ON}}{\text{N}} \times 100$$

15.2.3.11 **average BER**

Average BER, as measured on the receive control channel

$$\text{Average BER} = \frac{\text{count of No. of bit errors identified in the control channel}}{\text{count of total No. of bits received in the control channel}} \times 100$$

15.2.3.12 **BER excess**

% of time average BER exceeds 1×10^{-3} when averaged over 1.0 minute

$$\text{BER excess} = \frac{\text{No. of 1 minute periods in STI in which BER} > 1 \times 10^{-3}}{\text{No. of 1 minute periods in STI}} \times 100$$

given as an integer.

15.2.3.13 *Severely errored seconds ratio* (see Recommendation G.821)

It is important that the voice and data performance are measured separately for the following reasons:

– the effect of freezeout and clipping is different on voice calls and data calls;

– the DCME process gives priority to assigning activity classed as data and hence the freezeout figures for the data queue should always be less than the corresponding freezeout figures for the voice queue.

The summary statistics calculated at the end of the STI shall be output to a statistics data file on a secure storage medium (e.g. non-volatile RAM, hard disk, etc.).

15.3 *Synchronizer*

The state of synchronization of each primary group interface, the selected clock source, and the times of any failure or changes of clock source should be monitored.

15.4 *Communication links*

The condition of all communication links should be monitored to detect failures as far as practicable, including:

- control channels;
- ISC-DCME interface;
- man-machine interface.

15.5 *Reports*

The terminal should:

a) at operator defined intervals, or when set parameters have been exceeded, or as a worst 15 minutes report for any 24 hour period, file operator selected parameters from those monitored and stored plus header information such as terminal identification, date and measurement period covered by the file;

b) compare selected parameters, status or measurements with predetermined conditions;

c) upon detection that predetermined conditions have been met or exceeded for a given period of time, take the necessary action(s) which may include:

   1) filing of an anomaly report;
   2) transmission of alarm signals;
   3) blocking all new calls due to failure;
   4) switching to standby, if available;
   5) total shut down of the terminal.

15.6 *System configuration*

The terminal shall include a non-volatile back-up memory containing a copy of the latest configuration of the DCME, for use in failure situations. A non-working spare copy should also be provided to allow changes in configuration to be made without impact upon service security. In cases where cluster operation of terminals is used to provide additional service security, means must be provided for the standby terminal to adopt the configuration of the working terminal which it is intended to replace.

The configuration information shall include details of trunk side interface channel connections, modes of operation of any pre-assigned channels, and restrictions in force to any destination or on any block of circuits (e.g. limit on the number of 64 kbit/s calls) and synchronization source.

15.7 *Failure protection strategy*

Upon detection of service affecting conditions the DCME shall take the appropriate actions to protect existing traffic, such as switching to fallback timing sources or fallback units or where redundancy is provided, transmission of DLC signals, disconnection of failed circuits, or transmission of any appropriate alarm conditions.

15.8 *Coordinated traffic rearrangements*

A map change handler (MCH) function shall be provided which the operator can manually disable or enable. When disabled, it shall not be possible to command a map switch. When enabled, it shall be possible to manually command a map switch. The coordination of map changes may be performed between correspondents by voice orderwire.

When the MCH is enabled, the channel check procedure shall be inhibited and the DLC ON condition shall be automatically sent toward the local TCH and the local SCI.

The MCH enabled condition shall be terminated by operator selection of either MCH disabled or a map change command. Upon disabling, the channel check procedure shall restart and the normal DLC conditions shall apply as defined in § 9.

During a traffic rearrangement, the BC, IT and data word contents in the CC shall be set to 0. When such an assignment message is received, no action shall be taken based on the assignment message contents. However, the operator shall be notified.

After the map change command is given, the foreground and background maps shall be switched. The MCH shall initiate the MCH related processes associated with the DCME transmit unit, the DCME receive unit and the 64 kbit/s circuit handler after determining the parameters required for their operation in accordance with the new foreground map (note). The channel check procedure shall restart and the normal DLC conditions defined in § 9 shall apply.

Note – This function shall also initiate the MCH related processes which are required at DCME system start-up.

15.9 *Voice orderwire (VOW)*

It shall be possible to connect a VOW from the local DCME to any corresponding DCME by accessing an ADPCM channel in competition with voice traffic. The voice signal and signalling tone shall be PCM encoded using the companding law employed at the trunk interface. The off-hook condition at the calling end shall generate the following signalling tone:

– frequency: 2000 Hz ± 10 Hz;

– duration: 1 s ± 0.1 s;

– level: −6 dBm0 ± 1 dB.

ITs numbered 232, 233, 234 and 235 shall be used to route the VOW to a maximum of four corresponding DCMEs. Detection of the signalling tone pertaining to one of the destination directed ITs numbered 232, 233, 234 and 235 shall alert the operator of a pending VOW call. The destination number cross-reference for the VOW ITs is presented in Table 12/G.763.

TABLE 12/G.763

Destination number cross-reference

Destination	Rx number in Frame 56	Bit number for backward alarm in Frame 54	IT number used for VOW	Local IT number for received channel check
1	1	1	232	241
2	2	2	233	242
3	3	3	234	243
4	4	4	235	244

15.10 *In-service monitoring*

15.10.1 *Continuous BER measurements*

Continuous BER measurements shall be performed on the CC. The BER measurement shall make use of the error syndrome of the (24/12) rate 1/2 Golay code specified for protection of the CC in § 11. When the CC BER exceeds 1 in 10^3 (prior to correction) on the basis of a one minute measurement interval, consequent actions shall be taken in accordance with Table 13/G.763. When the CC BER exceeds 1 in 10^5 (prior to correction) on the basis of a 60 s measurement interval, a high BER condition shall be declared for use by the channel check procedure. (The CC BER threshold values are under study.)

TABLE 13/G.763

Fault conditions and consequent actions for the DCME
(Note 1)

Interface side or equipment	Fault conditions (see § 15.11.2)	Consequent actions (see § 15.11.3)							
		Generated on selected trunks toward local ISC			Prompt maintenance alarm indication generated	Generated on bearers to selected correspondent DCMEs			
		Backward alarm indication to remote end	Alarm indication on relevant circuits	AIS on all trunk groups		Indication of fault in affected trunk channels	Backward alarm indication to remote end (bit in DW)	Backward alarm indication to remote end (TS0, bit 3)	AIS on composite bearer
Trunk interface	Failure of incoming trunk group(s)	Yes			Yes (Note 2)	Yes			
	Remote alarm indication received from local ISC (TS0, bit 3)					Yes			
	Trunk group AIS or abnormal circuit supervision					Yes			

TABLE 13/G.763 *(cont.)*

Interface side or equipment	Fault conditions (see § 15.11.2)	Consequent actions (see § 15.11.3)							
		Generated on selected trunks toward local ISC			Prompt maintenance alarm indication generated	Generated on bearers to selected correspondent DCMEs			
		Backward alarm indication to remote end	Alarm indication on relevant circuits	AIS on all trunk groups		Indication of fault in affected trunk channels	Backward alarm indication to remote end (bit in DW)	Backward alarm indication to remote end (TS0, bit 3)	AIS on composite bearer
Bearer interface	Failure of one or more incoming bearer(s)		Yes		Yes (Note 3)		Yes	Yes (Note 5)	
	BER ≥ 10^{-3} in CC		Yes		Yes (Note 4)		Yes		
	Loss of DCME frame, multi-frame alignment		Yes		Yes (Note 4)				
	Remote alarm indication received from corr. DCME (bit in DW)		Yes (if applicable)		Yes (optional)				

TABLE 13/G.763 *(cont.)*

Interface side or equipment	Fault conditions (see § 15.11.2)	Consequent actions (see § 15.11.3)							
		Generated on selected trunks toward local ISC			Prompt maintenance alarm indication generated	Generated on bearers to selected correspondent DCMEs			
		Backward alarm indication to remote end	Alarm indication on relevant circuits	AIS on all trunk groups		Indication of fault in affected trunk channels	Backward alarm indication to remote end (bit in DW)	Backward alarm indication to remote end (TS0, bit 3)	AIS on composite bearer
Bearer interface (cont.)	Remote alarm indication received on incoming bearer (TS0 bit 3)		Yes		Yes (optional)				
	Indication of fault in affected trunk channels received		Yes						
DCME equipment	Functional or power supply failure			Yes (if practical)	Yes				Yes (if practical)

Note 1 – A Yes in the table signifies that an action shall be taken as a consequence of the relevant fault condition. An open space in the table signifies that the relevant action shall not be taken as a consequence of the relevant fault condition, if the condition is the only one present. If more than one fault condition is simultaneously present, the relevant action shall be taken if, for at least one of the conditions, a Yes is defined in relation to this action.

Note 2 – This action shall not be taken if AIS is detected on the incoming trunk groups.

Note 3 – This action shall not be taken if AIS is detected on the incoming bearer(s).

Note 4 – This action shall not be taken if AIS is detected in the CC.

Note 5 – In the multi-destination and multi-clique cases, this action shall be taken upon failure of *all* incoming bearers.

Note 6 – Fault conditions and consequent action for special tandem operation of DCME are under study.

15.10.2 *Channel check procedure*

The channel check procedure provides in-service verification of IT/BC channel assignments between DCME transmit units and DCME receive units.

15.10.3 *Test port*

A capability shall be provided to connect any IT to a TC test port for the purpose of injecting or receiving externally generated test signals. For this purpose, the test port may either be subject to DSI or may be a pre-assigned 64, 40, 32, or optionally 24 or 16 kbit/s channel.

15.11 *Fault conditions and consequent actions*

The philosophy of fault conditions and consequent actions from the point of view of maintenance of digital networks is consistent with that contained in the G.700-Series Recommendations in the *Red Book*, Volume II – Fascicle III.3, Malaga-Torremolinos, 1984.

Alarm conditions and the appropriate consequent actions are defined as follows:

15.11.1 *Normal traffic carrying operating conditions*

The following shall apply when the DCME is carrying traffic, when no digital links are exhibiting fault conditions and when the DCME is also not exhibiting a fault condition:

a) the absence of alarm indications on the DCME terminal shall indicate a normal condition;

b) the means used on the DCME terminal to indicate operating modes or to provide routine information shall be of such form, colour or type that they cannot be confused with alarm conditions.

15.11.2 *Fault conditions* (see Note)

The DCME unit shall detect the following fault conditions:

a) Failure of the incoming trunk primary group(s) – the fault conditions are loss of incoming signal, loss of frame alignment or BER detected in frame alignment signal greater than 1 in 10^3 as defined in Recommendation G.736, for 2048 kbit/s links, and Recommendation G.734 for 1544 kbit/s links.

b) Alarm indication from the remote end, received from the local ISC.

c) AIS detected on incoming primary trunk groups and/or abnormal (alarm) conditions of associated incoming trunk circuits detected or loss of incoming supervision channel. The circuit supervision function may be handled by the SCI.

d) Failure of the incoming bearer signal – the fault conditions are loss of incoming signal, loss of frame alignment or BER detected in frame alignment signal greater than 1 in 10^3 as defined in Recommendation G.736.

e) Bit error rate detected on the CC according to § 15.10 exceeding 1 in 10^3.

f) Loss of DCME frame or DCME multiframe alignment. (The time interval between recognition of an errored condition and a fault declaration is under study, e.g 2.5 s.)

g) Alarm indication from the remote end, received from correspondent DCME unit(s) [see § 15.11.3, f)].

h) Alarm indication from the remote end received on any incoming bearer [see § 15.11.3, g)].

i) Indication of fault in affected TCs detected on IT related alarm bits in the incoming CC data word [see § 15.11.3, e)].

j) DCME failure or DCME power failure.

Note – Optionally, a time delay selectable up to three seconds maximum shall be provided before alarms are initiated or indications are transmitted in fault conditions a, b, c and/or d of § 15.11.2.

15.11.3 *Explanation of consequent actions*

Following the detection of a fault condition, appropriate actions shall be taken as specified in Table 13/G.763.

The consequent actions are listed below:

a) Backward alarm indication to the remote end (towards local ISCs) generated. For 2048 kbit/s primary multiplex trunks, this is done by changing bit 3 of channel time slot 0 from state 0 to state 1 in those frames not containing the trunk frame alignment signal (see Recommendation G.732). For 1544 kbit/s primary multiplex trunks, this is done by forcing bit 2 in every channel time slot to the value 0 or by modifying the S-bit for the 12 frame multiframe or by sending a frame alignment alarm sequence for the 24 frame multiframe (see Recommendation G.733). This consequent action shall be effected as soon as possible.

   The transmit activity detector shall be disabled for the ITs which are associated with the faulty trunk interface and shall set the associated activity indication to inactive (INACT).

b) Alarm indication signal applied on relevant trunk circuits towards local ISC(s), e.g. by AIS on relevant time slots and by means of the out-of-service message through the SCI.

c) AIS on primary trunk groups (all time slots).

d) Prompt maintenance alarm indication generated to signify that performance is below acceptable standards and maintenance attention is required locally. When the AIS (see Note below) is detected, the prompt maintenance alarm indication, associated with loss of frame alignment, excessive error rate in the frame alignment signal and in the bearer assignment message (see § 15.11.2, a), d) and e)), and with the loss of the synchronous data word multiframe alignment (see § 15.11.2) shall be inhibited, while the rest of the consequent actions associated with these four fault conditions shall be followed in accordance with Table 13/G.763.

   Note – The equivalent binary content of the alarm indication signal (AIS) on the trunk groups or time slots is a continuous stream of binary 1s. The strategy for detecting the presence of the AIS will be such that with a high probability the AIS is detectable even in the presence of random errors having a mean error rate of 1 in 10^3. Nevertheless, a signal in which all the binary elements, with the exception of the frame alignment signal, are in state 1, will not be taken as an AIS.

e) Indication of fault in affected TCs generated on the corresponding ITs and forwarded to the correspondent DCME receive unit on a per channel basis by setting the appropriate IT related alarm bits in the CC data word to state 1, (see Table 5/G.763).

f) Alarm indication to the remote end DCME receive unit is generated by changing the appropriate remote alarm indication bit(s) of the CC data word to state 1, (see Table 5/G.763). This will be effected as soon as possible.

g) Backward alarm indication to the remote end (towards remote ISCs) generated. For 2048 kbit/s primary multiplex trunks, this is done by changing bit 3 of Time Slot 0 from the state 0 to the state 1 in those frames not containing the bearer frame alignment signal (see Recommendation G.732). For 1544 kbit/s primary multiplex trunks, this is done by forcing bit 2 in every channel time slot to the value 0 or by modifying the S-bit for the 12 frame multiframe or by sending a frame alignment alarm sequence for the 24 frame multiframe (see Recommendation G.733). This consequent action shall be effected as soon as possible.

h) AIS on bearer signal (all time slots).

15.11.4 *Alarm considerations specific to R2D line signalling*

When alarm conditions occur which require the signalling bits for the affected ITs to be set a=b=1 this shall be specifically notified to the transmit R2 USM for each affected IT.

When alarm conditions are cleared, the new signalling state conditions shall be notified as state changed from a=b=1, for the affected ITs, in the normal manner to the R2 USM.

When certain alarm conditions occur there is a danger of false activity being detected. For these conditions the activity detector should be disabled for the ITs in question, and re-enabled when the alarm condition has been cleared.

The fault conditions and consequent action for R2D line signalling are summarized in Table 14/G.763.

TABLE 14/G.763

Fault conditions and consequent actions for R2D line signalling circuits in the DCME

	Fault conditions (see § 15.11.4.1)	Consequent actions (see § 15.11.4.2)				
		Towards local ISC in circuits concerned	AD disabled	R2D line signalling message	Asynchronous word	Prompt maintenance alarm
Trunk input groups	Loss of incoming signal Loss of frame alignment High BER AIS detected	(Notes 1, 3)	Yes	a=b=1 for refresh only	IT fault bit set in DW (Note 4)	Yes (Note 4)
	Loss of multiframe alignment	(Note 2)	No action	a=b=1 for refresh only	IT fault bit in DW (Note 4)	Yes (Note 4)
	Remote alarm indication from ISC (bit 3 TS0, bit 6 TS16)	No action	No action	No action	IT fault bit in DW (Note 4)	Yes
Bearer input groups	Loss of incoming signal Loss of frame alignment High BER AIS detected	a=b=1	No action	No action	Bearer alarm (Notes 3 and 4) and bearer alarm (bit 3 TS0) if applicable	Yes (Note 4)
	Remote alarm indication received on bearer (bit 3 TS0)	No action	No action	No action	No action	Yes

TABLE 14/G.763 *(cont.)*

	Fault conditions (see § 15.11.4.1)	Consequent actions (see § 15.11.4.2)				
		Towards local ISC in circuits concerned	AD disabled	R2D line signalling message	Asynchronous word	Prompt maintenance alarm
CC decoder	High BER alarm	a=b=1	No action	No action	Bearer alarm (Note 4)	Yes
DCME frame	Loss of DCME frame or multiframe alignment	a=b=1	No action	No action	Bearer alarm (Note 4)	Yes
Rx asynchro-nous data word	Remote bearer alarm Remote IT alarm	No action	No action	No action	No action	No action
DCME	Functional of power supply failure	AIS if possible	No action	AIS if possible	No action	Yes

Note 1 – Backward alarm indication (bit 3 time slot 0).

Note 2 – Backward alarm indication (bit 6 time slot 16).

Note 3 – Inhibited if AIS is present.

Note 4 – See Table 5/G.763.

15.11.4.1 *R2D fault conditions*

The DCME unit shall detect the following fault conditions:

a) Failure of the incoming trunk primary group(s).

The fault conditions are loss of incoming signal, loss of frame alignment, BER greater than 1 in 10^3 detected in frame alignment signal, as defined in Recommendation G.736.

b) AIS detected in incoming primary trunk groups.

c) Loss of multiframe alignment (loss of incoming supervision channel) as defined in Recommendation G.732.

d) Remote alarm indication from local ISC (bit 3, TSO; bit 6, TS16).

The alarm conditions are bit 3 TSO set to 1 in those frames not containing the frame alignment signal and bit 6 TS16 set to 1 in frame 0 of the PCM multiframe, as described in Recommendation G.704.

e) Failure of the incoming bearers primary group(s).

The fault conditions are loss of incoming signal, loss of frame alignment, BER greater than 1 in 10^3 detected in frame alignment signal, as defined in Recommendation G.736.

f) AIS detected on incoming primary bearer group(s).

g) Remote alarm indication received on a bearer (bit 3, TSO).

The alarm condition is bit 3 TSO set to 1 in those frames not containing the frame alignment signal, as described in Recommendation G.704.

h) CC decoder, high BER alarm.

The high BER alarm is raised when the BER in the assignment channel, as defined in § 15.10.1, exceeds 1 in 10^3 prior to correction.

i) Loss of DCME multiframe or DCME frame alignment.

DCME frame and DCME multiframe alignment alarms shall be raised following loss of the unique word sequence, as defined in §§ 11.2 and 11.2.1, in the synchronous bit pattern of the CC.

j) Remote bearer alarm

The alarm condition is the appropriate remote alarm indication bit of the CC asynchronous word set to 1, as defined in Table 5/G.763.

k) Remote IT alarm received in the asynchronous data word.

The alarm condition is the relevant IT identification bit set to 1 in the asynchronous data word (see Table 5/G.763).

l) DCME functional or power supply failure.

Service affecting any internally detected fault condition.

15.11.4.2 *R2D consequent actions*

Further to the detection of a fault condition, appropriate actions shall be taken as specified in Table 14/G.763. However, if redundant equipment is provided and detection of a fault condition is effectively removed by an automatic switchover, prompt maintenance alarm (if applicable) shall be deferred and other consequent actions shall not be taken.

a) Backward alarm indication (bit 3 TSO) towards local ISC.

   This is done by setting bit 3 TSO to 1 in those frames not containing the frame alignment signal. This signal shall not be sent if the fault condition is AIS detected.

b) Backward alarm indication (bit 6, TS16) towards local ISC.

   This is done by setting bit 6 of TS16 to 1 in PCM frame 0 of the multiframe.

c) a=b=1 towards local ISC in circuits concerned.

   The corresponding a and b bits for the affected circuits in TS16 of frames 1 to 15 of the PCM multiframe shall be set to 1. Refer to Recommendation G.704.

d) AIS towards local ISC.

   AIS = alarm indication signal as described in Recommendation G.704.

e) Activity detector disabled.

   The activity detector output shall be set to the inactive state for the ITs concerned and remains in this state as long as the disabling applies.

f) R2D line signalling bits, a=b=1.

   The R2 USM local array a and b bits shall be set to 1 for the relevant circuits (see § 15.11.4.1).

g) Asynchronous word. IT fault bit set in data word.

   For the affected circuits the relevant IT related circuit supervision bits of the asynchronous data word shall be set to 1. Refer to Table 5/G.763.

h) Asynchronous word. Bearer alarm.

   For the affected bearer the relevant backward bearer alarm of the asynchronous data word is set to 1. Refer to Table 5/G.763.

i) Prompt maintenance alarm

   An audible/visual alarm indication to alert the operator to the presence of a fault condition. To be specified by the users.

16 Glossary

ADPCM	Adaptive Differential pulse code modulation
AIS	Alarm indication signal
BC	Bearer channel
BER	Bit error ratio
BMI	Bit map implementation process
B8ZS	Bipolar eight zero substitution
UCA	Capacity available
CC	Control channel
UCNA	Capacity not available

DAF	Dynamic assignment function
DCME	Digital circuit multiplication Equipment
DCMG	Digital circuit multiplication Gain
DCMS	Digital circuit multiplication System
DDI	Direct digital interface
DEC	Decoder control process
DEMUX	Demultiplex
DLC	Dynamic load control
DNI	Digital non-interpolated
DSH	Dual seizure handling
DSI	Digital speech interpolation
DW	Data word
D/S	Data/speech
ENC	Encoder control process
FDX	Full duplex
F-bit	Framing bit
FOF	Freeze-out fraction
HDX	Half duplex
HL	High load
HSC	Hangover control and signal classification process
IDR	Intermediate data rate
IG	Interpolation gain
IPS	Input processing and service request generation block
ISC	International switching centre
ISUP	ISDN User Part
IT	Intermediate trunk
LL	Low load
LRE	Low rate encoding
LSB	Least significant bit
MCH	Map change handler
MOS	Mean opinion score
MSB	Most significant bit
MUX	Multiplex
NRZ	Non-return-to-zero
O&M	Operations and maintenance
QDU	Quantization distortion unit
QPSK	Quadrature phase shift keyed

RAG	Request handling & assignment information generation process
RCP	Receive channel processing block
RUD	Receive channel status update and overload channel decoding process
SBC	SC bit map creation process
SCI	Switching centre interface
SRH	Service request handling block
SS	Signalling system
STI	Statistics time interval
TC	Trunk channel
TCH	Transparent circuit handler
TCP	Transmit channel processing
TDMA	Time division multiple access
TG	Transcoding gain
TMN	Telecommunications management network
TS	Time slot
TTF	Test time frame
USM	User signalling module
UW	Unique word
VBR	Variable bit rate
VOW	Voice orderwire
ZBTSI	Zero byte time slot interchange
ZCS	Zero code suppression

List of internal/external messages/indications

AD64	Activate DLC for 64-kbit/s traffic
ADVD	Activate DLC for voice/voice-band data traffic
DD64	De-activate DLC for 64-kbit/s traffic
DDVD	De-activate DLC for voice/voice-band data traffic
R64	Release 64-kbit/s circuit
R64Ack	Release 64-kbit/s circuit Acknowledged
S64	Seizure/Select 64-kbit/s circuit
S64Ack	Seizure/Select 64-kbit/s positive Acknowledged
S64NAck	Seizure/Select 64-kbit/s Negative Acknowledged
SA	Capacity for speech available
SNA	Capacity for speech not available
UCA	Capacity for 64-kbit/s unrestricted available
UCNA	Capacity for 64-kbit/s unrestricted not available
VDA	Capacity for 3.1-kHz data available
VDNA	Capacity for 3.1-kHz data not available

ANNEX A

(to Recommendation G.763)

Examples of DCME transmit/receive unit structure and SDL diagrams

A.1 *An example of a DCME transmit unit structure*

An example of a DCME transmit unit structure is shown in Figure A-1/G.763. Compliance with this transmit unit structure will permit the DCME transmit function to be tested with INTELSAT IESS–501(Rev.2) compliant DCME test equipment and software protocol references. This structure is based on a non-mandatory partitioning of functions and definition of signals.

Some of the functional blocks in Figure A-1/G.763 are internal to the DCME transmit unit structure, while others are external but provide required interface signals. The blocks that belong to the transmit unit structure are:

a) *Transmit Activity Detector* – This block produces an IT Active/Inactive input for the Transmit Channel Processing Function. The detector has no built-in hangover, since the hangover control task is performed by the transmit Channel Processing function. The specification for the Transmit Activity Detector is provided in § 12.1.

b) *Data/Speech Discriminator* – This block recognizes voice and single tones as speech and recognizes data and 2100 Hz tone as data. Its specifications are provided in Annex B.3.

c) *2400 Hz Tone Detector* – This block provides a detection indication when the 2400 Hz signalling tone is present. The specification for this block is provided in Annex B.4.

d) *The Transmit Channel Processing (TCP) Function* – This function consists of an ensemble of interconnected processes. Its task is to process the inputs received from blocks a), b) and c) above as well as inputs originating from external blocks. The TCP function produces three outputs, each directed to the Encoder Unit, the Assignment Message Encoder, and the BC Bit Assignment Unit(s). These blocks are defined below.

e) *The Encoder Unit* – This unit consists of a bank of ADPCM encoders which can be connected to any transmit IT and to any BC. Each BC can carry 8, 5, 4, 3, or optionally 2 bits per PCM sampling cycle or can be disconnected from the coders.

   The encoders can be set to 8/5/4/3/optionally 2 bit mode of operation and can be initialized to a known state. The IT and BC connection/disconnection information for each encoder, as well as the mode of operation selection and the initialization signal are provided by the TCP function.

f) *The Assignment Message Encoder* – This unit encodes the IT-to-BC association, and the channel type (data/speech, or 64 kbit/s) into the format specified in § 11. The necessary information is provided by the TCP function.

g) *The BC Bit Assignment Unit* – This unit is connected to the output of the Encoder Unit (BCs). The BC Bit Assignment Unit maps the bits of each BC onto the bits of the bearer pool channels. The bit map for the bearer pool channel association is provided by the TCP function.

The blocks a), b) and c) operate on a single IT in the representation in Figure A-1/G.763. Conceptually, these blocks must be considered as time multiplexed, scanning all relevant ITs.

The blocks which are external to the transmit Side Structure but provide required inputs are the following:

a) *Assignment Message Decoder* – Information on the data/speech type of the received IT is passed to the TCP function together with the corresponding transmit IT number. The receive IT/transmit IT association is performed by the Receive Channel Processing Function.

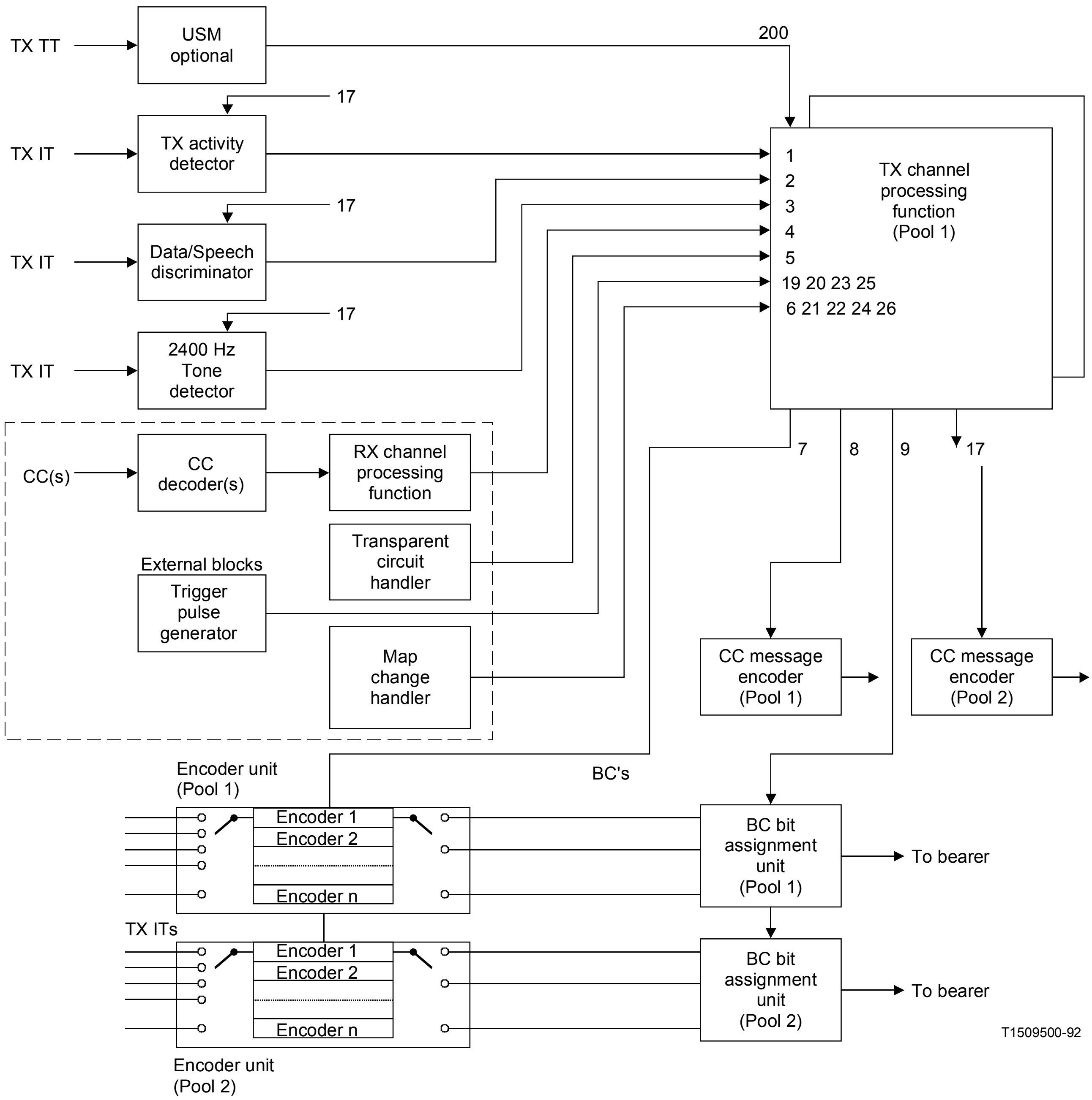
b) *Transparent Circuit Handler* – This process either forwards a request to the TCP function for a transparent 64 kbit/s channel or sends a message releasing the channel. The Transparent Circuit Handler Process is specified in § 8.

c) *Map Change Handler* – The Map Change Handler (MCH) is a process which controls the configuration data for the DCME. At start-up, this process issues signals making it possible to configure the system correctly. The same is done at a map change instant. Refer to §§ 15.1 and 15.6.

d) *Trigger Pulse Generator* – This unit provides a periodic 2 ms timing reference signal to the processing functions of the transmit unit (see Note).

e) *User Signalling Module (Optional)* – This user signalling module (USM) generates signalling state change signals. The specification of the USM is at the user's option.

Note – The trigger pulse generator will also provide a sync trigger pulse to identify the first frame of a DCME multiframe. This permits a capability to transfer out-of-band signalling within the control channel.



Note – Legend for signal paths shown in Table A-1/G.763.

FIGURE A-1/G.763

DCME transmit unit structure

TABLE A-1/G.763

Legend for transmit unit signal paths

Signal path No.	Signal type/message	Definition
1	"Act", "Inact"	§ A.1.1.1.1
2	"Data-detect", "Voice-detect"	§ A.1.1.1.1
3	"Signaldetect"	§ A.1.1.1.1
4	"Rxdata"	§ A.1.1.1.1
5	"Transpreq", "Transprel"	§ A.1.1.1.1
6 (and 21, 22, 24, 26)	Process-Reset from MCH	§ A.1.1
7	"Setcod"	§ A.1.1.2.4
8	"Assign"	§ A.1.1.2.1
9	"Addressmap-for-BCs"	§ A.1.1.2.3
10, 11	Not used	
12	"Voice", "Voiceinact", "Data", "Datainact", "Transp", "Discreq"	§ A.1.1.1.1
13	"Assign", "Reinsert", "Remove", "Seizesc", "Seizebank", "Releasesc", "Release"	§ A.1.1.2.1
14	"BC Bit Map"	§ A.1.1.2.2
15	"Mode Map"	§ A.1.1.2.2
16	"Assign-enc", "Release-enc", "Set-pre"	§ A.1.1.2.1
17	"Resetact", "Resetsignaldetect", "Default-Voice", "Default-Data"	§ A.1.1.1.1
18	No used	
19	Trigger Pulse, Sync Trigger Pulse	§ A.1.1.2.1
20, 23, 25	Trigger Pulse	
200	Change	§ A.1.1.2.1

A.1.1 *Transmit Channel Processing Function*

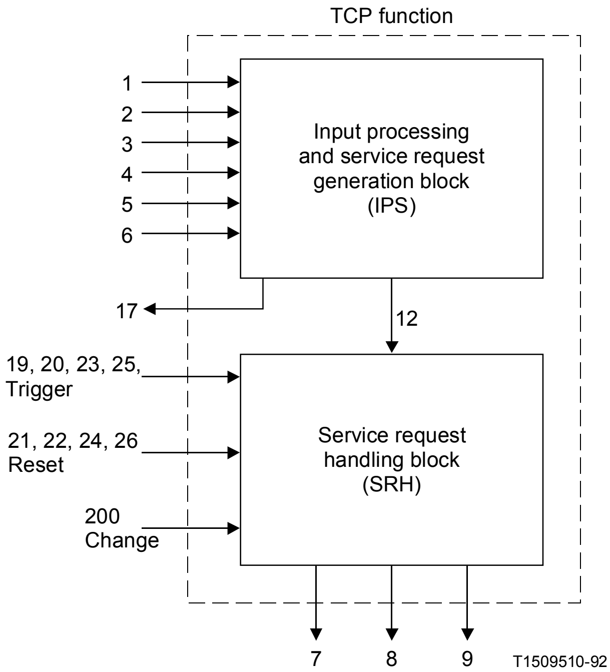
The Transmit Channel Processing Function (TCP) interfaces with the other elements of the transmit Side Structure as shown in Figure A-1/G.763. Each interface signal is identified in Figure A-1/G.763 with a specific number. The signal path originating from the MCH carries a reset signal to five different TCP processes and, therefore, takes five different numbers. The signal path originates from the trigger pulse generator, carries trigger signals to four different processes, and therefore, takes four different numbers.

The TCP function monitors the status of each IT and takes consequent actions. The status of each IT is used by the internal TCP processes to generate the information required by the Encoder Unit, the Assignment Message Encoder and the BC Bit Assignment Unit. Reset signals are provided to the internal blocks previously listed under a), b) and c).

The internal structure of the TCP functions is shown in Figure A-2/G.763. The TCP function contains the Input Processing and Service Request Generation Block (IPS) and the Service Request Handling Block (SRH).

A.1.1.1 *The IPS Block*

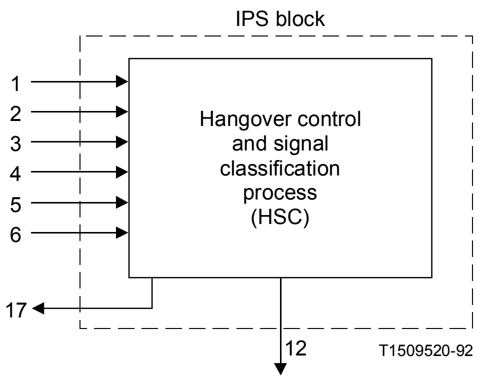
The IPS Block input/output connections are shown in Figure A-2/G.763. The IPS Block processes the TCP inputs (signal path 1 through 6) and generates IT status transition information (signal path 12) for the other block (SRH) of the TCP function. The IPS Block also generates a reset signal (signal path 17) for the transmit Activity Detector, the Data/Speech Discriminator and the 2400 Hz tone detector. The IPS Block must be considered as time multiplexed, processing all the ITs of the pool.



Note – Legend for signal paths shown in Table A-1/G.763.

FIGURE A-2/G.763
TCP function

The internal structure of the IPS Block is shown in Figure A-3/G.763. This block performs the Hangover Control and Signal Classification Process (HSC) on each IT.



Note – Legend for signal paths shown in Table A-1/G.763.

FIGURE A-3/G.763
IPS block

A.1.1.1.1 *The HSC process*

The HSC input/output connection is shown in Figure A-3/G.763. The HSC Process receives the IPS inputs 1 through 6, processes them, and provides an input (signal path 12) to the SRH Block. The HSC Process resets (signal path 17) the detectors and discriminator. The process-reset signal from the MCH (signal path 6) terminates the HSC Process at a map change instant. The above signal paths carry the following messages:

– Signal Path 1: Activity detected ("Act"), Inactivity detected ("Inact").

– Signal Path 2: Data Detected ("Data-detect"), Speech detected ("Voice-detect").

– Signal Path 3: 2400 Hz tone detected ("Signaldetect").

– Signal Path 4: Receive data detected ("Rxdata").

– Signal Path 5: Transparent channel request ("Transpreq"), Transparent channel release ("Transprel").

– Signal Path 6: Process-Reset signal from the MCH.

– Signal Path 12: The messages (related to changes of signal classification) are "Voice(IT)", "Voiceinact(IT)", "Data(IT)", "Datainact(IT)" and "Transp(IT)" and "Discreq(IT)".

– Signal Path 17: Carries the following messages:

  a) "Reset-act" (set activity detector to inactive).

  b) "Reset-Signaldetect" (reset 2400 Hz detector to nodetect).

  c) "Default-Voice" (set discriminator to voice).

  d) "Default-Data" (set discriminator to data).

The HSC process should perform signal classification and hangover control as specified below.

a) Initially, this process should declare an IT either "pre-assigned", if so designated by the configuration data, or "voice-inactive", if subject to DSI.

b) Whenever a "Transpreq" message is received, the IT should be classified as "Transparent" and should remain in this condition until a "Transprel" message is received, in which case the signal classification should change to "Voice-inactive".

c) If the IT is active and of the voice/signalling type and the "Data-detect" message is received from the Data/Speech Discriminator, the IT should be classified as "Data-active". The same applies to the case of reception of "Rxdata" from the DCME receive unit as long as the 1 s delay timer (hold mode, defined later) is not running. No action should be taken if the timer is running.

   If the IT is inactive (hangover timer either expired or running), and of the data type and the message "Act" is received from the Activity Detector, the IT should also be classified as "Data-active".

d) If the IT is inactive (hangover time expired) and of the voice/signalling type and the "Rxdata" message is received, the IT should be classified as "Data-inactive", as long as the 1 s delay timer is not running. No action should be taken if the timer is running (hold mode).

   If the IT is of the data type and the hangover timer expires, the IT should also be classified as "Data-inactive".

e) If the IT is inactive and in the hold mode and the 1 s delay timer expires, the IT should be classified as "Voice-inactive".

   If the IT is of the voice/signalling type and the hangover time expires, the IT should also be classified as "Voice-inactive".

f) If the IT is inactive (hangover timer either expired or running) and of the voice type and the "Act" message is received from the Activity Detector, the IT should be classified as "Voice-active".

g) If the IT is active and of the data type and the message "Voice-detect" is received from the Data/Speech Discriminator, a 1 s delay timer should be started and the IT should be classified as "Voice-active-hold".

   If the 1 s timer is running for an inactive voice IT (hangover timer either expired or running) and the "Act" message is received, the IT should also be declared "Voice-active-hold".

h) If the IT is active and of the voice type and the message "Signaldetect" is received from the signalling tone detector, the IT should be classified as "Signalling-active".

   If the IT is of the signalling type and the hangover timer is running and the message "Act" is received, the IT should also be classified as "Signalling-active".

i) If the IT is active and of the data type and the "Signaldetect" message is received, a 1 s delay timer should be started and the IT should be classified as "Signalling-active-hold".

   If the 1 s timer is running for an active voice IT and the "Signaldetect" message is received, the IT should be classified as "Signalling-active-hold".

j) If the IT is inactive and of the voice type, the hangover timer is running and an "Rxdata" message is received, the data detector shall be set to data and the IT should enter the "Wait-for-data" state.

   If the IT is inactive and of the signalling type, the hangover timer is running and an "Rxdata" message is received, the data detector should be set to data and the IT should enter the "Wait-for-data" state.

   If the hangover timer expires while the IT is in the "Wait-for-data" state, the "Voiceinact" message should be sent and the IT should be classified as "Datainactive".

   When activity first terminates on an IT, classified as "Data-active", the "initial data hangover" value should be used. Its duration should be settable to a maximum of 14 s. After the first expiration of the "initial data hangover", the "second data hangover" should be brought into use. This hangover should also be settable to a maximum of 14 s, but it is expected that in most cases, it will be set to a value significantly shorter than the initial data hangover. This permits higher efficiency of utilization of the return link for facsimile transmission.

   When activity terminates on an IT classified as "Voice-active" or "Voice-active hold", the Voice Hangover Value should be used. When activity terminates on an IT classified as "Signalling-active" or "Signalling-active hold", the Signalling Hangover value should be used. The Voice and Signalling Hangover values should be in accordance with the hangover masks specified in § 12.

   The message "Voice(IT)" is associated with the transition to "Voice-active", "Voice-active-hold" and "Signalling-active-hold". The message "Voice-inact(IT)" is associated with the transition to "Voice-inactive" and "Voice-inactive-hold". The messages "Data(IT)" and "Datainact(IT)" are associated with the transitions to "Data-active" and "Data-inactive", respectively. The message "Discreq(IT)" is generated whenever a transition occurs from "transparent" to "Voice-inactive".

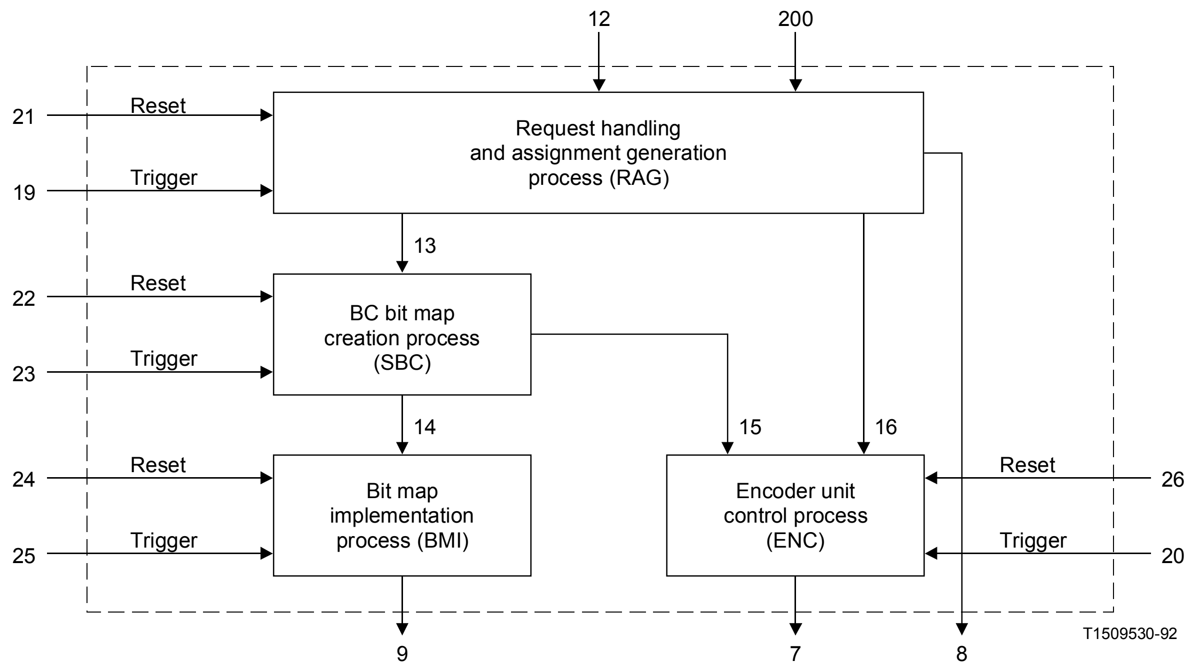
   The reset messages carried by signal path 17 should be generated at initialization (except "Default-Data").

The reset messages should also be generated during operation when the active/inactive or channel type classification changes for causes other than a corresponding change in the detector/discriminator output.

A.1.1.2 *The SRH Block*

The SRH Block input/output connections are shown in Figure A-2/G.763. The SRH Block processes the IT transition information (signal path 12) received from the IPS block. It also generates assignment information for the Assignment Message Encoder (signal path 8), encoder connect/disconnect and mode information for the Encoder Unit (signal path 7) and the BC Bit Map for the BC Bit Assignment Unit (signal path 9).

The internal structure of the SRH Block is shown in Figure A-4/G.763. This block contains the Request Handling and Assignment Generation Process (RAG), the BC Bit Map Creation Process (SBC), the Bit Map Implementation Process (BMI), and the Encoder Unit Control Process (ENC).



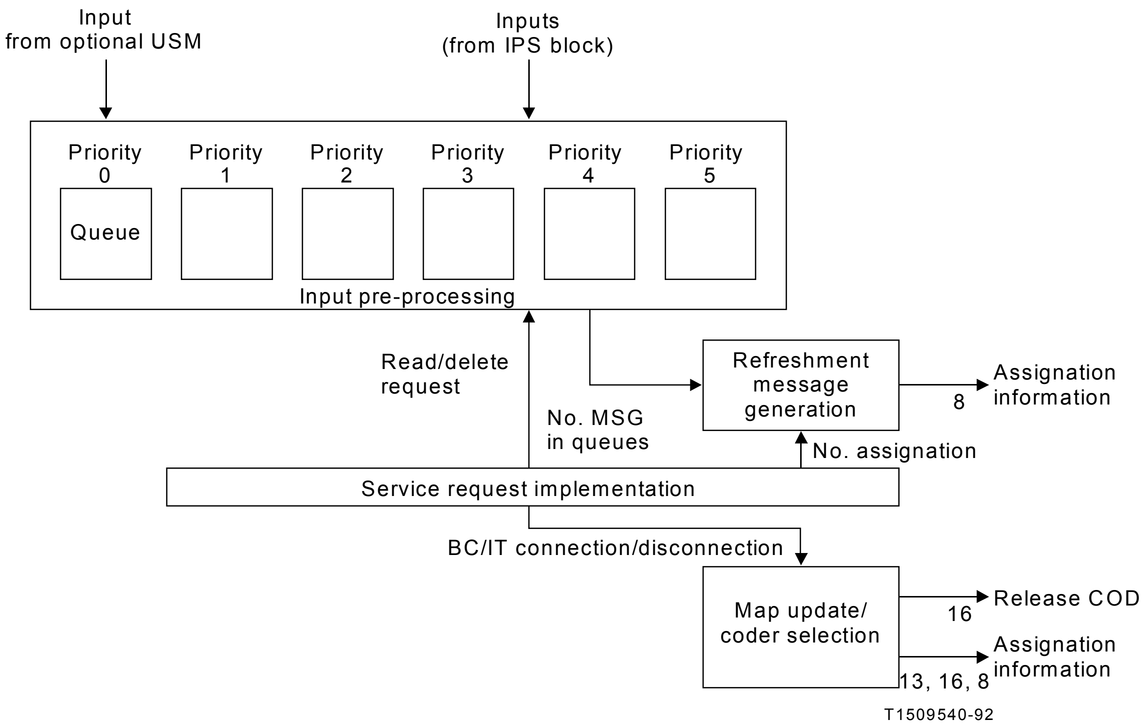
Note – Legend for signal paths shown in Table A-1/G.763.

FIGURE A-4/G.763
SRH block

A.1.1.2.1 *The RAG Process*

The RAG Process input/output connection is shown in Figure A-5/G.763. The RAG Process also receives an input signal (signal path 21) from the MCH, which resets the process at the map change instant. It also receives a trigger pulse (signal path 19). The trigger pulse provides a timing reference once per DCME frame for the RAG process. When required, the RAG process will also receive a signal from the optional USM. This signal (signal path 200) contains the “change (IT)” message (see Note).

Note – This signal permits a capability to transfer out-of-band signalling within the control channel.



Note – Legend for signal paths shown in Table A-1/G.763

FIGURE A-5/G.763

RAG process tasks

The RAG Process receives IT transition information from the IPS Block (signal path 12) and generates the following outputs:

– Signal Path 8: This signal path carries the "Assign" message which contains assignment information needed by the Assignment Message Encoder (and by the other processes of the block). This message is also present on signal path 13. The message contains a BC number, an IT number, the BC type, and encoder number with the format (BC, IT, type, encoder number). The Assignment Message Encoder extracts the relevant information elements from the "Assign" message and adds additional information, as required by the control channel message structure (specified in § 11.1). In the DCME frames which are used by the optional USM, the BC number should be 255, type data and encoder number 0.

– Signal Path 13: This signal path carries the following messages:

"*Assign*" – This is the same message as in signal path 8.

"*Reinsert (BC)*" – This message is used to reinsert a BC into the overload channel generation map within the SBC process when an implicit disconnect of a data call has occurred (see § A.1.1.2.2).

"*Remove (BC)*" – It removes an implicitly disconnected overload channel from the SBC overload BC list (see § A.1.1.2.2).

"*Seizesc (BC, encoder no., enc. mode)*" – It generates a fixed association between a BC number and an ADPCM encoder number, for a pre-assigned channel in the SBC Process (see § A.1.1.2.2). The ADPCM encoder mode could be 8/5/4/or optionally 3 or 2 bits per sample. This message is transmitted immediately after initialization.

"*Seizebank (BC)*" – This message notifies the SBC Process that a certain BC has been seized as a bit bank (see § A.1.1.2.2). It is transmitted immediately after initialization.

"*Releasesc (BC)*" – This message releases a bit bank connection and is given to the SBC process (see § A.1.1.2.2).

"*Release (enc. no.)*" – This message updates the ADPCM encoder map within the SBC process (see § A.1.1.2.2).

– Signal Path 16: This signal path carries the following messages:

"*Assign-enc (BC, IT, type)*" – This is used to give channel connection information to an ENC process.

"*Release-enc*" – This message causes the encoder to release any connection.

"*Set-pre (mode, IT)*" – This message causes seizure of an encoder for a pre-assigned connection. The mode can be 8, 5, 4 or optionally 3, or 2 bit. This message is transmitted immediately after initialization phase.

The RAG Process can be functionally divided into four tasks, namely, Input Pre-Processing, Service Request Implementation, Refreshment Message Generation, and Map Update/Coder Selection. This is illustrated in Figure A-5/G.763.

The Input Pre-Processing Task processes the input IT transition information, and either updates the channel type (discussed below), or generates service requests to be placed in prioritized queues.

The Service Request Implementation Task services the requests in the queues by assigning ITs to BCs or deleting the existing IT-to-BC association.

The Map Update/Coder Selection Task updates a central Resource Map based on the action of the Service Request Implementation Task. The Resource Map contains information to identify the IT-to-BC association (including disconnected BCs and ITs), the BC type, and the IT-to-ADPCM encoder association. The possible BC types are the following:

a) "Voice" – Indicates the BC carries a voice signal that is either active or in the hangover period;

b) "Data" – Indicates that the BC carries a data signal that is either active or in the hangover period;

c) "Transparent" – Indicates that the BC carries a transparent call;

d) "Disconnected" – Indicates that the BC is not connected;

e) "Voice-available" – Indicates that the BC is currently connected to a speech IT but could be used for a new assignment;

f) "Data-available" – Indicates that the BC is currently connected to a data IT but could be used for a new assignment;

g) "Pre-assigned" – Indicates that the BC is permanently assigned to an IT, in accordance with the DCME configuration data.

h) "Bank" – This 4-bit BC can be used to obtain the LSBs of up to four data channels (the bit bank concept is discussed later).

The ADPCM encoder selection and the generation of the messages carried by signal paths 8, 13 and 16 is functionally assigned to this task.

The Refreshment Message Generation Task generates assignment information for the Assignment Encoder when no higher priority Assignment message generation is required.

A.1.1.2.1.1 *Input pre-processing task*

The messages received from the IPS Block (signal path 12) contain signal transition information for each IT. When using the optional USM, messages (signal path 200) will be received. The Input Pre-Processing Task performs the following actions:

a) processes the IT transition information and generates service requests;

b) places the service requests in prioritized queues, which are accessed by the Service Request Implementation Task.

Six queues are established, each with an associated priority:

a) Priority 0 Queue – It stores the IT number contained in a "change (IT)" message;

b) Priority 1 Queue – It stores the 64 kbit/s IT disconnect requests;

c) Priority 2 Queue – It stores internally generated requests for disconnection of overload BCs.

d) Priority 3 Queue – It stores the 64 kbit/s IT connection requests;

e) Priority 4 Queue – It stores the request for assignment of data channels;

f) Priority 5 Queue – It stores the requests for assignment of voice channels.

When a "data-inact(IT)" message is received, the type of the BC connected to the IT shall be updated to "data-available", unless there is another request in the queue for the same IT and the BC type is "voice". In this case, the BC type shall be changed to "voice-available".

When a "voice-inact(IT)" message is received, the type of the BC connected to the IT shall be updated to "voice-available", unless there is another request in the queue for the same IT and the BC type is "data". In this case, the BC shall be changed to "data-available". If the BC is in the overload range, a disconnect request shall be stored in Priority 2 queue.

When a "Voice(IT)" message is received, the type of BC connected to this IT should be checked. If the type of BC is "Voice" or "Voice-available", the BC type should be changed to voice and no request shall be generated. If the BC type is other than "Voice-available", a request should be stored in the Priority 5 queue.

When a "Data(IT)" message is received, the type of the BC connected to this IT should be checked. If the type of BC is "Data" or "Data-available", the BC type should be changed to data and no request should be generated. If the BC type is other than "Data-available", a request should be stored in the Priority 4 queue.

When a "Transp(IT)" message is received, a request should be stored in the Priority 3 queue.

When a request pertaining to IT arrives, and there is a request for this IT in any of the queues, the stored request should be deleted if in Priority queues 2 to 5 and should be maintained if in Priority queue 1. If the stored request is in Priority queue 1 and the new request is also for Priority queue 1, the new request should be deleted. A message for Priority 0 queue should be stored in Priority 0 queue without checking any other queue.

A.1.1.2.1.2 *Service request implementation task*

At the time of reception of the trigger pulse, if there are messages in the queues, and if there are no 64 kbit/s or data assignments in progress, the Priority 1 Queue should be addressed. If the message count for this queue is one or more, the RAG Process should address the first request in the queue (first in, first-out order) and delete it when serviced. If the message count is 0, the next lower priority queue should be addressed. The same procedure should be repeated for the other queues. At the next trigger pulse, the cycle should restart from Priority 1 Queue. The actions to be taken for the implementation of the different service requests are specified separately below. At the first frame of a multiframe the trigger pulse is replaced by a sync trigger pulse (refer to § 11). In the case where the optional USM is not used, the Priority 0 Queue should not be addressed. In the case where the USM is used, the Priority 0 Queue should be addressed in DCME frames 0, n, 2n, etc. (i.e., every nth frame), of the DCME multiframe where n is a variable which may be selected by the user. Priority 1 through 5 queues should be addressed in the remaining DCME frames.

a) Change Message – If the optional USM is used, the Priority 0 Queue should be addressed in DCME frames 0, n, 2n, etc. (i.e., every nth DCME frame) of the DCME multiframe where n is a variable which may be selected by the user. Upon servicing the request, the message should be deleted from the Priority 0 Queue.

b) Discreq Requests – The request at the top of Priority 1 Queue should be addressed. An assignment should be generated which disconnects the IT. This request should be deleted from the queue.

c) BC Disconnect Requests – The request at the top of Priority 2 Queue should be addressed. An assignment should be generated which disconnects the IT. The service request should be deleted.

d) Transp Requests – The request at the top of Priority 3 Queue should be addressed. The IT for which the request was generated should be checked to determine whether connected or disconnected.

   If the IT is connected, a count of the usable bits in the pool should be taken to determine whether enough capacity exists to accommodate the additional bits required. If no capacity exists, the assignment which disconnects the available BC (and the associated IT) should be generated. The RAG Process should address the Priority 1 Queue again at the next occurrence of the trigger pulse.

   If the IT is connected and enough capacity exists to accommodate the additional bits, the BC number of the connected bearer channel (number k) should be checked to determine whether it is even or odd. If k is even, the next higher BC (number k+1) should be examined. If k is odd, the next lower BC (number k–1) should be examined. The objective is to allocate the first nibble (containing the MSB) of the 64 kbit/s channel to an even numbered BC. If k is even and BC k+1 is contained in the pool, the type of BC k+1 should be checked. If the type is "Disconnected", "Data-available" or "Voice-available", the 64 kbit/s IT should be assigned to BC k (and by implication, to BC k+1). If the type of BC k+1 is either "Data" or "Voice", a re-assignment of this IT will be required. This should be done by invoking the

Search-Data Procedure (§ A.1.1.2.1.6) or the Search-Voice Procedure (§ A.1.1.2.1.7), respectively. If the type of BC k+1 is either “Bank” or “Pre-assigned”, or if BC k+1 is not contained in the pool, the Search-Transparent Procedure should be invoked (as specified later).

A similar approach should be taken if k is odd. In this case the BC to be examined is k−1.

If the IT is disconnected, the number of usable bits in the pool should be counted to determine whether the request can be accommodated (8 bits are required). If the request can be accommodated, the Search-Transparent Procedure should be invoked. If the request cannot be accommodated, a Refreshment Message (§ A.1.1.2.1.3) should be generated.

The Search-Transparent Procedure is invoked (§ A.1.1.2.1.5) to select a suitable BC pair for the assignment. The Search-Transparent Procedure delivers the encoder number to be used (see § A.1.1.2.1.4 for the encoder selection) and the values for 11 variables (see Table A-2/G.763). These variables consist of the two bearer channels (bc and bc+1) selected for allocation of the 64 kbit/s IT, the two ITs (nrvl and nrv2) occupying bearer channels bc and bc+1, the bearer channels (bcv1 and bcv2) to which nrv1 and nrv2 can be re-assigned, the two ITs (nrv3 and nrv4) occupying bearer channel bcv1 and bcv2 and the overload bearer channels (bcv3 and bcv4) to which the ITs nrv3 and nrv4 can be re-assigned. The bearer channel bc is an even-numbered BC. The variable “success” gives the result of the search. If the search is successful, success is set to TRUE, or else FALSE.

TABLE A-2/G.763

Search-Transparent Procedure Variables

nr:	No. of IT requesting a 64 kbit/s transparent channel
cd:	Selected ADPCM encoder No. for the request
bc:	BC No. for allocation of the 1st nibble of the 64 kbit/s IT
bc+1:	BC No. for allocation of the 2nd nibble of the 64 kbit/s IT
nrv1:	No. of the IT currently occupying bc
nrv2:	No. of the IT currently occupying bc+1
bcv1:	BC No. for re-assignment of nrv1
bcv2:	BC No. for re-assignmnent of nrv2
nrv3:	No. of the IT currently occupying bcv1
nrv4:	No. of the IT currently occupying bcv2
bcv3:	BC No. for re-assignment of nrv3
bcv4:	BC No. for re-assignment of nrv4
Success	Result of the search (TRUE or FALSE)

Note 1 – nrv1, nrv2, nrv3, nrv4 = 0 indicates that IT re-assign/disconnect is not necessary.

Note 2 – bcv1, bcv2, bcv3, bcv4 = 0 indicates that these BCs are not required for reassignment.

Note 3 – bcv3 and bcv4 are overload BCs.

Note 4 – bc is an even number.

Note 5 – The first nibble of the 64 kbit/s IT contains the MSB.

Note 6 – The variable nr is the only input variable.

Note 7 – bc+1 is derived from the variable bc.

Whenever a nrv variable (nrv1 through nrv4) is 0, re-assignment/disconnection of the IT is not required. Whenever a bcv variable (bcv1 through bcv4) is 0, the BC is not required for re-assignment.

The IT nrv4 should be examined first. If nrv4 is 0 (IT re-assignment/disconnection not required), nrv3 should be examined. If nrv4 is different from 0 (IT re-assignment/disconnection required) and bcv4 is also different from 0 (BC required for re-assignment of nrv4), nrv4 should be re-assigned to bcv4. At the next trigger pulse, nrv3 should be examined.

If nrv4 is different from 0 and bcv4 is 0 (BC not required for re-assignment of nrv4), nrv4 should be (internally) disconnected and nrv3 should be examined.

Equivalent steps should be implemented for nrv3 and nrv2.

When nrv1 is examined, if equal to 0 (IT re-assignment/disconnection not required), the 64 kbit/s IT should be assigned to bearer channel bc.

If nrv1 is different from 0 (IT re-assignment/disconnection required) and bcv1 is also different from 0 (BC required for re-assignment of nrv1), nrv1 should be re-assigned to bcv1. At the next trigger pulse, the 64 kbit/s IT should be assigned to bearer channel bc.

If nrv1 is different from 0 (connected) and bcv1 is 0 (BC not required for re-assignment of nrv1), nrv1 should be (internally) disconnected and the 64 kbit/s IT should be assigned to bearer channel bc.

At implementation, the service request should be deleted from Priority 3 Queue.

e) Data Requests – Five bit encoding is required for the transmission of a data channel. Four bits are obtained by assigning the data IT to a 4-bit BC in the normal BC range. The 5th bit (LSB) is obtained from a pool of 4 bits referred to as the bit bank. Special 4-bit BC channels of the "Bank" type are created in the bearer frame for this purpose. The criteria for creation of bank channels are specified in Table A-3/G.763.

TABLE A-3/G.763

Criteria for bank creation/deletion

Creation of bit bank (at new data channel assignment)		
"Data-avail" BC present?	Yes	Bank not required
	No	Bank required if $nb < \frac{nd}{4}$
Deletion of bit bank (see Note)		
$nb \geq \frac{nd}{4} + 1$		Delete highest numbered bank

Note – If bank was just created, wait 2 DCME frames before applying deletion criterion. If during the 2 DCME frame wait period a USM signalling message is sent, wait one additional DCME frame.

nd: Number of data channels in use and requested (pre-assigned and DSI).

nd: Number of banks in use.

The request at the top of the Priority 4 Queue should be addressed. First, it should be determined whether a new bit bank is required. Then, the IT for which the request was generated should be checked to determine whether connected to a BC.

If the IT is connected, a bit count should be made to determine whether enough bearer capacity exists for the allocation of the data IT (including the creation of an additional "Bank" channel if required).

If sufficient capacity exists and the connected BC is in the normal range, and a new bit bank is not required, an assignment of the data IT to the connected BC should be made.

If a bit bank is required, it should be assigned in the same way as a data channel by invoking the Search Data Procedure (as specified later). An assignment message connecting the selected BC to IT No. 250 should be generated. At the next trigger pulse, the data IT should be assigned to the connected BC.

If sufficient capacity is available, but the connected BC is in the overload range, the data IT should be assigned by invoking the Search Data Procedure (specified later).

If sufficient capacity is not available and the IT is connected, a disconnect message should be generated.

If the IT is disconnected, a count of the usable bits in the pool should be made to determine whether enough capacity exists to allocate the data call. If sufficient capacity is not available, a refreshment message should be generated. If sufficient capacity is available and a new bit bank is not required, the Search Data Procedure (§ A.1.1.2.1.6) should be invoked to select a suitable BC for the assignment of the data IT. If the bit bank is required, it should be assigned first, and then the data IT should be assigned as specified below. The Search Data Procedure delivers the ADPCM encoder number to be used (see § A.1.1.2.1.4 for the ADPCM encoder selection) and three values for the three variables defined in § A.1.1.2.1.6.

The IT nrv should be examined. If nrv is 0 (BC disconnected), the data IT should be assigned to bearer channel bc.

If nrv is different from 0 (BC connected) and bcv is also different from 0 (re-assignment required), nrv should be re-assigned to bcv. At the next trigger pulse, the data IT should be assigned to bearer channel bc.

If nrv is different from 0 (BC connected) and bcv is 0 (re-assignment not required), nrv should be (internally) disconnected and the data IT should be assigned to bearer channel bc.

The service request, at implementation, should be deleted from Priority 4 Queue.

f) Voice Requests – The request at the top of Priority 5 Queue should be addressed. The IT for which the request was generated should be checked to determine whether connected to a BC.

If connected and the BC type is "Available", the IT should be assigned to the available BC. If the BC type is "Data", an assignment should be made to that BC.

If the IT is disconnected, the usable bits in the pool should be counted to determine whether enough capacity exists to accommodate the voice call. If sufficient capacity does not exist, a refreshment message should be generated.

If sufficient capacity exists, the Search-Voice Procedure should be invoked to select a suitable BC for assignment. This procedure delivers the ADPCM encoder number to be used (see § A.1.1.2.1.4 for the ADPCM encoder selection) and the values of the variables nrv and bc, defined in § A.1.1.2.1.3.

If nrv is 0 (BC disconnected), the voice IT should be assigned to bearer channel bc. If nrv is different from 0 (BC connected), nrv should be (internally) disconnected and the voice IT should be assigned to bearer channel bc.

The service request, at implementation should be deleted from Priority 5 Queue.

g) The Search-Transp Procedure, Search-Data Procedure and Search-Voice Procedure search for BC(s) to allocate to the IT(s) of Transp Requests, Data Requests and Voice Requests, respectively. In each procedure, a scan of the normal BC range shall begin at a randomized starting point which is not specified herein.

A.1.1.2.1.3 *Refreshment Message generation task*

In DCME frames when Priority 0 Queue is not to be addressed and there are no messages in queues 1 through 5, a Refreshment Message should be generated. A Refreshment Message should also be generated when Priority 3, 4 and 5 queue contains a request(s) which cannot be serviced due to unavailable bearer capacity unless a "disconnect" message is generated.

The refreshment should be done by scanning the range of normal BCs (from BC No. 1 to the upper pool boundary) and the range of overload BCs (from BC No. 64 to the highest number permitted by the pool size). Pre-assigned BCs should not be refreshed. Each dynamically assigned 64 kbit/s connection should be refreshed but only limited to the even-numbered BC (the next higher BC should not be refreshed). Every other Refreshment Message should alternate between the normal and the overload range. In each range, the BCs should be refreshed sequentially from the lowest to the highest BC, restarting the cycle after refreshment of the highest BC in the range. Whenever a BC is refreshed, all the required information elements should be inserted in the "ASSIGN" message. The refreshment of Bit Bank BC should be transmitted with IT No. 250.

A.1.1.2.1.4 *Map Update/Coder Selection*

The RAG stores information of two types:

a) Process parameters, consisting of both numbers and indexed arrays. This information is of static nature (derived from the configuration data);

b) the Resource Map – this information is dynamically variable, identifies the status of the BC, IT, IT type and ADPCM encoder connections.

At initialization (caused by the MCH), the Resource Map should be set to a known state (BCs, ITs and ADPCM encoders disconnected) and the process parameters should be loaded into the RAG Process. The RAG should then generate the "Seizesc" and the "Seizebank" messages in order to provide the SBC process with the information necessary for the allocation of pre-assigned channels and bit banks (associated with these channels). The pre-assigned channel allocation (determined by the configuration data) should be in accordance with the bearer structure requirements (§ 5.2).

Immediately after initialization, the RAG Process should also generate the "Set-Pre" message for the ENC Process. This message causes seizure of an ADPCM encoder for a pre-assigned connection and the setting of the encoding mode to 8, 5, 4 or optionally 3, or 2-bit.

The Map Update/Coder Selection Task performs the following actions as a result of channel type changes and implementation of service requests:

a) update the BC and IT connections, BC type and store the information in the Resource Map;

b) update the encoder connections and store the information in the Resource Map;

c) generate the output messages on signal paths 8, 13 and 16.

The Resource Map can be represented with indexed arrays as follows:

a) The Sat Array – This array, indexed by IT number, contains a BC number for each IT entry, i.e., Sat(IT) = BC number. This array provides the IT-to-BC association map. If the IT is associated to BC number 0 in the array, the IT is disconnected.

b) The IT Array – This array, indexed by BC number, contains an IT number for each BC entry, i.e., IT(BC) = IT number. This array provides the BC-to-IT association map. If the BC is associated to IT number 0 in the array, the BC is disconnected.

c) The Type Array – This array, indexed by BC number, contains a BC type identification for each entry, i.e., Type(BC) = BC type. The BC types are those listed earlier in this section.

d) The Cod Array – This array, indexed by IT number, contains the connected encoder number for each IT entry, i.e., Cod(IT) = encoder number. When the IT is connected to encoder number 0, the IT is disconnected.

The BC, IT connections and the BC types should be updated in accordance with the events occurring in the RAG Process in accordance with Tables A-4/G.763 and A-5/G.763, respectively. The bit bank deletion should be in accordance with the criterion specified in Table A-3/G.763. When the conditions for the deletion of a bit bank exist, the highest numbered bit bank BC should be deleted by changing its type to "Disconnected". Figure A-6/G.763 provides examples of connection and BC type update (the BC, IT connections are shown as points in the BC, IT cartesian coordinate plane).

Whenever a new assignment of a previously disconnected IT is made (other than IT No. 250), an ADPCM encoder should be selected from the available encoders of the ADPCM encoder pool and logged at the Cod Array for that IT entry. Whenever a re-assignment of a previously connected IT is made, the encoder currently associated with the IT should be maintained.

Whenever an IT is disconnected, the associated ADPCM encoder should be returned to the pool of available encoders.

Whenever an assignment request is serviced and the connection/BC type is updated, the "Assign" message (signals 8, 13 and 16) should be generated. The message format is (BC, IT, type, encoder number). Only a subset of the BC types can appear in the message namely: "Voice", "Data", "Transparent", "Bank", and "Disconnected". If the BC type is "Disconnected", the encoder number identifies the ADPCM encoder connected to the IT (and BC) prior to the disconnection.

Whenever an implicit disconnection takes place, the action to be taken should be in accordance with Table A-6/G.763.

If an optional USM is being used, the MAP update/coder selection task should not be performed in DCME frames 0, n, 2n, etc. (i.e., every nth DCME frame) of the DCME multiframe.

TABLE A-4/G.763

BC, IT connection update

	Event (RAG Process)	IT Array Entry for BC	BC Array Entry for IT	IT Array Entry for BC+1
Direct Action	Assign IT to BC (other than 64 kbit/s	IT	BC	No change
	Disc. IT from BC (other than 64 kbit/s)	0	0	No change
	Assign IT to BC (64 kbit/s)	IT	BC	0
	Disc. IT from BC (64 kbit/s)	0	0	0
Implied Disconnect	Assign a "Bank" BC	0	No change	No change
	Assign ITn to BC connected to IT	ITn	0	No change
	Re-assignment of IT from BC to BCn	0	BCn	No change

Note – BC-to-IT No. 0 connection means disconnection of BC. IT-to-BC No. 0 connection means disconnection of IT.

TABLE A-5/G.763

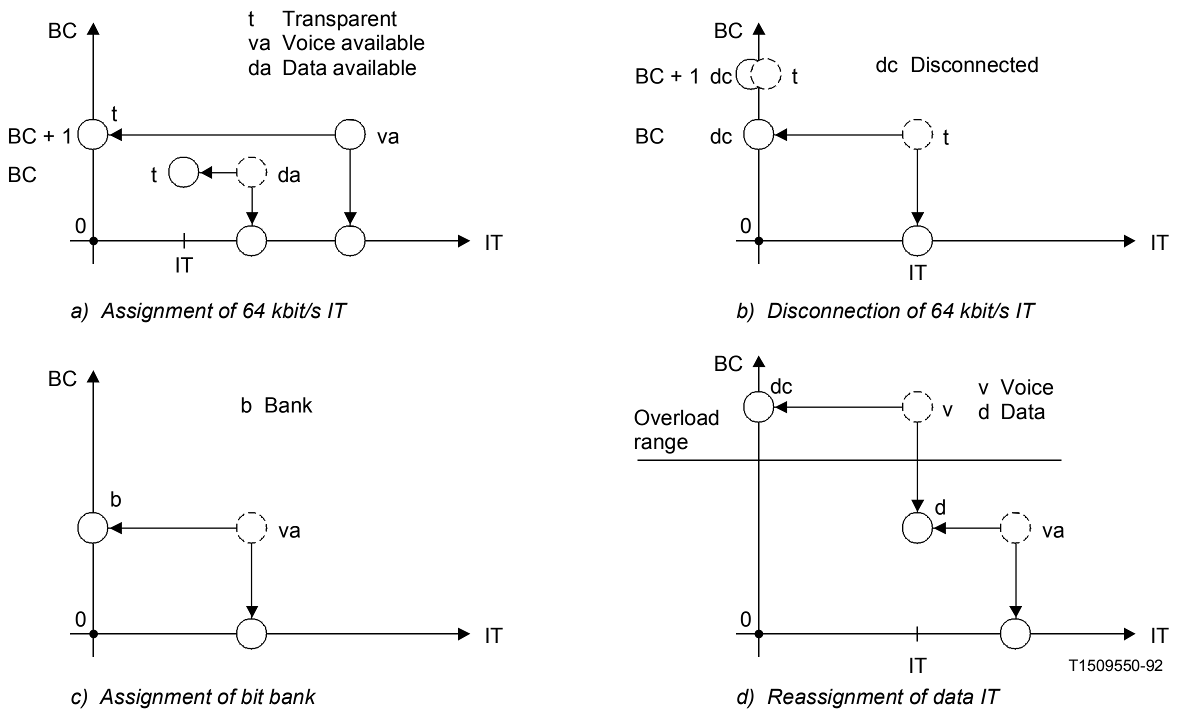
BC type update

Event (RAG Process)	BC Type	BC+1 Type
Message received: "Datainact (IT)" "Voiceinact (IT)" "Data (IT)" "Voice (IT)"	As determined by Input Pre Processing Task	No change
Assign Voice IT to BC	Voice	No change
Assign Data IT to BC	Data	No change
Assign Bit Bank to BC	Bank	No change
BC disconnection/bank deletion	Disconnected	No change
Assign 64 kbit/s IT to BC	Transparent	Transparent
Disc. 64 kbit/s IT from BC	Disconnected	Disconneted

TABLE A-6/G.763

Actions caused by implicit disconnections

Implicit Disconnect	Action
"Data" BC	Generate "Reinsert (BC)" message
Overload BC	Generate "Remove (BC)" message
"Bank" BC	Generate "Releasesc (BC)" message
IT	Generate "Release (ADPCM encoder number)" message



Note – Each circle represent a connected BC, IT pair and it shows the BC type. Circles on the vertical and horizontal axes are disconnected BCs and ITs, respectively. Dashed line circles are previous connections which, at assignment, are replaced by new connections (solid line circles).

FIGURE A-6/G.763

Examples of connection and BC type update

A.1.1.2.1.5 *Search-Transp Procedure*

The Search-Transp Procedure searches for capacity for the allocation of the 64 kbit/s IT. The search should be limited to the normal BC range.

The procedure should generate, as an output, 11 values for the 11 variables defined in Table A-2/G.763 and illustrated in Figure A-7/G.763. It should also select an ADPCM encoder number from the pool of available encoders. However, if IT is connected, the currently used ADPCM encoder should be kept for the new connection.

The procedure should select the bc, bc+1 pair using the priority scheme specified in Table A-7/G.763 in the normal BC range. The search should proceed from Priority 1 to Priority 10. There is a possibility that none of the 10 choices will be available. In this case, if the requesting IT is connected to a BC, the IT shall be disconnected. If the requesting IT is not connected, a refreshment message should be generated.

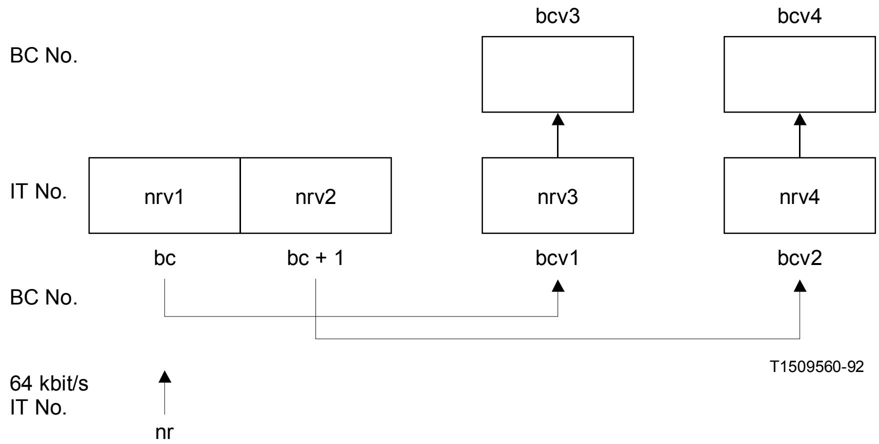


FIGURE A-7/G.763

Variables of the search-transp procedure

TABLE 7/G.763

Priority scheme for search-transp procedure

Priority	BC Types
1	disc/disc
2	avail/disc disc/avail
3	avail/avail
4	voice/disc disc/voice
5	voice/avail avail/voice
6	voice/voice
7	data/disc disc/data
8	avail/data data/avail
9	data/voice voice/data
10	data/data
priority	voice/avail

Disc BC disconnected.
BC types types of candidate pair BCs (bc and bc+1).
Avail BC available/either "Voice-avail" or "Data-avail".

If bc (bc+1) contains the *data* IT nrv1 (nrv2), the bearer channel bcv1 (bcv2) should be selected (in the normal BC range) for the re-assignment of nrv1 (nrv2) using the following priorities:

Priority 1 – "Data-avail" BC;

Priority 2 – "Disconnected" BC;

Priority 3 – "Voice-avail" BC;

Priority 4 – "Voice" BC.

If the IT nrv3 (nrv4), occupying the selected BC, is of the voice type, the overload BC bcv3 (bcv4) should be selected for re-assignment of nrv3 (nrv4).

If bc (bc+1) contains the *voice* IT nrv1 (nrv2), the bearer channel bcv1 (bcv2) should be selected for the re-assignment of nrv1 (nrv2), using the following priorities:

Priority 1 – "Disconnected" normal BC;

Priority 2 – "Voice-avail" or "Data-avail" normal BC;

Priority 3 – "Disconnected" overload BC.

A.1.1.2.1.6 *Search-Data Procedure*

The Search-Data Procedure searches for a BC for the allocation of a data IT. The search should be limited to the normal BC range. The procedure should select a BC using the following priority scheme:

Priority 1 – "Data-avail" BC;

Priority 2 – "Disconnected" BC;

Priority 3 – "Voice-avail" BC;

Priority 4 – "Voice" BC.

One of the four choices will be available.

The procedures should select an encoder number (if IT is connected, use the current encoder for selection) and should generate as an output, three values for the variables defined below:

bc: BC no. for allocation of the data IT;

nrv: No. of the IT currently occupying bc;

bcv: BC no. for re-allocation of nrv.

Note that nrv = 0 indicates a disconnected BC and bcv = 0 indicates that a re-assignment is not required.

A.1.1.2.1.7 *Search-Voice Procedure*

The Search-Voice Procedure searches for a BC for the allocation of the voice IT. The search should first scan the normal BC range and should attempt to select one of these two types of BC based on the specified priority:

Priority 1 – "Disconnected" BC;

Priority 2 – "Voice-avail" or "Data-avail" BC.

If this search is unsuccessful, the overload BC range should be scanned from the lowest BC to the highest permissible BC until a disconnected BC is found.

The procedure should select an ADPCM encoder number and should generate, as an output, two values for the two variables defined below:

bc: BC no. for allocation of the voice IT;

nrv: No. of the IT currently occupying bc.

Note that nrv = 0 indicates a disconnected BC.

A.1.1.2.2 *BC Bit Map creation process*

The SBC Process input/output connection is shown in Figure A-4/G.763. The SBC Process receives the messages in signal path 13 and generates the BC Bit Map (signal path 14) and the Mode Map (signal path 15).

One function of the SBC Process is to establish the association between the bits of each ADPCM encoder output and the bits of the bearer frame (signal path 14: BC Bit Map). The SBC also determines the 4/3/optionally 2 bit mode of the ADPCM encoders used for voice (signal path 15: Mode Map).

Inherent to the above two functions is the bit bank handling and the overload channel creation. The bit bank handling consists of deriving the LSB of each channel from one of the existing bit banks according to predetermined rules.

When the overload mode is required, the use of 3 bit per sample encoding is distributed across the entire set of speech channels. The objective is to make the average encoding rate almost equal for the normal voice BCs (subject to bit stealing) and the overload channels. This is obtained by stealing bits from eligible BCs in a pseudo-random fashion and by making each overload BC alternate between 4 and 3 bit encoding (also in a pseudo-random fashion).

When the 4/3 bit overload channel creation procedure, described above, cannot provide the required number of overload channels, the optional 3/2 bit overload channel creation procedure may be used. This procedure, similarly to the 4/3 bit procedure, makes the average ADPCM encoding rate almost equal for the normal voice BCs (subject to bit stealing) and the overload channels. Pseudo-random bit rotation and switching between two alternate overload channel sizes (3-bit and 2-bit channels) are used also in this case.

The SBC Process maintains ten lists. All lists contain, in ascending order, the BC numbers that fall into the categories defined below. BCs are extracted from, or inserted into, these lists always keeping the BC numbers in ascending order. A BC can appear only in one list at the same instant.

Voice list – This list contains all BC numbers of type “Voice”, “Voiceavail” or “Disc” that are within the normal BC range. Reception of messages on signal path 13 may change the contents of the list. At initialization, this list contains all normal BC numbers subject to DSI.

Overload BC List – This list contains all BC numbers of type “Voice” that are in the overload BC range. Reception of messages on signal path 13 may change the contents of this list. At initialization, this list contains no entries.

Pre-assigned 40 List – This list contains all BC numbers that are pre-assigned as 40 kbit/s channels. At initialization, this list contains no entries. The RAG process will send information (“Seizesc” message) directly after initialization which brings the contents of this list to its final state. After this has been done, the contents will not change.

Pre-assigned 32 List – This list contains all BC numbers that are pre-assigned as 32 kbit/s channels. It is treated in an analogous manner to the pre-assigned 40 list.

Pre-assigned 24 List – This list contains all BC numbers that are pre-assigned as 24 kbit/s channels. It is treated in an analogous manner to the pre-assigned 40 list.

Pre-assigned 16 List – This list contains all BC numbers that are pre-assigned as 16 kbit/s channels. It is treated in an analogous manner to the pre-assigned 40 list.

Pre-assigned 64 list – This list contains the even BC numbers that are pre-assigned as 64 kbit/s channels. It is treated in an analogous manner to the pre-assigned 40 list.

Data list – This list contains all BC numbers of type “Data” or “Data-avail”. Reception of messages on signal path 13 may change the contents of this list. At initialization this list contains no entries.

Bit Bank List – This list contains all BC numbers of type "Bank". Reception of messages on signal path 13 may change the contents of this list. At initialization, this list contains no entries. The RAG Process will send information ("Seizebank" message) directly after initialization, which inserts the bank BCs for pre-assigned channels into the list.

Transplist – This list contains the even BC number of type "Transp". Reception of messages on signal path 13 may change the contents of this list. At initialization, this list contains no entries.

The possible BC transitions between non-pre-assigned lists are illustrated in Figure A-8/G.763. Note that for "Transparent" BCs, only BC bc (the lower nibble) is either put into the "Transplist" or extracted from it. BC bc+1 (higher nibble) is extracted from the voice list or data list at connection of the 64 kbit/s call and inserted back in the voicelist at disconnection. It is noted that a BC number appears only in one list.

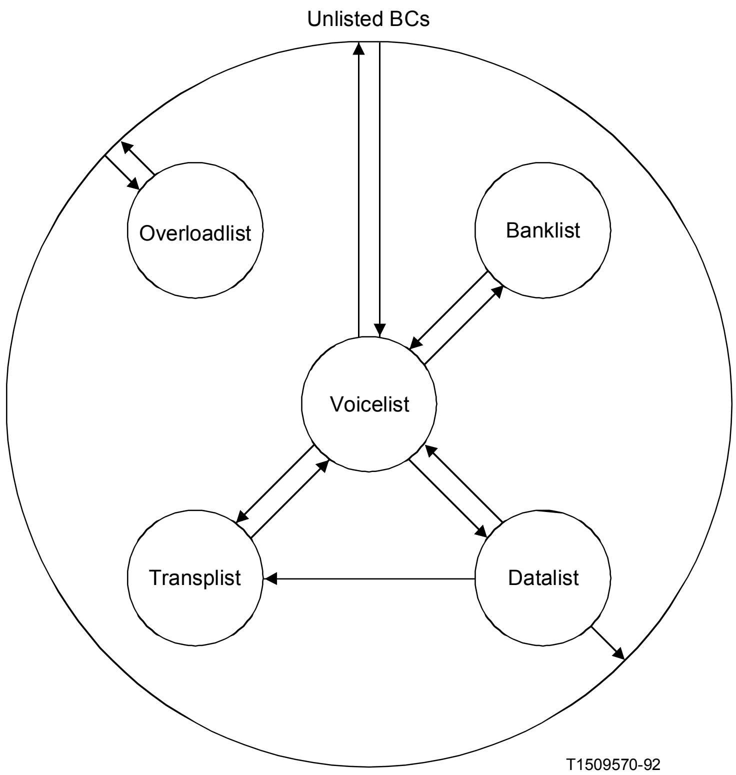


FIGURE A-8/G.763

Possible BC transitions in the SBC process

Figure A-9/G.763 shows the BC transitions corresponding to the example cases a), b), c) and d) shown in Figure A-6/G.763.

The SBC Process also maintains the Coder Array. In this array, each index corresponds to a possible BC number. The indexed item is the encoder number used by that BC. At initialization, it contains all zeroes.

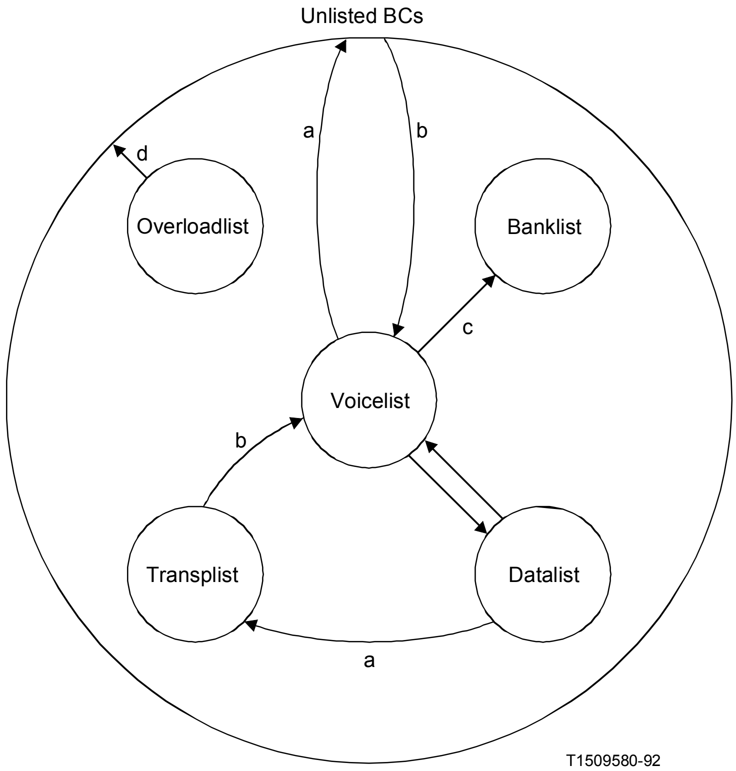


FIGURE A-9/G.763
BC transitions in cases a, b, c and d of Figure A-6/G.763

A.1.1.2.2.1 *Bit bank handling*

A bit bank BC number should be placed into the bank list at the reception of an "Assign" message containing IT No. 250, if the associated BC number does not already exist in the bit bank list. The BC number in question should be removed from the voice list when this occurs.

A bit bank BC number is removed from the bank list at the reception of a "Releasesc" message for that BC. The BC number should then be put back into the voice list.

At the occurrence of the trigger pulse, the bit position of the LSBs of the handled data calls (pre-assigned 40 or DSI channels declared as data) should be generated in the following way:

a) LSB of BC number in pre-assigned 40 list position 1:

MSB of BC number in bank list position 1;

b) LSBs of BC numbers in pre-assigned 40 list positions 2 to 4:

2nd, 3rd and 4th bits, respectively, at BC number in bank list position 1;

c) LSB of BC number in pre-assigned 40 list position 5:

MSB of BC number in bank list position 2.

This procedure is followed until the BC numbers in the pre-assigned 40 list have been handled, after which the BC numbers in the first position in the data list follows.

A.1.1.2.2.2 *Overload channel creation*

When any of the signal path 13 message "Assign", "Reinsert", or "Remove" is received the voice list or overload list are updated and the associated list lengths Nv (Voice list) and Nov (Overload list) computed. If Nov is 0, overload channel creation is not required.

If Nov is greater than 0, but not greater than Nv/3, the number (N4) of channels in the overload range that will be carried at four bits per sample shall be computed as follows:

$$N4 = \text{Integer}\left[\frac{Nv \times 4 \times Nov}{Nv + Nov} + \frac{1}{2}\right] - Nov \times 3$$

In addition, when Nov is greater than 0, the integer variables Pv and Pov shall be computed as follows:

Pv = (IT modulo Nv)

Pov = (IT modulo Nov)

where IT is the IT number contained in the "Assign" message (see Note). Pv and Pov represent voice and overload list positions. These positions are numbered from 0 to Nv – 1 and from 0 to Nov – 1, respectively.

Note – If an optional USM is being used, the IT number information in DCME frames 0, n, 2n, etc. (i.e., every nth DCME frame) of the DCME multiframe should also be used to create overload channels.

At the occurrence of the trigger pulse, the BCs stored in the overload list at the first N4 locations (modulo Nov) starting from the list position Pov should be marked as 4-bit channels. The remaining BCs of the overload list should be marked as 3–bit channels.

The 4/3 bit mode of the first BC in the overload list should be checked to determine whether it is 4 or 3-bit. If the mode is 3-bit, the BCs stored in the voice list at the first three locations, starting from the list position, Pv, should be set to 3-bit mode. The following bit associations should be entered into the BC Bit Map:

a) MSB of BC in overloadlist position 0

   LSB of BC in voicelist position Pv;

b) 2nd bit of BC in overloadlist position 0

   LSB of BC in voicelist position (Pv+1 modulo Nv);

c) LSB of BC in overloadlist position 0

   LSB of BC in voicelist position (Pv+2 modulo Nv).

   If the first BC in the overloadlist is a 4-bit channel, items a) and b) above remain applicable, c) changes and d) is added as shown below:

c) 3rd bit of BC in overload list position 0

   LSB of BC in voice list position (Pv+2 modulo Nv);

d) LSB of BC in overload list position 0

   LSB of BC in voice list position (Pv+3 modulo Nv).

The same procedure should be applied to the second BC in the overload list, starting at either position Pv+3 or Pv+4, depending on the 4/3-bit mode of the first BC.

The same procedure should be applied to the remaining BCs of the overload list. The voice list BCs subject to bit stealing will be marked as 3-bit channels, the remaining ones as 4-bit channels. An example of the procedure is illustrated in Figure A-10/G.763.

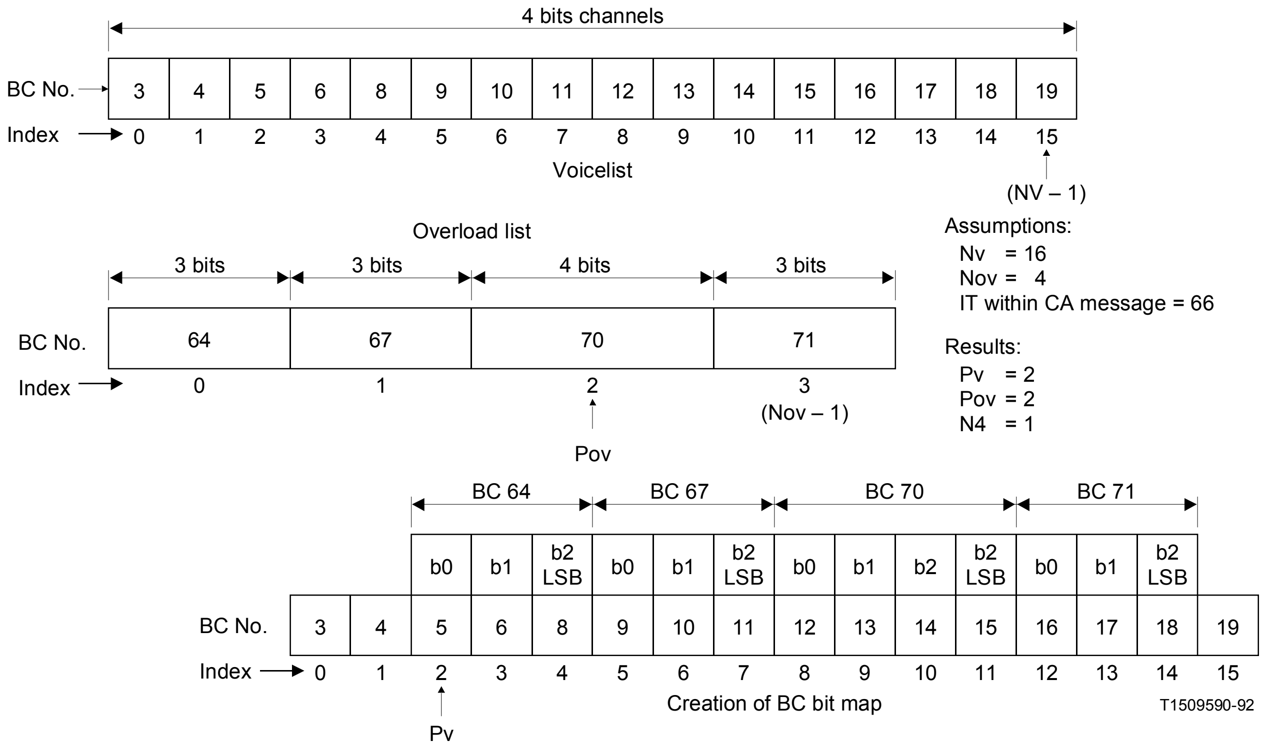


FIGURE A-10/G.763

Overload channel creation process (example)

If Nov is greater than Nv/3 and the optional 2-bit overload feature is available and enabled, the number (N3) of channels in the overload range that will be carried at 3 bits per sample shall be computed as follows:

$$\text{N3} = \text{Integer}\left[\frac{\text{Nv} \times 4 \times \text{Nov}}{\text{Nv} + \text{Nov}} + \frac{1}{2}\right] - \text{Nov} \times 2$$

The number (n2) of bearer channels in the voicelist that will operate at 2-bits per sample (i.e., these channels “donate” 2 bits) shall be computed as follows:

$\text{n2} = \text{N3} - \text{Nv} + \text{Nov} \times 2$.

In addition, the integer variables Pv and Pov shall be computed as follows:

Pv = (IT modulo Nv)

Pov = (IT modulo Nov)

where IT is the IT number contained in the assignment message (see Note 1). Pv and Pov represent voice and overload list positions. These positions are numbered 0 to Nv–1 and from 0 to Nov–1, respectively.

The BCs stored in the overload list at the first N3 locations (modulo Nov), starting from the list position Pov, shall be marked as 3-bit channels. The remaining BCs of the overload list shall be marked as 2-bit channels.

The BC stored in the voice list at the first n2 locations (modulo Nv), starting from the list position Pv, shall be marked as 2-bit channels. The remaining BCs of the voice list shall be marked as 3-bit channels (i.e., these channels “donate” one bit).

In order to obtain the bit associations for the BC Bit Map, the "donated" bits from the channels of the voice list shall be arranged in the following ordered sequence:

Place in the sequence	*Bit* (see Note 2)
1st	2nd bitof BC at position Pv
2nd	LSB of BC at position Pv
3rd	2nd bit of BC at position Pv+1
4th	LSB of BC at position Pv+1
5th	2nd bit of BC at position Pv+2
6th	LSB of BC at position Pv+2, etc.

The bits of the overload channels shall be arranged in the following ordered sequence:

Place in the sequence	*Bit (see Note 2)*
1st	MSB of BC at position 0
2nd	2nd bit of BC at position 0
3rd	LSB of BC at position 0
4th	MSB of BC at position 1
5th	2nd bit of BC at position 1
6th	LSB of BC at position 1, etc.

Note 1 – If an optional USM is being used, the IT number information in DCME frames 0, n, 2n, etc. (i.e., every nth frame) shall also be used to create overload channels.

Note 2 – One or more bits indicated below may not be available, in which case the sequence is compacted upwards.

The first bit, second bit, third bit, etc., of the voice list bit sequence shall be associated with the corresponding bits of the overload bit sequence. This is illustrated in Figure 12/G.763. A particular example, for a pool of nine BCs, from BC No. 1 to BC No. 9, is shown in Figure 13/G.763.

A.1.1.2.2.3 *Mode Map and BC Bit Map Update*

As a result of the overload channel creation, BCs in the voice and overload lists are marked as either 4, 3 or optionally 2-bit channels. This information is stored in the Mode Map, which is updated (or refreshed) once per DCME frame.

The update (or refresh) of the "Mode Map" message implies the generation of a message intended for the ENC Process. This message should address all ADPCM encoders connected to the BC numbers that are in existence within the voice list and the overload list and give their associated mode (4, 3 or optionally 2). The BCs that are disconnected will not have an ADPCM encoder number associated with their BC numbers in the Cod Array and should not be addressed within the "Mode Map" message.

The information contained in the SBC lists and array and the results of the Bit Bank Handling and Overload Creation Procedures permit the generation of the "BC Bit Map". This map contains the bit association of all used BCs (ADPCM encoder outputs) with all used bearer channels. This map is updated (or refreshed) once per DCME frame.

An update or refresh of the "BC Bit Map" message implies the generation of a message intended for the BMI Process. This message should contain the bit association of all used BCs (ADPCM encoder outputs) with all used bearer channels.

A.1.1.2.3 *Bit Map Implementation Process*

The BMI Process input/output connection is shown in Figure A-4/G.763. This process receives the "BC Bit Map" from the SBC Process (signal path 14) and delivers it after a delay to the BC Bit Assignment Unit on signal path 9. This output is referred to as "Addressmap for BCs". The delay is such that the BC Bit Map is implemented at the beginning of the DCME frame which occurs 3 frames after the start of the DCME frame containing the applicable assignment message. Refer to Figure A-11/G.763.

The BC Bit Assignment Unit uses the BC Bit Map to route the ADPCM encoder unit output (BCs) to the appropriate bits in the bearer frame.

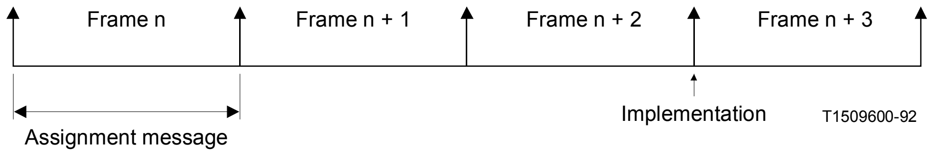


FIGURE A-11/G.763

Implementation timing

A.1.1.2.4 *Encoder Unit Control Process*

The ENC Process input/output connection is shown in Figure A-4/G.763. At initialization, the ENC Process received the "Set-pre" message (signal path 16), which allocates ADPCM encoders to pre-assigned channels and sets their modes to 8, 5, 4 or optionally 3, or 2-bit. This process also receives the "Mode Map" message from the SBC Process (signal path 15) and the "Assign-enc" and "Release-enc" messages from the RAG Process (signal path 16). It generates the "Setcod" message on signal path 7 for the Encoder Unit.

The ENC Process should be considered associated with a single ADPCM encoder so that conceptually, there are as many processes as there are ADPCM encoders. In practical implementation, the process can be time-shared among ADPCM encoders. The ENC Process sets the operating parameters of the ADPCM encoder to which it is dedicated, based on the messages received. The ADPCM encoder operating parameters indicate the BC and IT connection, the 8/5/4/3/optionally 2-bit mode, and whether the ADPCM encoder needs resetting.

Resetting will be required when the IT connection to an ADPCM encoder is changed (the encoder must be reset before establishing a new connection).

When the "Assign-enc" message is received, the ENC Process should determine whether the ADPCM encoder number in the message is the same as the ADPCM encoder number to which the process is dedicated (cod). If the number is different, no action should be taken.

If the ADPCM encoder number is the same as cod, the ADPCM encoder connection should be set in accordance with the received BC type and IT numbers. If the BC type is "Disconnected", the encoder should be disconnected.

Reception of the "Release-enc" message should result in the same action as the reception of the "Assign-enc" message, indicating a "Disconnected" BC type.

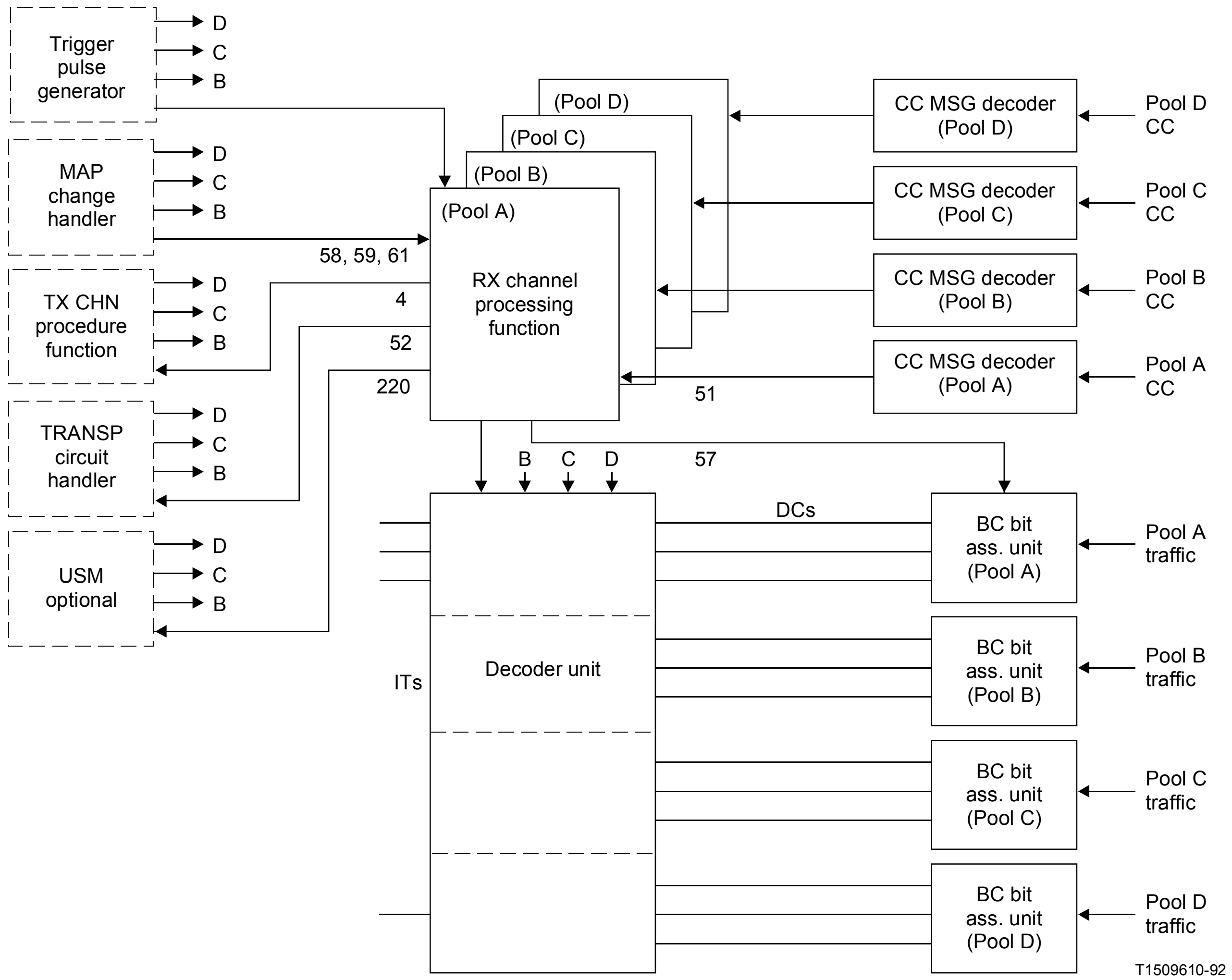
The ADPCM encoder bit mode should be set to 8, 5, 4, optionally 3, or 2-bit in accordance with the contents of the "Set-pre" message (8, 5, 4 or optionally 3 or 2-bit pre-assigned channels), the "Assign-enc" message (5-bit for data, 8-bit for transparent channels) or the "Mode Map" (voice channels).

The “Setcod” message containing the encoder operating parameters is sent to the Encoder Unit. Each “Setcod” message is destination directed to an encoder (cod). The message “Setcod” (cod, IT, mode, reset) indicates the connection for the ADPCM encoder as well as the 8/5/4/3/optionally 2-bit mode of operation and whether the ADPCM encoder needs resetting. The message “Setcod” (cod, 0, ...) indicates that the ADPCM encoder must be disconnected.

The “Setcod” message for pre-assigned channels is sent immediately after initialization. The “Setcod” message for DSI channels should be sent after the ADPCM encoder parameters are set, such that the ADPCM encoder mode/connection is switched at the beginning of the DCME frame which occurs 3 frames after the start of the DCME frame containing the applicable assignment message. Refer to Figure A-11/G.763.

A.2 *An example of a DCME receive unit structure*

An example of a DCME receive unit structure is shown in Figure A-12/G.763. Compliance with this receive unit structure will permit the DCME receive unit function to be tested with INTELSAT IESS-501(Rev.2) compliant DCME test equipment and software protocol references. This structure is based on a non-mandatory partitioning of functions and definition of signals.



Note – Signal paths are defined in Table A-8/G.763.

FIGURE A-12/G.763

DCME receive unit structure

TABLE A-8/G.763

Legend for Receive Unit Signal Paths

Signal path No.	Signal type/message	Definition
4	"Rxdata"	§ A.2.1.1
51	"Assign"	§ A.2.1.1
52	"Rxtranspreq", "Rxtransprel"	§ A.2.1.1
53	"BC Bit Map"	§ A.2.1.1
54	"Seize", "Seizev", "Release", "Mode Map"	§ A.2.1.1
55 (and 56)	"Trigger Pulse" from external unit	§ A.2.1
56	"Setcod"	§ A.2.1.3
57	"Addressmap for BCs"	§ A.2.1.2
58 (and 59, 61)	"Process-reset" from MCM	§ A.2.1
220	"Change" to MSU	§ A.2.1.1

Some of the functional blocks in Figure A-12/G.763 are internal to the DCME receive unit structure, while others are external but provide or receive required interface signals. The represented structure shows a multidestination (MD) DCME, corresponding with four origins. However, since the internal blocks in the figure are defined on a single pool basis, the structure can also represent the case of a point-to-point configuration receiving 1 pool. The blocks internal to the structure need to be duplicated or shared between pools. The blocks that belong to the receive unit structure are:

a) *The Control Channel Message Decoder* – This unit receives the control message associated with the received pool and decodes it from the format specified in § 11. This constitutes the input for the Receive Channel Processing Function. The control message decoder also distributes control message contents not pertaining to the Receive Channel Processing Function:

- the encoded background noise level within the Synchronous Data Word is provided to a separate unit for decoding and use in accordance with § 11.3.3.1;

- the Asynchronous Data Word is provided to a separate unit for decoding and use in accordance with § 11.3.3.2;

- a channel check type indication within the Synchronous Data Word is provided to a separate unit for use in accordance with §§ 11.3.3.1 and 10.

b) *The Receive Channel Processing (RCP) Function* – This function consists of an ensemble of interconnected processes. It receives an input from the Assignment Message Decoder, provides outputs to blocks internal to the Receive Unit Structure (Decoder Unit and BC Bit Assignment Unit) and provides outputs to blocks which are external to the Receive Unit Structure.

c) *The BC Bit Assignment Unit* – This unit is connected to the input of the Decoder Unit (BCs). The BC Bit Assignment Unit derives the bits required for each ADPCM decoder input from the correct bits of the received bearer channel. The bit map for this association is provided by the RCP function.

d) *The Decoder Unit* – This unit consists of a bank of ADPCM decoders which can be connected to any allocated IT and to any BC of the pool. Each BC can carry 8, 5, 4, 3, or optionally 2-bit samples or can be disconnected from the ADPCM decoders. A sufficient number of ADPCM decoders must be provided to ensure that freeze-out due to unavailability of ADPCM decoders cannot occur.

The ADPCM decoders can be set to an 8, 5, 4, 3, or optionally 2-bit mode of operation and can be initialized to a known state. The IT and BC connection/disconnection information for each ADPCM decoder, as well as the mode of operation selection and the initialization signal are provided by the RCP function.

The blocks which are external to the Receive Unit Structure but which have signal paths with the RCP are:

a) *Transmit Channel Processing Function* – Information on the data connection of received ITs is passed to the TCP function by the RCP.

b) *Transparent Circuit Handler* – This process which is described in § 8 is informed by the RCP that a 64 kbit/s assignment or disconnection has been performed for an IT.

c) *Map Change Handler* – The Map Change Handler (MCH) is a process which controls the configuration data for the DCME. At start-up, this process issues signals making it possible to configure the system correctly. The same is done at a map change instant. Refer to §§ 15.1 and 15.6.

d) *Trigger Pulse Generator* – This unit provides a periodic 2 ms timing reference signal to the processes of the receive unit structure.

e) *User Signal Modual (optional)* – This USM receives signalling state change signals. The specification of the USM is at the users' option.

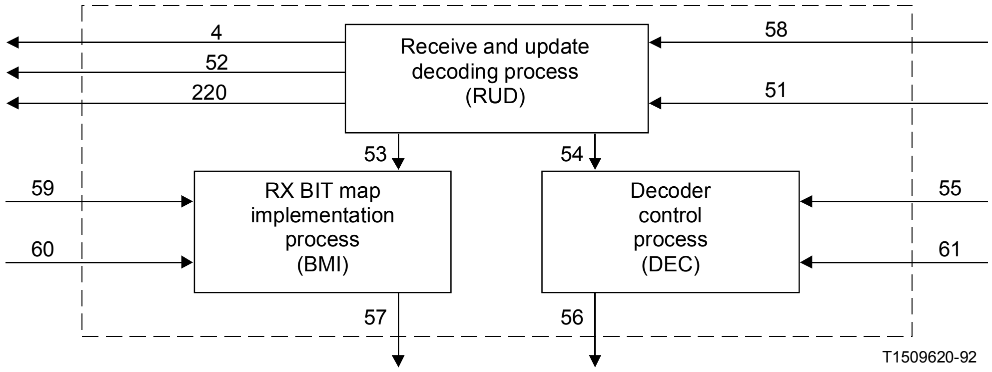
A.2.1 *Receive Channel Processing Function*

The Receive Channel Processing Function interfaces with other elements of the receive unit structure as shown in Figure A-12/G.763. The RCP function processes the output of the Assignment Channel Decoder and takes consequent actions by providing required information to the Decoder Unit, the BC Bit Assignment Unit, the Transparent Circuit Handler and the Transmit Channel Processing Function. The RCP function receives a reset signal from the Map Change Handler which terminates the processes at the map change instant.

The internal structure of the RCP as shown in Figure A-13/G.763 is comprised of the Received Channel Status Update and Overload Decoding Process (RUD), the Bit Map Implementation Process (BMI) and the Decoder Control Process (DEC).

A.2.1.1 *The Receive Channel Status Update and Overload Decoding Process (RUD)*

The RUD Process is dedicated to one received pool. There will be (conceptually) as many processes as there are received pools. The RUD process analyses the control channel message and generates the required actions based on the contents of this message.



Note – Signal paths are defined in Table A-8/G.763.

FIGURE A-13/G.763

RCP function

The RUD input/output connections are shown in Figure A-13/G.763. The RUD receives an input (signal path 51) from the Assignment Channel Decoder and an input (signal path 58) from the Map Change Handler. The contents of these signal paths are defined below:

Signal Path 51: This signal path carries the "Assign" message which contains assignment information obtained from the Assignment message decoder. The message format is (BC, IT, Call). The last variable defines the decoded BC type. The "Call" variable can define three BC types, "Voice", "Data" and "Transparent". Two additional BC types, "Disconnected" and "Bank" are defined by the reception of the IT No. 0 and No. 250, respectively.

Signal Path 58: This signal path carries the "Process-reset" message. This message is issued by the MCH in association with a map change. The reception of this message causes the termination of the RUD Process.

The RUD Process generates outputs for the TCP Process (signal path 4), the DEC process (signal path 54), BMI Process (signal path 53) and the Transparent Circuit Handler (signal path 52). When required the RUD Process also generates a signal to the optional USM. This signal (signal path 220) contains the "change(IT)" message. The outputs are defined below:

Signal Path 4: This signal path carries the following message "Rxdata(IT)". This message is sent to the transmit unit assignment procedures when the transition occurs from a previous BC type to a data BC (IT is the transmit IT number).

Signal Path 52: This signal path carries the following messages (IT is the transmit IT number):

– *"Rxtranspreq(IT)"* – This message is given to the Transparent Circuit Handler when the transition occurs from another BC type to a transparent BC type.

– *"Rxtransprel(IT)"* – This message is the reverse of the above. It is sent when a transition occurs from a transparent BC to something else (disconnection).

Signal Path 53: This signal path carries the message "BC Bit Map". This message defines which bearer channel bits should be given to the different ADPCM decoders.

Signal Path 54: This signal path carries the following messages:

– *"Seize (IT, Mode)"* – This message contains the local IT number and the mode in which the ADPCM decoder should be set.

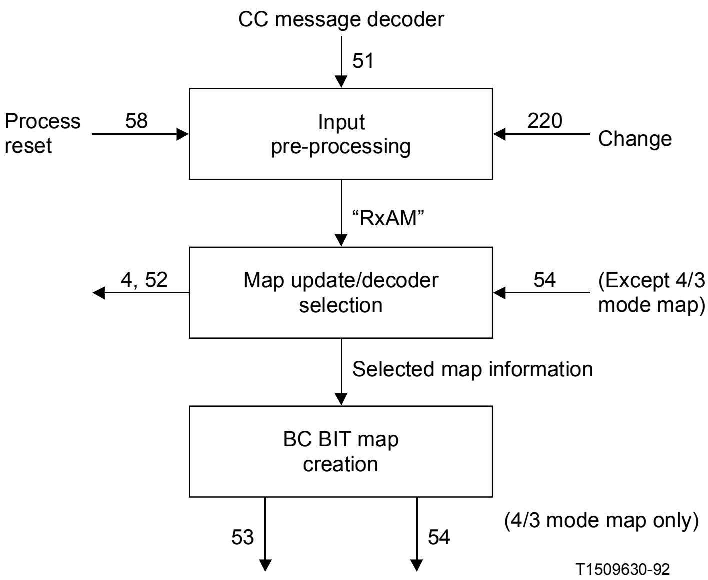
  This message is sent to the DEC Process immediately after initialization, to establish the ADPCM decoder connections for pre-assigned 64 kbit/s (transparent), 40 kbit/s (data), 32 kbit/s, 24 kbit/s and 16 kbit/s (option) calls. The decoder mode will be 8, 5, 4, 3 or optionally 2-bit, respectively. The "Seize" message is also sent, during the DCME operation, to establish decoder connections for dynamically assigned data and transparent calls. The ADPCM decoder mode will be 5- and 8-bit, respectively.

– *"Seizev (IT)"* – This message is delivered in order to associate a dynamically assigned voice channel with an ADPCM decoder. The same parameter as in the signal above is given, with the exception of the mode.

– *"Release"* – This message is used to release a designated ADPCM decoder back into the decoder pool.

– *"Mode Map"* – This message contains the modes that are to be used for the different ADPCM decoders that are connected to voice channels.

The RUD Process can be functionally divided into three tasks, namely the Input Pre-Processing Task, the Map Update/Decoder Selection Task and the BC Bit Map Creation Task (see Figure A-14/G.763).



Note – Signal paths are defined in Table A-8/G.763.

FIGURE A-14/G.763

RUD process tasks

The Input Pre-Processing Task performs a validity check on the “Assign” message and derives the implicit BC types (determined by the BC number).

The Map Update/Decoder Selection Task analyses the pre-processed “Assign” message, updates the internal maps of the RUD Process and generates messages on signal paths 4, 52 and 54 (except the “Mode Map” message).

The BC Bit Map Creation Task performs the bit bank handling and overload channel derivation functions and generates the BC Bit Map message on signal path 53 and the “Mode Map” message on signal path 54.

A.2.1.1.1 *Input Pre-Processing Task*

Upon reception of the “Assign” message, a validity check should be performed to ensure that the message is consistent with the transmit unit assignment rules and with the DCME configuration data. A minimum list of conditions to be verified should be as specified below:

a) if the BC is in the overload range, or if the IT number is 250, the MSB of the BC Identification Word in the Assignment message must be 0 (“Voice”);

b) if the BC type is transparent, the MSB of the BC Identification Word must be 0 (“Voice”) and the BC number must be even;

c) the BC number must be contained in the range allocated to the received pool (including overload channels) and not already used for a pre-assigned channel;

d) the IT number must be contained in the range which the corresponding DCME (transmit unit) can address for all destinations;

e) the BC number must be in the normal range if the BC type is data, transparent or if the IT number is 250;

f) if the optional USM is used, “RxAM” messages of the form (BC number 255, ITn) will be delivered in DCME frames 0, n, 2n, etc. (i.e. every nth DCME frame) of the DCME multiframe.

If any of the above conditions are not satisfied, or if the DCME frame alignment is lost, no further processing of the assignment message shall be performed. The received IT number shall be assumed to be 0 for the purpose of providing a pointer value for the overload channel derivation (§ A.2.1.1.3).

If the validity check is successful, the received assignment message should be processed as follows:

a) if the IT number is 0, the BC type should be set to “Disconnected”;

b) if the IT number is 250, the IT number should be changed to 0 and the BC type shall be set to “Bank”.

The processed assignment message, referred to as “RxAM(BC, IT, Rxtype)” should then be passed to the Map Update/Coder Selection Task for further processing.

A.2.1.1.2 *Map Update/Decoder Selection Task*

The RUD stores information of two types:

a) Process parameters, consisting of both numbers and indexed arrays – This information is of a static nature (derived from the configuration data).

b) The Resource Map – This information which is dynamically variable, identifies the status of the BC/IT connection, BC type and ADPCM decoder connections.

At initialization (caused by the MCH), the Resource Map should be set to a known state (BCs, ITs and ADPCM decoders disconnected) and the process parameters should be loaded into the RUD Process. This includes the information necessary for the allocation of pre-assigned channels and bit banks (associated with these channels). The pre-assigned channel allocation (determined by the configuration data) should be in accordance with the bearer structure requirements (§ 5.8). A map which identifies the remote IT numbers intended for the DCME and associates them with the local IT numbers (making up the circuit), is included in the information loaded at initialization. The local IT numbers are the numbers used by the DCME in its transmitted assignment message. The remote IT numbers are those used in the received assignment message(s).

Immediately after initialization, the RUD Process should generate "Seize" messages to the DEC Process. This will cause seizure of ADPCM decoders for pre-assigned connections and the setting of the ADPCM decoding mode to 8, 5, 4, or optionally 3 or 2-bit.

The Map Update/Decoder Selection Task performs the following actions as a result of the processing of the received assignment message ("RxAM"):

a) update and store the BC/IT connections and BC types in the Resource Map;

b) select the decoder connections and store the information in the Resource Map;

c) generate the messages for signal paths 4, 52 and 54 (except the "Mode Map" message).

The Resource Map can be represented with the four indexed arrays Sat, IT, Type and Dec. The first three are identical to the arrays with the same name, defined in the transmit unit structure (§ A.1.1.2.1.4). The BC types which can be stored in the Type Array are "Transp", "Data", "Voice", "Disc" and "Bank".

The Dec Array, indexed by IT numbers, contains the connected ADPCM decoder number for each IT entry, i.e. Dec(IT) = ADPCM decoder number. When the IT is connected to ADPCM decoder number 0, the IT is disconnected. The IT numbers used are the local IT numbers.

At the reception of the "RxAM" message, the IT-to-BC connection should be logged in Sat Array, the BC-to-IT connection should be logged in the IT Array and Rxtype should be logged in the Type Array for the BC entry (the previously stored BC, IT connection and the BC type will be updated). Additional changes to the information stored in the IT, Sat and Type Arrays should be made as specified below.

a) If the Receive Type is "Transparent", BC + 1 should be disconnected in the IT array if it is connected before (i.e. BC + 1 connected to IT No. 0) and the BC Type Array entry for BC + 1 should be logged as "Transp";

b) if the connection of a BC changes to a different IT, the previously connected IT, defined as ITp, should be disconnected in the Sat Array (i.e. ITp connected to BC No. 0). This is an implicit IT disconnect;

c) if the connection of an IT changes to a different BC, the previously connected BC should be disconnected in the IT Array and its type should be changed to "Disc";

d) if a BC of the transparent type changes to a different type, the other BC of the transparent BC pair should be disconnected in the IT and Type Arrays. Its associated IT should be disconnected in the Sat Array.

If, as a result of the above actions, the conditions exist for the deletion of a bit bank (as per Table A-3/G.763), the BC type "Bank" should be changed to "Disc".

If the optional USM is used and the BC number 255 is received, the Map Update/Decoder selection task should take no action. However, ITn should be used as a pointer in the BC Bit Map Creation Task (refer to § A.2.1.1.3).

It should be noted that some of the connection/type changes are not strictly permissible under the assignment rules specified in the DCME transmit unit structure. These transitions, however, although abnormal, may occur at the DCME receive unit as a result of loss of assignment messages. Note that the abnormal transitions are different from erroneous assignment messages (rejected by the Input Pre-Processing Task).

Another function of the task discussed in this section is the ADPCM decoder selection (and consequent update of the Dec Array). The rules for the decoder selection should be as follows:

a) the ADPCM decoder selection should be performed only if the remote IT number is destined for DCME;

b) when a new assignment of a previously disconnected IT is made (this includes the re-assignment from "Bank" type to other type), an ADPCM decoder should be selected from the available decoders of the ADPCM decoder pool;

c) when a re-assignment of a previously connected IT to a different BC is made, the ADPCM decoder currently associated with the IT should be maintained;

d) whenever an IT connection changes to BC No. 0 (disconnection), the ADPCM decoder associated with the IT should be released to the decoder pool.

The Map Update/Decoder Selection Task generates the output messages of signal path 54 (except the "Mode Map"), signal path 52 and signal path 4. The rules for the generation of these messages should be as follows:

a) The messages below should only be generated if the received remote IT number is destined for the DCME.

b) When the IT connection changes to a different BC (not No. 0) and/or when the BC type changes, the "Seize" message should be generated, if the BC type is "Transparent" or "Data". The "Seizev" message should be generated, if the BC type is "Voice". In both cases, the BC, IT and the selected ADPCM decoder number should be included in the message. The ADPCM decoder mode (included in the "Seize" message) for "Transparent" and "Data" BC types should be 8- and 5-bit, respectively.

c) When an ADPCM decoder is released to the decoder pool, the "Release" message should be generated for that ADPCM decoder.

d) The "Rxdata" message should be generated only when a transition occurs from a BC type other than "data" to "data".

e) The "Rxtranspreq" message should be generated when the transition occurs from another BC type to "transparent".

f) The "Rxtransprel" message should be generated when the transition occurs from a transparent BC type to a different type.

A.2.1.1.3 *BC Bit Map Creation Task*

This task performs two actions:

a) derivation of the 5th bit of each data channel (from the bit banks),

b) derivation of the overload BCs from the bearer BCs.

As an output, these tasks generate the "BC Bit Map" and the "Mode Map" messages.

The type of each BC is stored in the RUD maps and updated when required. Functionally, this Task rearranges the pre-assigned data BCs, the "Voice" and "Disc" BCs (normal range), the DSI "Data" BCs and the connected overload BCs into the pre-assigned 40 kbit/s list, the voice list, the data list and the overload list, respectively. These lists are the same as those defined for the SBC Process (§ A.1.1.2.2). In the SDL representation in § A.3 of the RUD process, lists other than voice lists and overload list are assumed to be generated from the Type Array.

The rules for insertion of the BCs into the various lists and deletion from the lists shall be the same as those defined for the SBC Process. The rules for bit bank handling, overload channel derivation and map update (Mode Map and BC Bit Map) shall also be the same.

The only differences are that when an assignment message is erroneous (or lost):

1) the pointer variables Pv and Pov shall be set to 0;

2) if there is not enough bit capacity available, the affected channels shall receive dummy bits set to 0;

3) the variables N4 or N3 (number of 4-bit or 3-bit overload channels) shall be set to 0 if its calculated value is negative.

A.2.1.2 *Bit Map Implementation Process (BMI)*

The BMI Process input/output connections are shown in Figure A-13/G.763. This process receives the BC Bit Map (signal path 53) from the RUD Process, the "Process-reset" signal (signal path 59) from the MCH Handler and a "trigger" pulse (signal path 60) which indicates that the process output message should be delivered to the hardware.

The function of the BMI Process is to delay the incoming BC Bit Map message before sending the delayed contents in the "Addressmap for BCs" message. The delay is such that the BC Bit Map is implemented at the beginning of the DCME frame which occurs three frames after the start of the DCME frame containing the applicable assignment message. Refer to Figure A-11/G.763.

The "Addressmap for BCs" message (signal path 57) contains the exact bit association required to connect the appropriate bits of the bearer BCs to each ADPCM decoder.

A.2.1.3 *Decoder Control Process (DEC)*

The DEC Process input/output connections are shown in Figure A-13/G.763. The process receives "Seize", "Seizev", "Release" and "Mode Map" messages (signal path 54) from the RUD Process, the "Process-reset" message (signal path 61) from the Map Change Handler and the "Trigger" message (signal path 55). It generates the "Setcod" message (signal path 56) for the Decoder Unit.

At initialization, the DEC Process should receive "Seize" message for 8, 5, 4 or optionally 3 or 2-bit pre-assigned channels from the RUD Process. This message allocates ADPCM decoders to pre-assigned channels, indicating the connection to the IT and the ADPCM decoder mode.

The DEC Process is assumed to be associated with each ADPCM decoder of the decoder unit so that conceptually, there are as many processes as there are ADPCM decoders. In practical implementations, one process can be time-shared among ADPCM decoders.

The DEC Process sets the operating parameters of the ADPCM decoder to which it is dedicated, based on the messages received. The ADPCM decoder operating parameters indicate the IT connection, the 8, 5, 4, 3 or optionally 2-bit mode and whether the ADPCM decoder needs resetting. Resetting shall be performed when the IT connection to an ADPCM decoder is changed (the decoder must be reset before establishing a new connection).

When the "Seize" or "Seizev" message is received, the DEC Process should determine whether the ADPCM decoder number in the message is the same as the ADPCM decoder number to which the process is dedicated. If the number is different, no action should be taken. If the number is the same, the ADPCM decoder parameters should be set in accordance with the IT number and mode (only for the "Seize" message).

The BC Mode Map (signal path 54) received from the RUD Process should be scanned to determine the 4, 3 or optionally 2-bit mode of the ADPCM decoders connected to voice BCs.

Reception of the "Release" message for an ADPCM decoder should cause the decoder to be designated as disconnected.

The ADPCM decoder operating parameters, established by the DEC Process, should be sent to the Decoder Unit via the "Setcod" message. Each "Setcod" message (signal path 56) is destination directed to an ADPCM decoder (decode). The message "Setcod" (decode, IT, mode, reset) indicates the IT connection for the ADPCM decoder as well as the 8, 5, 4, 3, or optionally 2-bit mode of operation and whether the ADPCM decoder needs resetting. The message "Setcod" (decode, 0, etc.) indicates that the ADPCM decoder must be disconnected.

The "Setcod" message for pre-assigned channels should be sent immediately after initialization. The "Setcod" message for DSI channels should be sent such that the ADPCM decoder connection/mode is switched at the beginning of the DCME frame which occurs three frames after the start of the DCME frame containing the applicable Assignment message. Refer to Figure A-11/G.763.

A.3 *An example of DCME SDL diagrams*

Note that in this annex, the intermediate trunk (IT) is indicated as IC and the bearer channel (BC) as SC. Also note that in this example the use of 24 and 16 kbit/s pre-assigned channels for maintenance are not included.

The diagrams are consistent with the CCITT functional specification and description language (SDL) defined in Recommendation Z.100.

In SDL representations of processes, transitions between states are assumed to occur instantaneously. Practical implementations must account for the time delays.

A.3.1 *Logic diagrams for DCME transmit side*

The logic diagrams in this section of the annex A.3 are supplemental to the description of the DCME transmit side structure given in § A.1. The transmit portion of the assignment procedures have been divided into two blocks:

a) IPS – Input processing and service request generation block

b) SRH – Service request handling block

A.3.1.1 *The IPS block*

The HSC process operates on an IC-by-IC basis. This means that the protocol described below will exist separately for each of the transmit ICs. The following signals are included in the block.

L1: Act, Inact

L2: Data-detect, Voice-detect

L3: Signal-detect

L4: Rxdata

L5: Transpreq, Transprel

L6: Process-reset

L10: Not used

L11: Not used

L12: Voice (Integer), Voiceinact (Integer), Data (Integer), Datainact (Integer), Transp (Integer), Discreq (Integer)

L17: Reset-act, Default-voice, Default-data,
Reset-signaldetect

The following states have been defined within the process included within the block:

– (HSC) hangover control and signal classification process (O,)

Voice-inactive, Voice-active, Voice-wait, Data-active, Data-inactive, Data-wait, Transp, Signalling-active, Signalling-wait, Preassigned, Voice-wait-hold, Signalling-active-hold, Voice-inactive-hold, Voice-active-hold, Signalling-wait-hold, Wait-for-data

The signals arriving from all functional units external to the block are assumed not to require IC addressing in order for the correct process instance to respond to the signal for its IC. There is one exception to this. Address information is required for the signals arriving from the Rx assignment block.

L1 – "Act" and "Inact" signals (for each IC) are received from the transmit activity detector. This unit is assumed to operate according to the following principles:

– The unit has two internal states (for each IC), Activity and No-Activity. At the transition from No-activity to Activity, an Act signal is sent. At the transition from Activity to No-activity, Inact is sent.

– The activity detector is reset to No-activity for an IC by a "Reset-act" signal for that IC.

L2 – "Data-detect" or "Voice-detect" signals are received from the data/speech discriminator for each IC. This unit is assumed to operate according to the following principles:

– All activity apart from data and 2100 Hz tones will be declared as voice.

– 2100 Hz tones and actual data transmissions will be declared as data.

– Inactivity will retain the previous declaration (data or speech) until a new declaration is made due to activity or reset signals.

– The Default-voice signal is used to reset the data/speech discriminator to voice.

– The Default-data signal is used to reset the data/speech discriminator to data.

– The unit has two internal states (data, voice). A signal is sent at the transition between these two states.

L3 – "Signaldetect" is received from the 2400 Hz tone detector for each IC. The unit is assumed to operate according to the following principles:

– The unit is assumed to have two states, Signal-detect and No-signal-detect. A signal will be sent at the transition between No-signal-detect to Signal-detect (see Note).

  Note – Detection of 2400 Hz causes the detector to transit to Signal-detect from No-Signal-detect. When the detector ceases to detect 2400 Hz, it should transit from Signal-detect to No-Signal-detect.

– The unit will be reset to No-Signal-detect by the reception of a Reset-signal-detect signal for that IC.

L4 – Rxdata is received from the receive side assignment handling process and indicates that the dataflag has been set for a certain IC.

L5 – Transpreq and Transprel arrive from the TCH. They will be issued after the TCH receives a request for connection or disconnection of a 64 kbit/s transparent channel from the local ISC or via the receive assignment messages from the remote DCME.

L12 – The HSC Process issues six different signals to the SRH block. These are:

– *Voice (Integer)* – Indicates a transition from voice-inactive to voice-active for an IC.

– *Voiceinact (Integer)* – Indicates a transition from voice-active to voice-inactive for an IC.

– *Data (Integer)* – Indicates a transition from data-inactive to data-active for an IC.

– *Datainact (Integer)* – Indicates a transition from data-active to data-inactive for an IC.

– *Transp (Integer)* – Indicates a transition from the previous state of an IC to a transparent condition.

– *Discreq (Integer)* – Indicates a transition from the transparent condition to voice-inactive for an IC.

L17 – Default-voice, Default-data, Reset-act, and Reset-signal-detect are reset signals.

L6 – The function of this signal is associated with map changes. The following assumptions are here made regarding the IPS block.

A map-changer-handler (MCH) will exist external to the assignment protocol. This process will control the configuration data for the IPS block. At start-up, this process will issue signals making it possible to configure the system correctly. The same will be done at a map change instant. The signal used at the map change instant is:

– *Process-reset* – The L6 signal will cause a termination of the process that receives it.

A.3.1.1.1 *Handling of the IPS processes at initialization*

A number of variables are passed from the map-change-handler to the HSC process when they are created by the map-change-handler. This occurs at system start-up or after a map change. These variables are:

– *ch* – The IC number associated with the process.

– *hot, sh, lh* – sh and lh are the hangover values associated with speech. The hangover is set to "sh" provided that the active signal duration preceding it is shorter than "hot". If this is not the case, "lh" is applied.

– *bhot, bsh, blh* – These are similar variables associated with periods of signalling.

– *dh* – This is the first hangover value nominally associated with data and is considered to be in the order of 14 seconds. It is operator settable.

– *dhs* – This is the second hangover value associated with data and is operator settable.

– *pre* – A Boolean variable which passes on the information whether the IC is pre-assigned or not.

– *RAGPID* – A process identifier variable which is used to address the signals to the SRH block to the correct process instance.

The process uses the following variables:

– *t1, t2* Time variables, stores the current time to be used for hangover handling.

– *d1* – Difference between two times.

– *already* – A Boolean variable used to check that the first data hangover value has only been applied once before starting to use the second hangover value.

– *ti, tia* – Timer variables.

A.3.1.2 *The SRH block*

This block contains four different processes. The following signals are used within this block:

L7: Setcod (Integer, Integer, Boolean)

L8: Assign (Integer, Integer, Call-type, Integer)

L9: Addressmap-for-SCs

L10: Not used

L11: Not used

L12: Voice (Integer), Voiceinact (Integer), Data (Integer), Datainact (Integer), Transp (Integer), Discreq (Integer)

L13: Assign (Integer, Integer, Call-type, Integer), Reinsert (Integer), Remove (Integer), Seizesc (Integer), Integer, (Integer), Release (Integer), Releasesc (Integer), Seizebank (Integer)

L14: SC-bitmap

L15: Mode-map (Integer)

L16: Assign-enc (Integer, Integer, Call-type), Release-enc, Set-pre (Integer, Integer)

L18: Not used

L19: Trigger, Sync-Trigger

L20: Trigger

L21: Process-reset

L22: Process-reset

L23: Trigger

L24: Process-reset

L25: Trigger

L26: Process-reset

L200: Change (Integer)

L201: Sync-Alarm (Integer)

L300: Go-ahead

The processes of the SRH block and their states are the following:

a) *(RAG) request handling and assignment information generation process (0.2)*

   No-messages-in-queue, Messages-in-queue, Wait-for-next, Wait-for- sync

b) *(SBC) SC bit map creation process (0.2)*

   Wait

c) *(BMI) bit map implementation process (0.2)*

   Wait

d) *(ENC) encoder control process (0.)*

   Wait-for-signal

The service request handling uses the following symbols:

sat(nr)=bc – The array "sat" uses the transmit IC number to index the SC number to which it is connected. The array is initialized to zero for all the IC numbers used by the DCME at system start-up.

ic(bc)=nr – This array uses the SC number to index the IC number to which it is connected. This array is initialized as zero at system start-up.

typ(bc)=call-type – This array uses the SC number to index the type of connection that is connected to the SC number in question. The values of "call-type" are defined as follows:

"disc", "voiceavail", "voice", "dataavail", "data", "transp", "bank", "preassigned".

The array is initialized as "disc" for all channels at system start-up.

cod(nr)=cd – This array uses the IC number to index the physical coder number to which it is connected. All elements are initialized to zero at system start-up.

The signals used are explained as follows:

L7 – Setcod (Integer, Integer, Boolean) – This signal is issued by the ENC Process and causes the encoder associations to be made. The values delivered are the following:

IC number, Mode (3,4,5 or 8), and a Reset command.

This last variable will be TRUE if a Reset is to be made, FALSE otherwise.

L9 – Addressmap for SCs – Contains the bit addresses to use on the bearer. The signal is a delayed version of the SC-bitmap signal.

L12 – Voice (Integer), Voiceinact (Integer), Data (Integer), Datainact (Integer), Transp (Integer), Discreq (Integer).

L8, L13 –Assign (Integer, Integer, Call-type, Integer) – The variables included have the following meaning:

a) The first integer value gives the SC number to which the IC shall be connected.

b) The second integer value gives the IC number to which the SC shall be connected.

c) The third parameter (Call-type) contains the type of channel that is being assigned.

d) This third integer value contains the actual physical encoder to be used.

This signal is sent to the SBC process and to the environment.

L13 – Reinsert (Integer) – This signal is used to reinsert an SC into the voice list within the SBC process when an implicit disconnect of a data call has occurred.

L13 – Remove (Integer) – Removes an implicitly disconnected overload channel from the SBC overload channel list.

L13 – Seizesc (Integer, Integer, Integer) – Generates a fixed association between an SC number and an encoder number for a pre-assigned channel. The first variable contains the SC number; the second variable contains the encoder number to be used, and the third variable contains the mode (4/5/8).

L13 – Releasesc (Integer) – This signal releases a bit bank connection and is given to the SBC Process. The integer value identifies the SC to be released.

L13 – Seizebank (Integer) – This signal notifies the SBC Process that a certain SC has been seized as a bit bank. It is only used in association with initialization. The integer value indicates the SC that is used as a bit bank.

L13 – Release (Integer) – This signal updates the resource maps within the SBC Process.

L14 – SC bitmap – Contains the bitmap positions for the various channels. This is used to assemble the bearer channel from the output of the different encoders.

L15 – Mode Map (Integer) - This signal is issued by the SBC Process to the ENC Process in order for the correct encoder mode (3/4) to be set for voice connections. The variable contains the mode.

L16 – Set-pre (Integer, Integer) – Seizes a coder for a certain connection. The variables contained imply:

– mode (4/5/8), IC number.

L16 – Assign-enc (Integer, Integer, Call-type) – The variables included have the same meaning as the first three variables defined above for signal L8, L13 Assign (Integer, Integer, Call-type). The signal is sent to ENC process.

L16 – Release-enc – Causes the encoder, identified by the integer value, to release any connection it may have.

L19 – Trigger, Sync-trigger – Trigger signal occurs once every 2 msec period. The Sync-Trigger signal informs the process that the next 2 msec period is the first frame in the DCME multiframe structure. When the Sync-Trigger is present, the Trigger signal is suppressed.

L20, L23, L25 – Trigger – These signals are assumed to occur once every 2 msec period.

L21, L22, L24, L26 – Process-reset – This signal is generated by the map-change-handler in association with a map change and causes a termination of the process that receives it.

L200 – Change (Integer) – This signal arrives from the USM module and contains an IC number that is to be loaded into the Priority 0 Queue which is served by the RAG every nth frame.

L201 – Sync-Alarm (Integer) – This signal is sent if there seems to be a logical problem with the multiframe synchronization within the RAG. The Integer variable identifies the pool number for which the alarm has been raised.

L300 – Go-ahead (Pld, Pld, pre-assigned_list, pre-assigned_list, ic access_list, ic_access_list) – This signal is sent from the MCHA2 process to the MCHA1 process at traffic start up or at traffic reconfiguration. The signal contains information regarding the usage of the ILS and applied hangover times and hangover thresholds.

A.3.1.2.1 *The RAG Process*

The RAG Process is created by the Map-change-handler at system start-up or after a map change. Depending on the possible use of one or two pools, one or two instances of the process will be created. The Map-change-handler delivers a number of parameters to the process. The functions of these parameters are explained below:

– *b* – This integer variable contains the total number of 4-bit samples that are contained within the pool.

– *no* – This integer value contains the total number of SCs in the normal range in the pool that are not pre-assigned.

– *pre(i)* – This array contains the IC numbers of pre-assigned channels.

– *cdlist* – This list contains the physical encoder-numbers that the process may choose from when establishing a connection. The encoders that are to be used for pre-assigned connections are not included (see Note).

  Note – A variable instance of type "list" contains a list of integer numbers which can be accessed separately.

– *presc(i)* – This array contains the SCs to which the pre-assigned ICs should be connected. In the case where the pre-assigned IC is 64 kbit/s, only the even numbered SC is contained within the array.

– *premode(i)* – This array contains the mode (4/5/8) associated with each pre-assigned IC.

– *sclist* – This list contains the SCs which can be used by the process. The pre-assigned SC numbers are not included.

– *ptot* – This integer contains the total number of pre-assigned ICs that should be handled by the process.

– *sel(i)* – This array contains the encoder numbers that are to be used for the pre-assigned ICs.

– *bitbank(i)* – This array, with a maximum of 12 entries, contains the SC numbers to be used for bit banks. The SC numbers are maintained in numerically ascending order. At start-up, the array will contain the SC numbers required to handle pre-assigned 40 kbit/s channels.

– *btot* – This integer value contains the total number of bit banks required at any given moment to handle the number of data calls that are connected. At start-up, the variable will contain the number of bit banks required to handle pre-assigned 40 kbit/s channels.

– *sq* – This Boolean variable contains the value TRUE if the optional USM information is to be handled by the RAG Process.

– *n* – This integer value contains the periodicity of the optional USM information handling as the number of frames.

– *ENCPID(i)* – This array uses encoder numbers as indexes and identifies the process identifier applicable to the encoder number process instance.

– *pnr* – This integer variable identifies the pool number of the RAG process instance.

– *s* – This integer variable defines the lowest number of bits/sample permitted. Its value is either 3, for 3-bit encoding, or 2, for 2-bit encoding.

The process uses a number of different procedures and variables. The procedures are only included within the diagrams as procedure calls and the variables as names. Their meaning is described below in order of appearance:

– *Rm* – This variable is TRUE if an SC number is to be removed from an SBC resource list, otherwise FALSE.

– *Prev* – This variable is TRUE if a previous connection exists for another type of call for that IC, otherwise FALSE.

– *Reins* – This variable is TRUE if a SC number is to be reinserted in an SBC resource list, otherwise FALSE.

– *Rethere* – This variable is TRUE if a bit bank has to be created as a function of changing an IC already connected to an SC to "data", FALSE otherwise.

– *Return1* – This variable is TRUE if a re-assignment due to a connection of a transparent call is in progress, otherwise FALSE.

– *Return2* – This variable is TRUE if a re-assignment due to a connection of a transparent call is in progress, otherwise FALSE.

– *i* – Counter.

– *again* – This variable is TRUE if a refreshment message for the current SC should not be generated, otherwise FALSE.

– *r* – Counter.

– *r1* – Counter.

– *nr* – This integer variable stores the IC number associated with an incoming request.

– *ovlr* – This variable is TRUE if an overload channel is to be refreshed, otherwise FALSE.

– *f* – A local counter used to keep track of the frame number within the multiframe number, set to 0 after the reception of a sync trigger signal.

– *Store"X" (nr)* – This procedure stores the variable "nr" at the bottom of the priority queue marked "X".

– *req in queue (nr)* – This array is indexed by the IC numbers and stores the value 0 for a given index if there are no requests for that IC in any of the queues 2 to 5. It stores the value 1 if there is a request for that IC in any of the queues 2 to 5.

– *pr"X"count* – A variable which stores the number of requests that exist in priority queue "X".

– *req in discqueue(nr)* – This array is indexed by the IC numbers and stores the value 0 for a given index if there are no requests for that IC in queue 1. It stores the value 1 if there is a request for that IC in queue 1.

– *Remove from RAG queue (nr, more)* – This procedure removes any request for the IC "nr" from any of the queues 2 to 5. The pr"x"count variable in the queue is updated accordingly. The procedure also stores the value TRUE in the variable More if there is at least one request in any of the five queues after the removal has been performed, otherwise More is set to FALSE.

– *Additional-messages (more)* - This procedure checks if there are any messages remaining in queues 1 through 5. If this is the case, the variable "more" is set to TRUE, otherwise FALSE.

– *Read"X" (nr)* – This procedure reads the IC at the top of the "X" queue and delivers this value in the variable "nr".

– *Pop"X" (pr"X"count, more)* – This procedure removes the IC value at the top of the queue and pops the queue by one step. It updates the pr"X" count variable for that queue and delivers the value TRUE in the variable More if there is at least one request in any of the queues 1 through 5 after this operation is performed, otherwise More is set to FALSE.

– *Count data (difference)* – This procedure checks the number of 40 kbit/s pre-assigned channels and 40 kbit/s data channels that exist and compares it to the number of bit banks that are in use. If it is possible to delete a bit bank, the variable "difference" is given as TRUE, otherwise FALSE.

– *Count (nt, nd, nb, nv)* – This procedure checks the "typ" array and delivers the number of transparent calls currently being handled in the variable "nt", the number of data calls currently being handled in the variable "nd" and the number of bit banks in use in the variable "nb" and the number of voice calls currently being handled in the variable "nv".

– *d* – This variable is used to store the mean number of bits per voice call that the handling of an additional call would result in, or the total number of bits in the frame left for use after handling a request.

– *Search transp (bc, nr, cd, nrv1, nrv2, bcv1, bcv2, nrv3, nrv4, bcv3, bcv4, success)* – This procedure searches the array "typ" for a possible place to connect a transparent call. There are ten possibilities which are searched for in descending order of priority. It cannot be guaranteed that the search will find at least one of these possibilities even when the check of available bits has been successfully passed. If the search fails the variable success is given as FALSE, otherwise TRUE. The procedure delivers the results of the search in a number of parameters that indicate if action is to be taken. The variables have the following meaning:

  1) *bc* – The even SC number which the transparent call is to be assigned to.

  2) *bc + 1* – The SC number just above that will also be used by the transparent call (a derived variable).

  3) *nr* – The IC number containing the transparent call.

  4) *cd* – The encoder number picked by the procedure from the pool of available encoders. It should be noted that there may be a need to pick-up the encoder from one of the selected channels if this is declared "voiceavail" or "dataavail". Specifically, if "cod(nr)" is not equal to 0, cd = cod(nr).

  5) *nrv1* – The IC already connected to "bc".

  6) *nrv2* – The IC already connected to "bc+1".

  7) *bcv1* – The SC that "nrv1" is to be re-assigned to.

  8) *bcv2* – The SC that "nrv2" is to be re-assigned to.

  9) *nrv3* – The IC already connected to "bcv1".

  10) *nrv4* – The IC already connected to "bcv2".

  11) *bcv3* – An overload SC that "nrv3" is to be re-assigned to.

  12) *bcv4* – An overload SC that "nrv4" is to be re-assigned to.

  13) *success* – Result of the search (TRUE or FALSE).

– *k* – The SC that the IC was previously connected to before changing to a transparent/data/voice call.

– *tk* – A temporary variable used to store the value of "k".

– *tnr* – A temporary variable used to store the value of "nr".

– *Check for additional bit bank (new)* – This procedure checks if a bit bank is required if one additional data call was to be handled by the bearer. If this is the case, the variable "new" is set to 1, otherwise to 0. It should be noted that if at least one SC declared "data-avail" exists, "new" is always equal to 0.

– *Make room in bitbankarray (nw, bc, bitbank)* – This procedure handles the bitbankarray in order to make it possible to inset "bc" in its correct position, thus keeping the SC numbers used for bit banks in ascending numerical order. The entry which should have the value "bc" associated with it is given in the variable "nw".

– *Search data (bc, nr, cd, nrv, bcv, data success)* – This procedure searches the "typ" array for possible places to connect a data call. There are four possibilities which are searched for in descending order of priority. It also checks the need for an overload channel re-assignment to handle a data connection request. If an overload channel re-assignment is not needed, the variable "data success" is given the value TRUE, otherwise it is given the value FALSE. The procedure will deliver its results in variables having the following meaning:

  1) *bc* – The SC number that the data call is to be connected to.

  2) *nr* – The IC number that is to be connected as a data call.

  3) *cd* – The encoder number picked by the procedure. It should be noted that there may be a need to pick an encoder freed by the use of a channel declared "Voiceavail" or "Dataavail". Specifically, if cod(nr) is not equal to 0, cd = cod(nr).

4) *nrv* – The IC number previously connected to "bc".

5) *bcv* – The SC number that "nrv" is to be re-assigned to. This number is always an overload SC number.

6) *data success* – Result of the check for an overload channel re-assignment requirement (TRUE not needed, FALSE needed).

– *Search voice (bc, nr, cd, nrv)* – This procedure searches the "typ" array for possible places to connect a voice call. There are three possibilities which are searched for in descending order of priority.

The results of the procedure are delivered in variables having the following meaning:

1) *bc* – The SC number that the voice call is to be assigned to.

2) *nr* – The IC number containing the voice call.

3) *cd* – The encoder number picked by the procedure. It should be noted that there may be a need to pick an encoder freed as a consequence of handling that request. Specifically, if cod(nr) is not equal to 0, cd = cod(nr).

4) *nrv* – The IC number previously connected to "bc".

– *SBCPID* – The process identifier variable which is used to address the signals to the correct SBC process.

– *Check_overload_reassignment_when_data (data_success)* – This procedure checks the need for an overload re-assignment to handle a data connection request. If an overload channel re-assignment is not needed, the variable data_success is given the value TRUE, otherwise it is given the value FALSE.

– *Check_overload_reassignment_when_transp (nr,success)* – This procedure checks the need for an overload re-assignment to handle a transparent connection request. If an overload channel re-assignment is not needed, the variable data_success is given the value TRUE, otherwise FALSE. It is noted that if "nr" is "voice" or "voice_avail", two voice channels may be taken from an overload channel creating voice channel pool, and if "nr" is "data" or "data_avail", only one channel may be taken from the pool.

A.3.1.2.2 *The SBC Process*

This process is created by the RAG Process and receives at its creation three parameters as input. These are:

– *sclist* – The current list of SC numbers that is used by this pool. The pre-assigned SCs are not included within this list.

– *bt* – The total number of 4-bit samples within the pool. This number determines the maximum number of SC numbers that can exist, namely:

bt + Integer [bt/3]

This is required for handling the various maps and arrays within the process.

– *ENCPID (i)* – This array uses encoder numbers as indexes and identifies the process identifier applicable to that encoder number process instance.

The following internal resource maps exist within the process:

Voicelist, Overloadlist, Datalist, Transplist, Preassign40list, Preassign64list, Preassign32list, Banklist, Coder (Integer).

The functions and the internal rules governing their use are contained within the specification.

The following parameters and procedure calls are made use of within the process:

– *Generate maps* – This procedure takes the input parameters and generates the various lists and arrays in accordance with the rules regarding initialization of these lists and arrays.

– *Change coder array (cod)* – This procedure goes through the coder array until it finds a SC indexing the coder number "cod". This SC number has its coder number set to zero.

– *Include in voicelist and extract (b)* – This procedure takes the SC number "b" and includes it in the appropriate place in the voicelist and removes it out of any other list that it might exist in. If it finds the SC number in question within the transplist, this should be extracted after which the SC number "b+1" should also be inserted into the voicelist. If "b" is already included in the voicelist, no action shall be taken.

– *Delete overload (b)* – This procedure removes SC number "b" from the overloadlist. If "b" is not included in the overloadlist, no action shall be taken.

– *Generate address mode (bit map)* – This procedure generates the bitpositions and the various modes associated with the encoders. It uses the "ic" variable defined by the received Assign signal as a pointer. This is done in accordance with the rules outlined within the specification. The output is put into the signals mode-map and SC-bitmap.

– *Preassign40 (b,cod)* – This procedure will include SC number "b" in the preassign40list and set the coder array entry for "b" to "cod".

– *Preassign64 (b,cod)* – This procedure will include SC number "b" in the preassign64list and set the coder array entry for "b" and "b+1" to "cod".

– *Preassign32 (b,cd)* – This procedure is used to insert the number "b" into preassign32list upon reception of the signal "seizesc" containing a mode set to 4. It also sets Coder(b) to "cod".

– *Include in banklist and extract (b)* – This procedure will include SC number "b" in the banklist and extract SC number "b" from any other list it might exist in. If already included within the banklist, no action shall be taken.

– *Included in banklist (b,included)* – This procedure checks if SC number "b" is included within banklist or not. If this is the case, the variable "Included" is set to TRUE, otherwise to FALSE.

– *Included in datalist (b,included)* – This procedure checks if SC number "b" is included in the datalist. If this is the case, the variable "Included" is set to TRUE, otherwise to FALSE.

– *Included in transplit (b,included)* – This procedure checks if SC number "b" is included in transplist. If this is the case, the variable "Included" is set to TRUE, otherwise to FALSE.

– *Included in datalist and extract (b)* – This procedure includes the SC number "b" in the datalist and extracts it from any other list it might exist in. If "b" already included in the datalist, no action shall be taken.

– *Update coder association (b,cod)* – This procedure sets the coder array entry for "b" to "cod". It also checks to see if "cod" is associated with any other SC numbers in the array. Should this be the case, the entries for those SC are set to zero.

– *Update coder association for transp (b,cod)* – This procedure sets the entries for "b" and "b+1" to "cod" in the coder array. It also checks the array to see if "cod" is given as an entry for other SC numbers. If this is the case, these entries are set to zero.

– *Include in transplist and extract (b)* – This procedure will extract SC numbers "b" and "b+1" from any list they might exist in and insert SC number "b" into the transplist. If "b" is already included in the transplist, no action shall be taken.

– *Included in voicelist (b,included)* – This procedure checks if SC number "b" is included within voicelist. If this is the case, the variable "Included" is set to TRUE, otherwise FALSE.

– *Included in overloadlist (b,included)* – This procedure checks to see if SC number "b" is included within the overloadlist. If this is the case, the variable "Included" is set to TRUE, otherwise to FALSE.

– *Include in overloadlist (b)* – This procedure includes SC number "b" in the overloadlist. If "b" is already included in the overloadlist, no action shall be taken.
– *i* – counter.
– *md* – This variable indicates the mode (4/5/8) received in the signal siezesc.
– *ic* – The IC number received in the signal Assign.
– *typ* – The Call-type received in the signal Assign.
– *mode(i)* – This array contains the mode of each connection (2/3/4/5/8). The list is updated every DCME frame by the procedure Generate-address-mode.
– *BMIPID* – A process identifier variable which is used to address signals to the correct BMI process instance.

A.3.1.2.3 *ENC Process*

There are as many instances of this process as there are encoders. The processes are created by the Map-change-handler at system start-up. The following variables and procedures are used within the process.

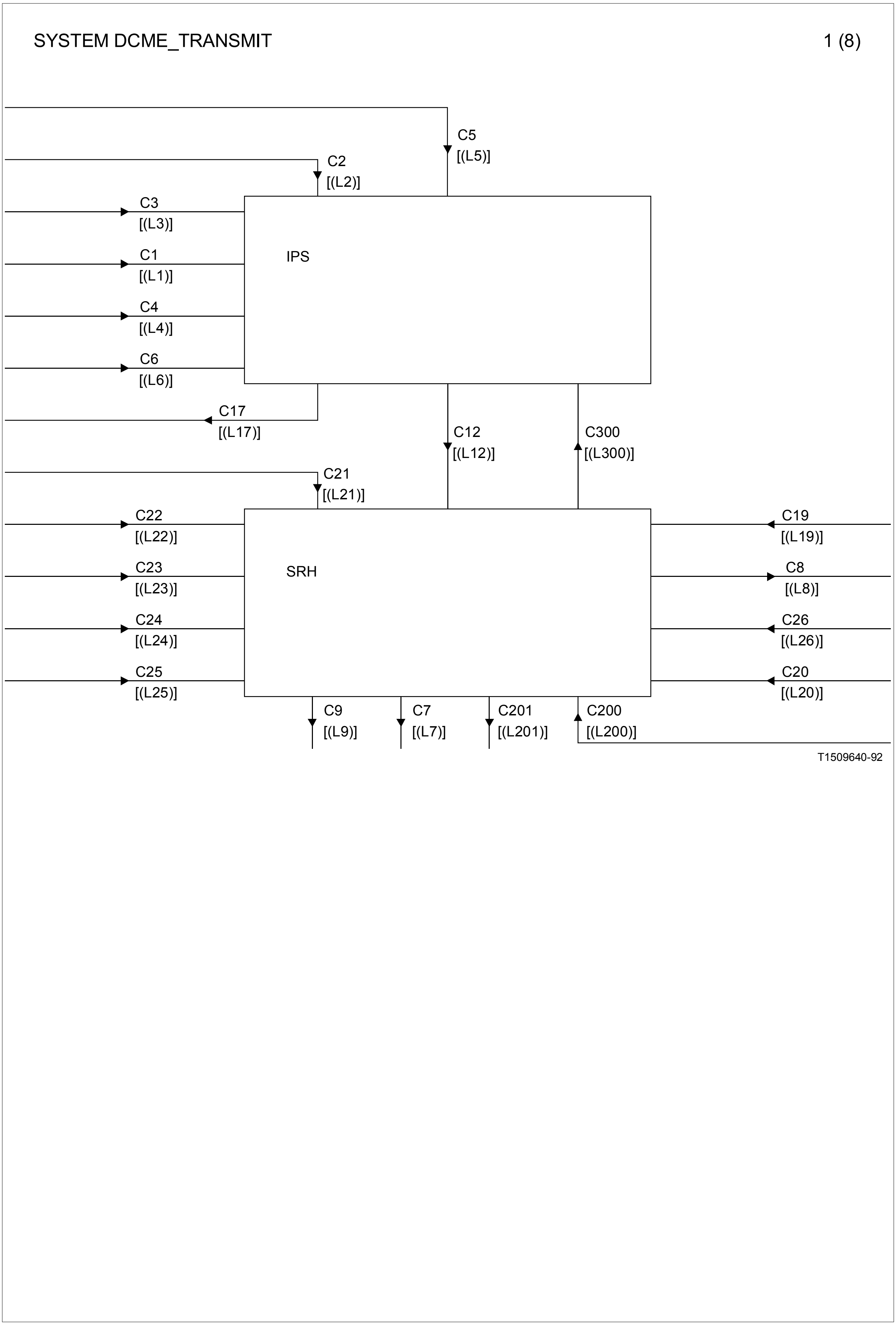
– *change* – This variable stores the value TRUE if data for the encoder has been received since the last Trigger signal. Otherwise FALSE is stored.
– *ic* – The IC number that the unit currently is connected to. Initialized as zero.
– *mode* – The mode of the encoder (2/3/4/5/8 bits/sample). Initialized as zero.
– *reset-coder* – This variable stores the value TRUE if a reset of the coder is to be made, otherwise it stores the value FALSE.
– *fic* – This variable stores a future IC connection for the encoder.
– *fmode* – This variable stores the future mode for the encoder.
– *cic* – This variable stores a current IC connection for the encoder.
– *cmode* – This variable stores a current mode for the encoder.
– *amd* – This variable stores the received mode from the SBC process.
– *cd* – This variable stores the encoder number addressed by the signals Release-enc and Assign-enc.
– *Store(fic, fmode)* – This procedure stores the values of the parameters included at the bottom of a queue. At initialization, this queue shall store 0 for all variables in all positions.
– *Retrieve (cic,cmode)* – This procedure takes the values stored three DCME frames ago from the top of the queue and delivers the results in the variables "cic", and "cmode". The values at the lower queue positions are moved up one place.
– *b* – SC number contained in an Assign-enc signal.
– *nr* – IC number contained in an Assign-enc signal.
– *typ* – The type of connection contained in an Assign-enc signal.
– *md* – The mode contained in a Set-pre signal.

It should be noted that a setcod signal containing ic=0, mode=0 and Reset-coder=False should not have any impact on the encoder being addressed.

A.3.1.2.4 *The BMI Process*

This process is created at system start-up and only delays the signal by three DCME frames. It contains the following internal procedure calls:

– *Store (bit map)* – This procedure takes the information contained within the SC-bitmap signal and places this information at the bottom of a queue. At initialization, the queue shall contain an all zeros address in all its queue positions.
– *Retrieve (bit map)* – This procedure extracts the information queued three DCME frames ago and delivers this to the signal Addressmap-for SCs. The values at the lower positions in the queue are moved up one place.
– It should be noted that an all zeros address in the signal addressmap-for-SCs should not cause any connections to be made between encoder outputs and the bearer.



SYSTEM DCME_TRANSMIT
1 (8)
C5
[(L5)]
C2
[(L2)]
C3
[(L3)]
C1
[(L1)]
IPS
C4
[(L4)]
C6
[(L6)]
C17
[(L17)]
C12
[(L12)]
C300
[(L300)]
C21
[(L21)]
C22
[(L22)]
SRH
C19
[(L19)]
C23
[(L23)]
C8
[(L8)]
C24
[(L24)]
C26
[(L26)]
C25
[(L25)]
C20
[(L20)]
C9
[(L9)]
C7
[(L7)]
C201
[(L201)]
C200
[(L200)]
T1509640-92

```
SYNONYM number_of_ICs Integer=EXTERNAL;
SYNONYM number_of_SCs Integer=EXTERNAL;
SYNONYM number_of_encoders Integer=EXTERNAL;

SYNTYPE ic_range=Natural
  CONSTANTS 1:number_of_ICs
ENDSYNTYPE ic_range;

SYNTYPE sc_range=Natural
  CONSTANTS 1:number_of_SCs
ENDSYNTYPE sc_range;

SYNTYPE encoder_range=Natural
  CONSTANTS 1:number_of_encoders
ENDSYNTYPE encoder_range;

SYNTYPE bitbank_array_range=Natural
  CONSTANTS 1:12
ENDSYNTYPE bitbank_array_range;

SYNTYPE zero_one=Natural
  CONSTANTS 0:1
ENDSYNTYPE zero_one;

SYNTYPE bit_mode=Natural
  CONSTANTS 0,3:5,8
ENDSYNTYPE bit_mode;

syntype RAGPID_type=Natural
  constants 1 : 2
endsyntype RAGPID_type;
```

T1509650-92

SYSTEM DCME_TRANSMIT DATADEFINITION2 (8)

```
syntype rag_queue_type = Natural
  constants 1 : 5
endsyntype rag_queue_type;

newtype Bit_mode_matrix
  literals
    1,
    0;
endnewtype Bit_mode_matrix;

newtype call_type
  literals
    'disc',
    'voiceavail',
    'voice',
    'dataavail',
    'data',
    'transp',
    'bank',
    'preassigned';
operators
  ORDERING;
endnewtype call_type;
```

T1509660-92

```
NEWTYPE ic_to_sc_connections
  Array(ic_range, integer)
ENDNEWTYPE ic_to_sc_connections;

NEWTYPE sc_to_ic_connections
  Array(sc_range, integer)
ENDNEWTYPE sc_to_ic_connections;

NEWTYPE sc_usage_array
  Array(sc_range, call_type)
ENDNEWTYPE sc_usage_array;

NEWTYPE ic_to_coder_connections
  Array(ic_range, integer)
ENDNEWTYPE ic_to_coder_connections;

NEWTYPE preassigned_list
  Array(ic_range, integer)
ENDNEWTYPE preassigned_list;

NEWTYPE encoder_list
  Array(encoder_range, integer)
ENDNEWTYPE encoder_list;

NEWTYPE preassigned_sc_list
  Array(sc_range, integer)
ENDNEWTYPE preassigned_sc_list;
```

T1509670-92

SYSTEM DCME_TRANSMIT DATADEFINITION4 (8)

```
NEWTYPE assigned_mode
  Array(sc_range, bit_mode)
ENDNEWTYPE assigned_mode;

NEWTYPE sc_access_list
  Array(sc_range, integer)
ENDNEWTYPE sc_access_list;

NEWTYPE ic_access_list
  Array(ic_range, integer)
ENDNEWTYPE ic_access_list;

NEWTYPE select_encoder_list
  Array(encoder_range, integer)
ENDNEWTYPE select_encoder_list;

NEWTYPE bitbank_list
  Array(bitbank_array_range, integer)
ENDNEWTYPE bitbank_list;

NEWTYPE request_in_queue_list
  Array(ic_range, zero_one)
ENDNEWTYPE request_in_queue_list;

NEWTYPE sc_to_coder_connections
  Array(sc_range, integer)
ENDNEWTYPE sc_to_coder_connections;
```

T1509680-92

```
NEWTYPE ENCPID_array
  Array(encoder_range, PId)
ENDNEWTYPE ENCPID_array;

NEWTYPE RAGPID_array
  Array(RAGPID_type, PId)
ENDNEWTYPE RAGPID_array;

NEWTYPE queue /* = EXTERNAL*/
ENDNEWTYPE queue;

NEWTYPE rag_queue_array
  Array(rag_queue_type, queue)
ENDNEWTYPE rag_queue_array;
```

T1509690-92

SYSTEM DCME_TRANSMIT SIGNALDEFINITION1 (8)

```
/* Signal definitions */
SIGNAL
  Act, Inact,
  Data_detect, Voice_detect,
  Signal_detect,
  Rxdata,
  Transpreq, Transprel,
  Process_reset,
  Voice(Integer), Voiceinact(Integer),
  Data(Integer), Datainact(Integer),
  Transp(Integer), Discreq(Integer),
  Reset_act, Default_voice, Default_data,
  Reset_signaldetect,
  Setcod(Integer,Integer,Boolean),
  Assign(Integer,Integer,Call_Type,Integer),
  Addressmap_for_SCs(Bit_mode_matrix),
  Trigger, Sync_trigger,
  Change(Integer),
  Sync_alarm(Integer),
  Go_ahead(PId,PId,preassigned_list,preassigned_list,
  ic_access_list,ic_access_list);

/* Signallist definitions */
SIGNALLIST L1 = Act,Inact;
SIGNALLIST L2 = Data_detect, Voice_detect;
SIGNALLIST L3 = Signal_detect;
SIGNALLIST L4 = Rxdata;
SIGNALLIST L5 = Transpreq, Transprel;
SIGNALLIST L6 = Process_reset;
SIGNALLIST L7 = Setcod;
SIGNALLIST L8 = Assign;
SIGNALLIST L9 = Addressmap_for_SCs;
SIGNALLIST L12 = Voice, Voiceinact, Data, Datainact, Transp, Discreq;
SIGNALLIST L17 = Reset_act, Default_voice, Default_data,
                 Reset_signaldetect;
SIGNALLIST L19 = Trigger, Sync_trigger;
SIGNALLIST L20 = Trigger;
SIGNALLIST L21 = Process_reset;
SIGNALLIST L22 = Process_reset;
SIGNALLIST L23 = Trigger;
```

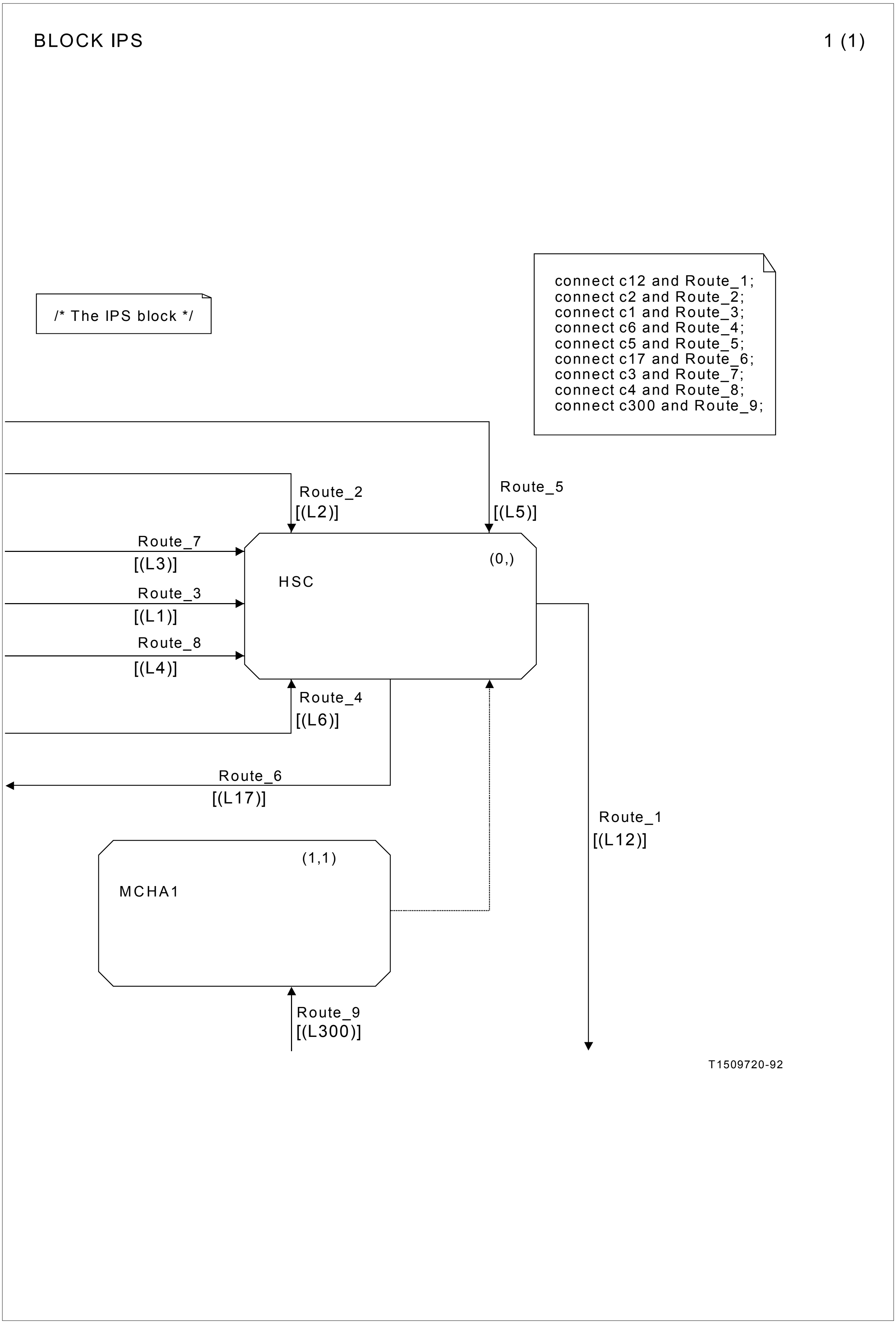
T1509700-92

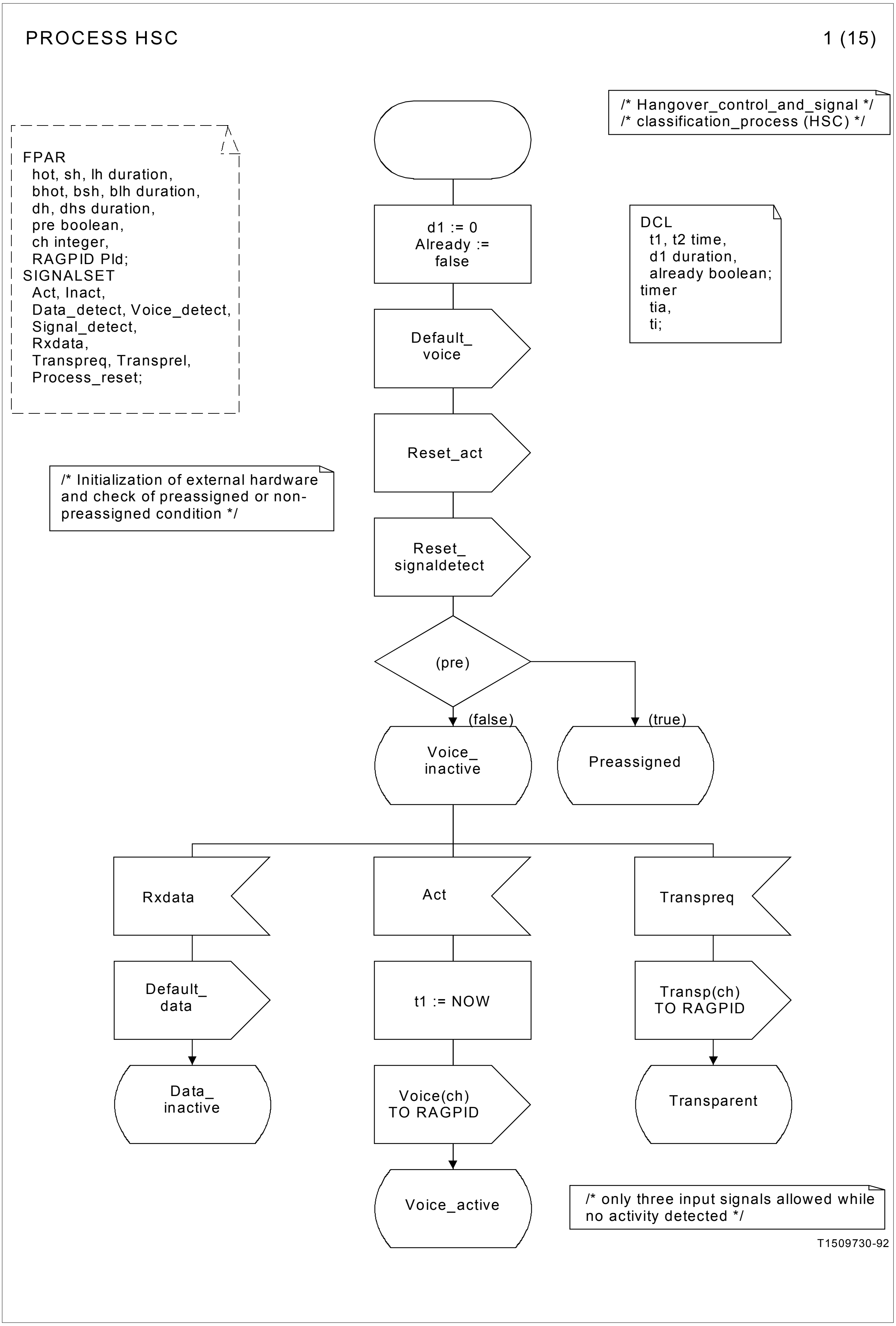
SYSTEM DCME_TRANSMIT

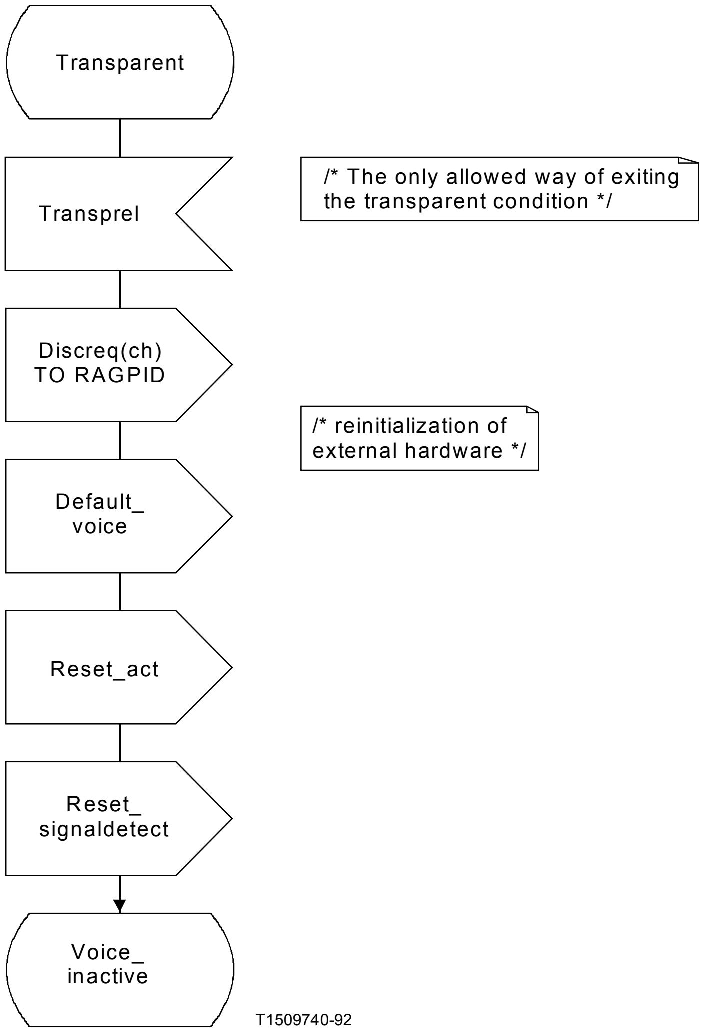
SIGNALDEFINITION2 (8)

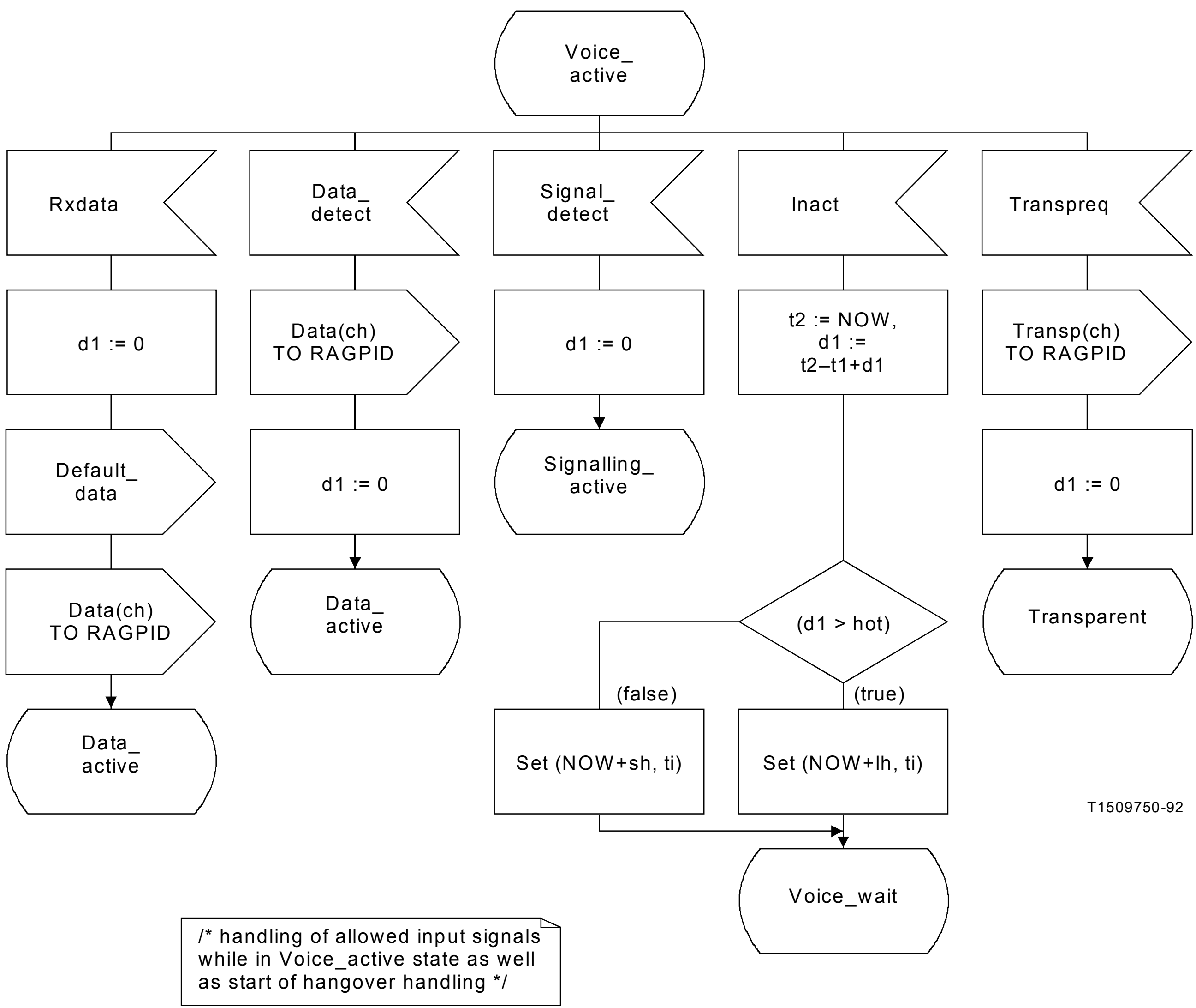
SIGNALLIST L24 = Process_reset;
SIGNALLIST L25 = Trigger;
SIGNALLIST L26 = Process_reset;
SIGNALLIST L200 = Change;
SIGNALLIST L201 = Sync_alarm;
SIGNALLIST L300 = Go_ahead;

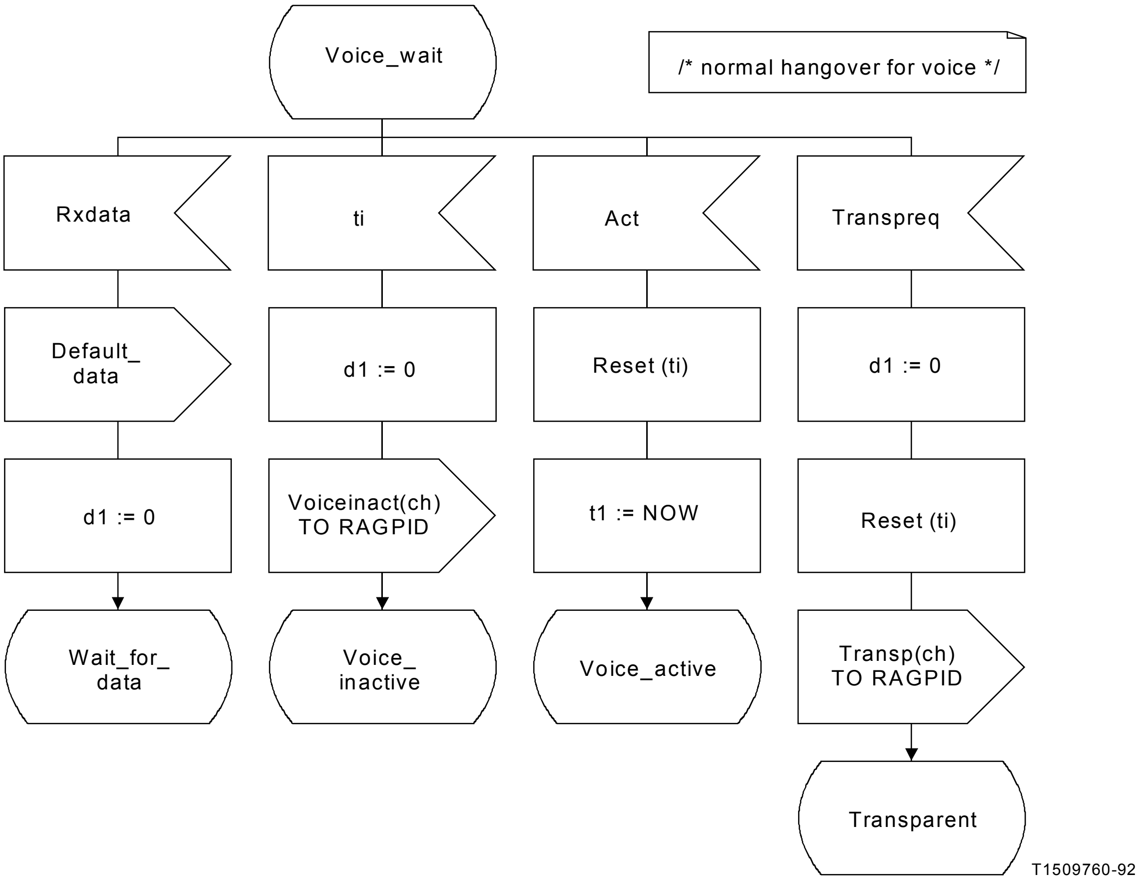
T1509710-92

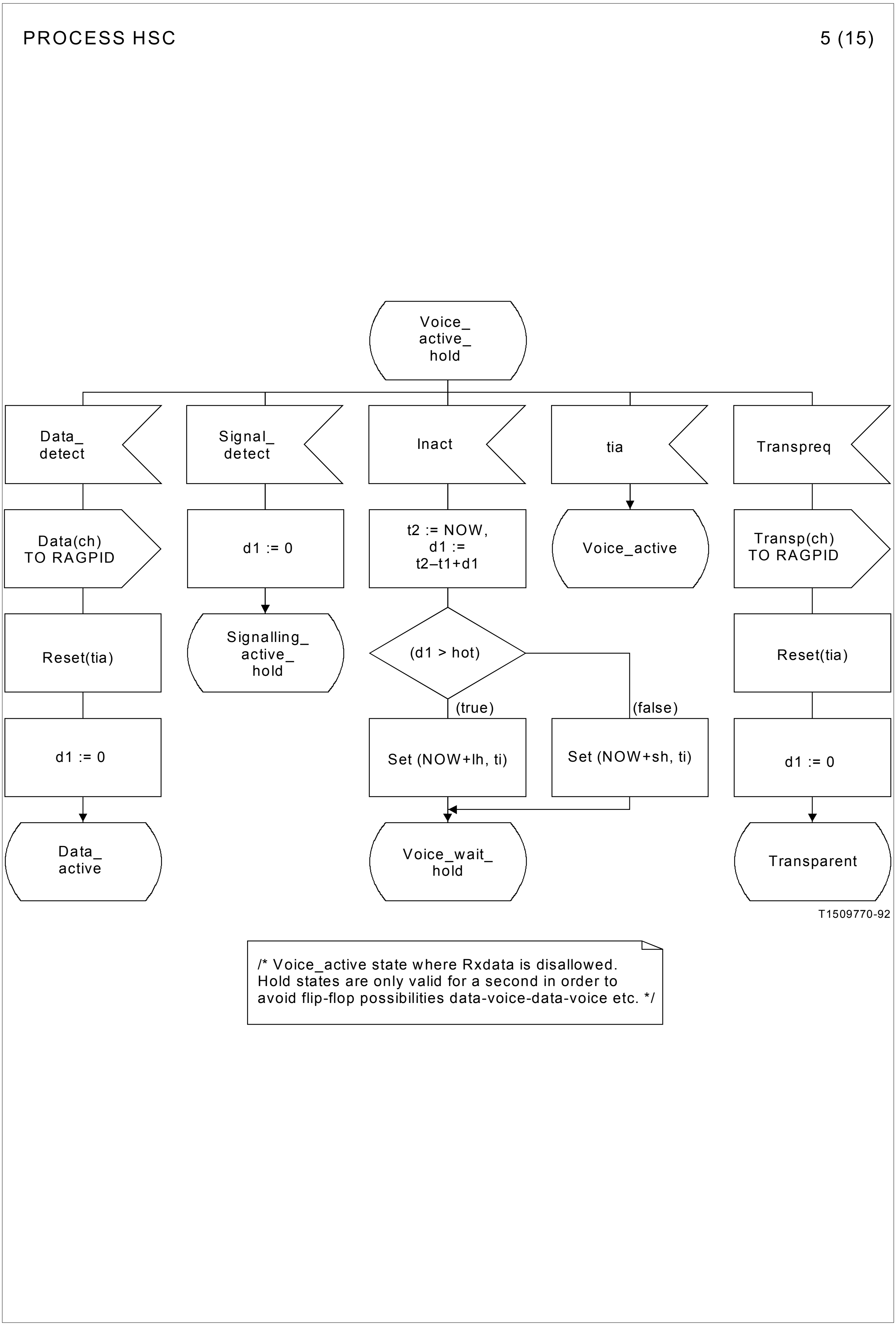

BLOCK IPS
1 (1)
/* The IPS block */
connect c12 and Route_1;
connect c2 and Route_2;
connect c1 and Route_3;
connect c6 and Route_4;
connect c5 and Route_5;
connect c17 and Route_6;
connect c3 and Route_7;
connect c4 and Route_8;
connect c300 and Route_9;
Route_2
[(L2)]
Route_5
[(L5)]
Route_7
[(L3)]
HSC
(0,)
Route_3
[(L1)]
Route_8
[(L4)]
Route_4
[(L6)]
Route_6
[(L17)]
Route_1
[(L12)]
MCHA1
(1,1)
Route_9
[(L300)]
T1509720-92

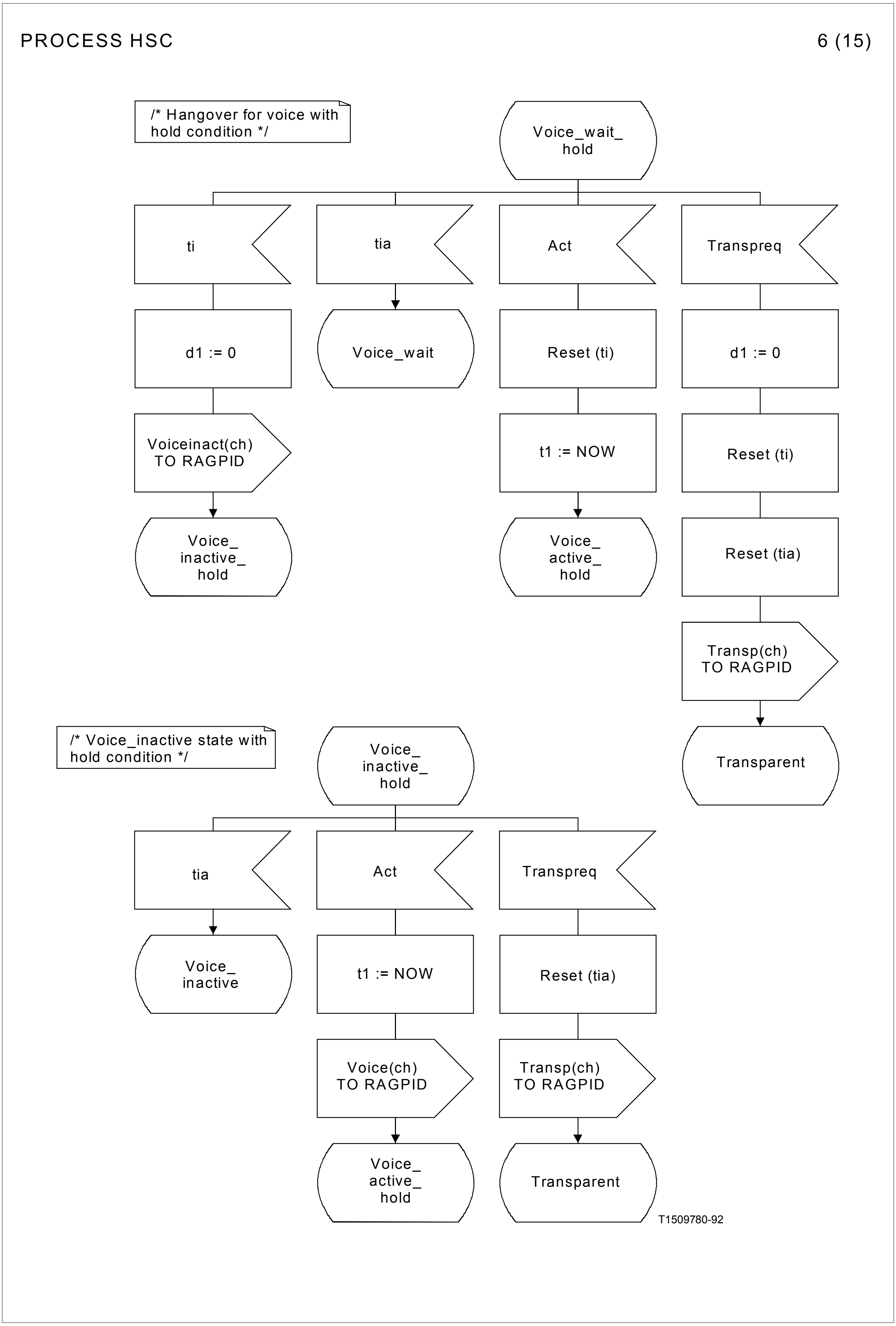

PROCESS HSC
1 (15)
/* Hangover_control_and_signal */
/* classification_process (HSC) */
FPAR
hot, sh, lh duration,
bhot, bsh, blh duration,
dh, dhs duration,
pre boolean,
ch integer,
RAGPID PId;
SIGNALSET
Act, Inact,
Data_detect, Voice_detect,
Signal_detect,
Rxdata,
Transpreq, Transprel,
Process_reset;
d1 := 0
Already :=
false
DCL
t1, t2 time,
d1 duration,
already boolean;
timer
tia,
ti;
Default_
voice
Reset_act
/* Initialization of external hardware and check of preassigned or non-preassigned condition */
Reset_
signaldetect
(pre)
(false)
(true)
Voice_
inactive
Preassigned
Rxdata
Act
Transpreq
Default_
data
t1 := NOW
Transp(ch)
TO RAGPID
Data_
inactive
Voice(ch)
TO RAGPID
Transparent
Voice_active
/* only three input signals allowed while no activity detected */
T1509730-92


Transparent
Transprel
/* The only allowed way of exiting the transparent condition */
Discreq(ch) TO RAGPID
/* reinitialization of external hardware */
Default_ voice
Reset_act
Reset_ signaldetect
Voice_ inactive
T1509740-92

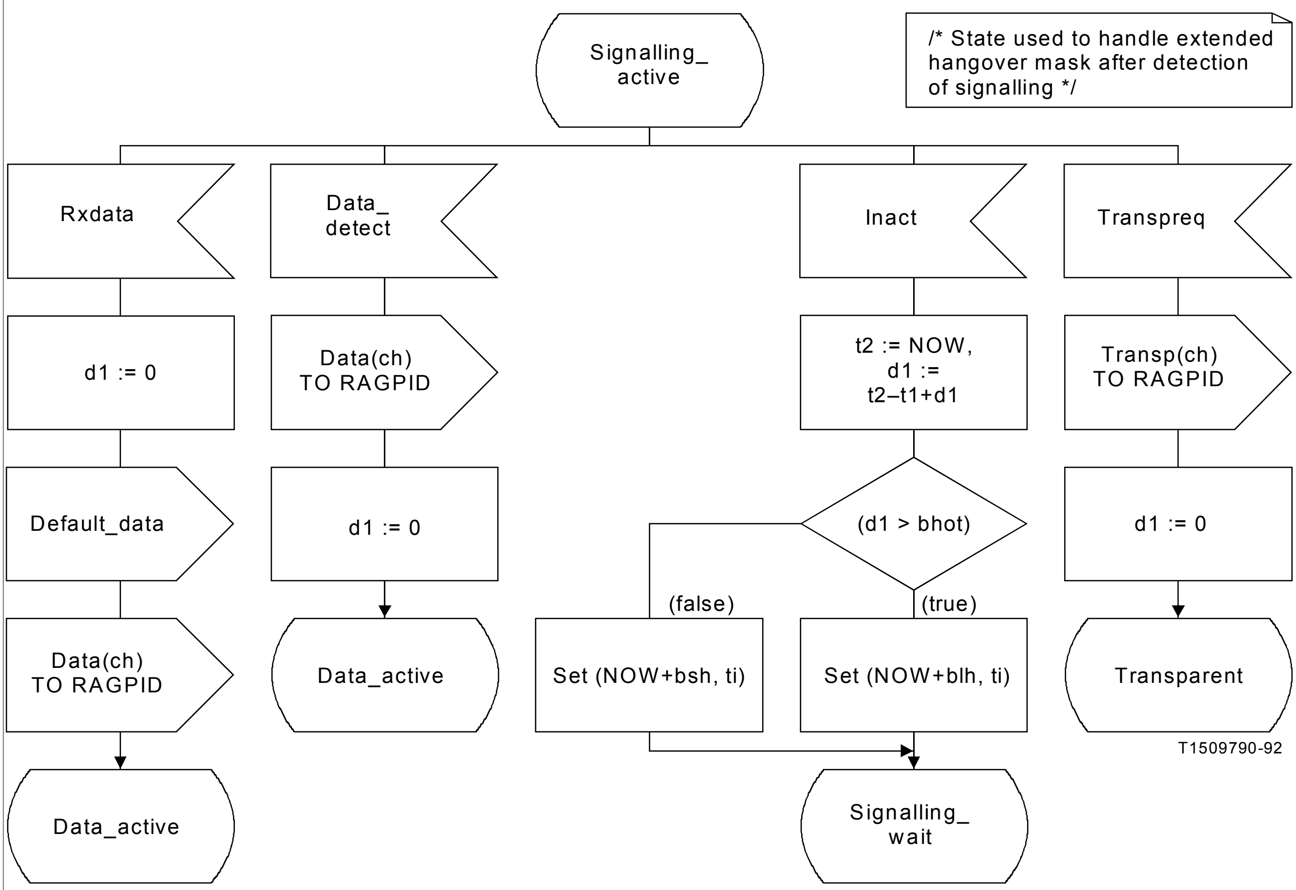

Voice_
active
Rxdata
Data_
detect
Signal_
detect
Inact
Transpreq
d1 := 0
Data(ch)
TO RAGPID
d1 := 0
t2 := NOW,
d1 :=
t2–t1+d1
Transp(ch)
TO RAGPID
Default_
data
d1 := 0
Signalling_
active
d1 := 0
Data(ch)
TO RAGPID
Data_
active
(d1 > hot)
Transparent
(false)
(true)
Data_
active
Set (NOW+sh, ti)
Set (NOW+lh, ti)
T1509750-92
Voice_wait
/* handling of allowed input signals
while in Voice_active state as well
as start of hangover handling */

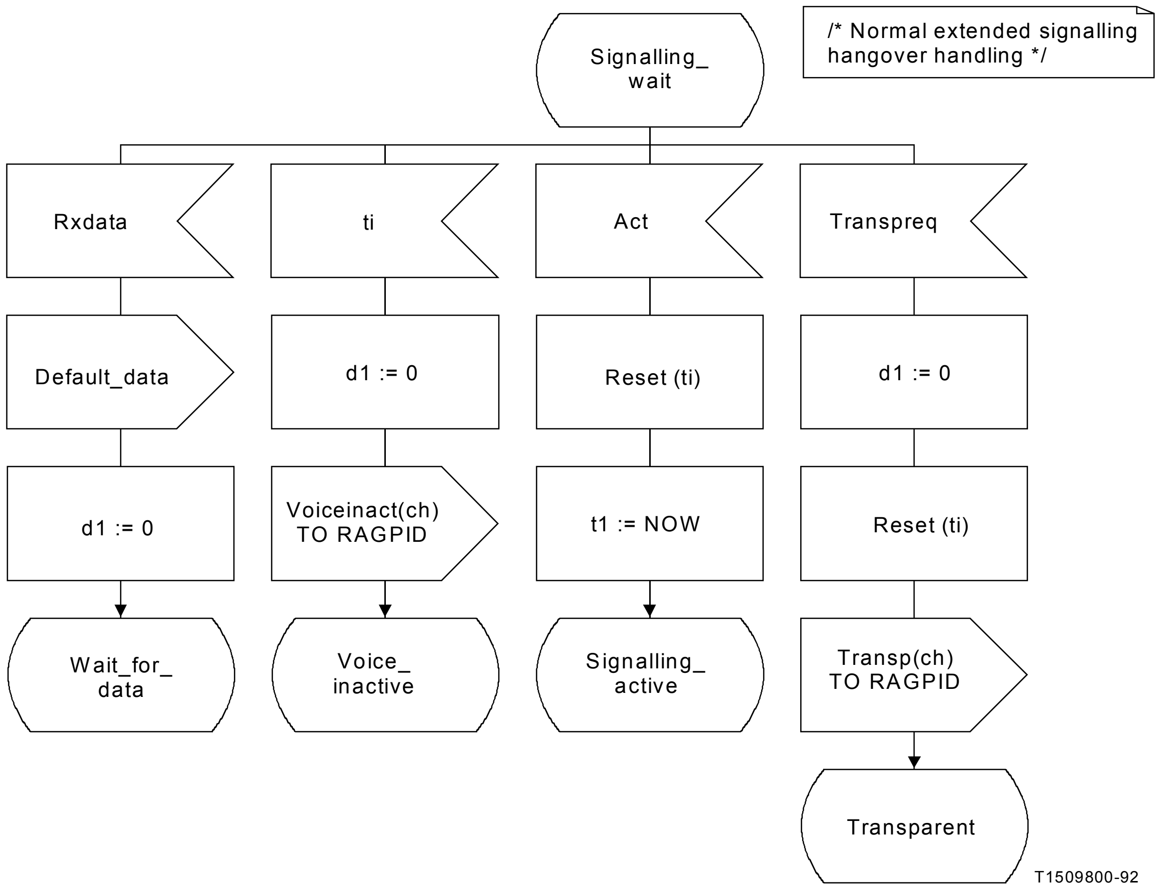

Voice_wait
/* normal hangover for voice */
Rxdata
ti
Act
Transpreq
Default_ data
d1 := 0
Reset (ti)
d1 := 0
d1 := 0
Voiceinact(ch) TO RAGPID
t1 := NOW
Reset (ti)
Wait_for_ data
Voice_ inactive
Voice_active
Transp(ch) TO RAGPID
Transparent
T1509760-92

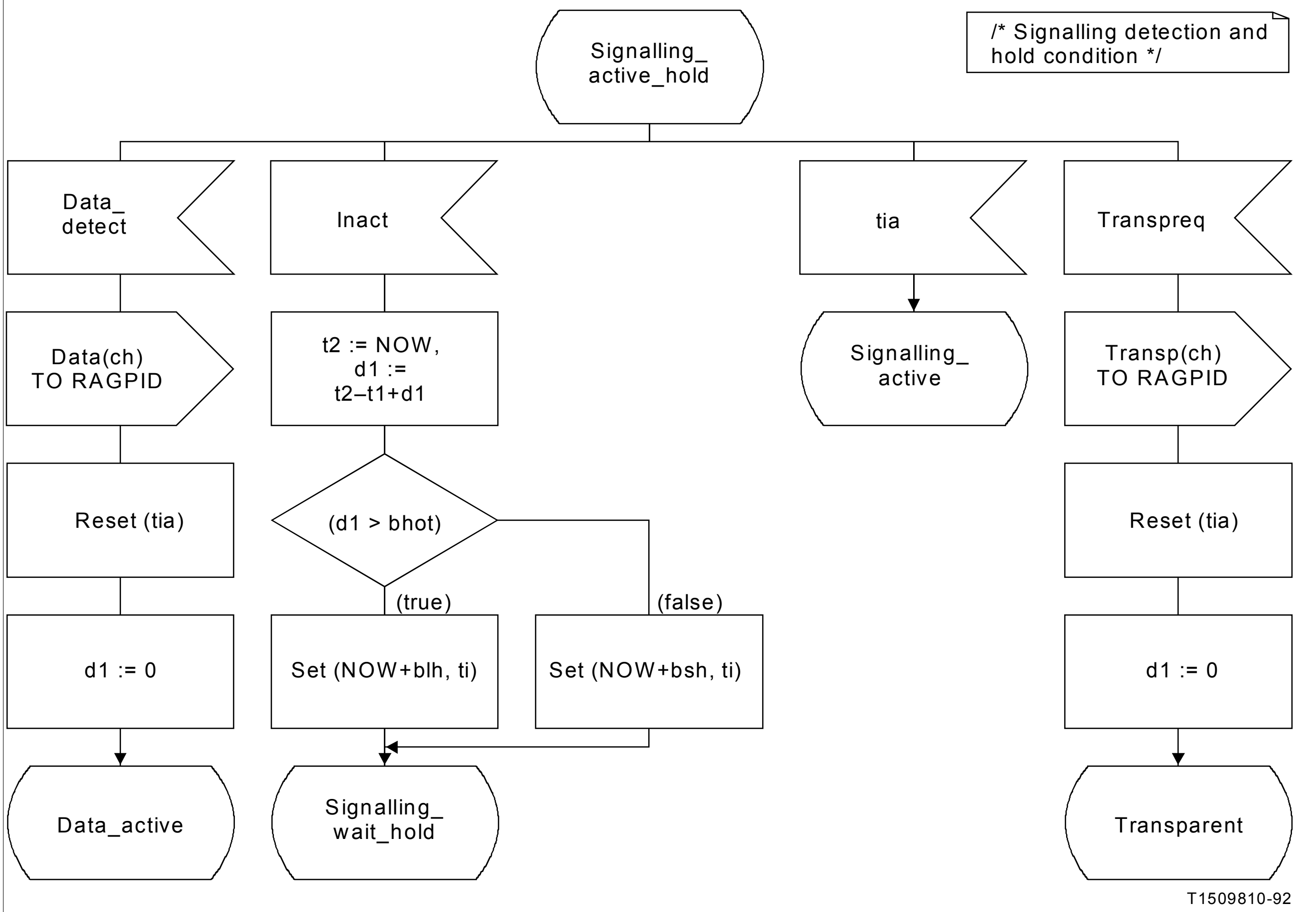

PROCESS HSC
5 (15)
Voice_ active_ hold
Data_ detect
Signal_ detect
Inact
tia
Transpreq
Data(ch) TO RAGPID
d1 := 0
t2 := NOW, d1 := t2–t1+d1
Voice_active
Transp(ch) TO RAGPID
Reset(tia)
Signalling_ active_ hold
(d1 > hot)
Reset(tia)
(true)
(false)
d1 := 0
Set (NOW+lh, ti)
Set (NOW+sh, ti)
d1 := 0
Data_ active
Voice_wait_ hold
Transparent
T1509770-92
/* Voice_active state where Rxdata is disallowed. Hold states are only valid for a second in order to avoid flip-flop possibilities data-voice-data-voice etc. */

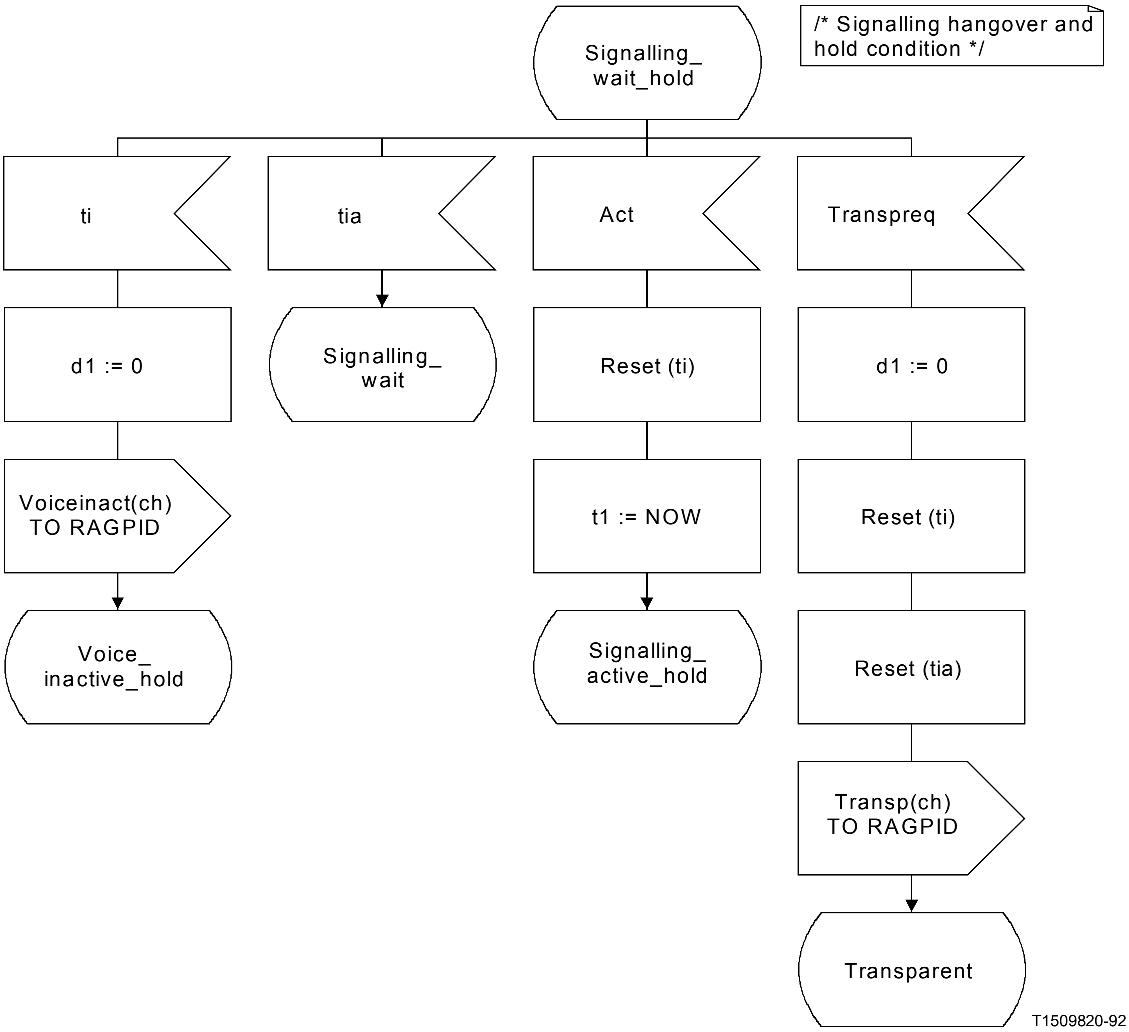

PROCESS HSC
6 (15)
/* Hangover for voice with hold condition */
Voice_wait_ hold
ti
tia
Act
Transpreq
d1 := 0
Voice_wait
Reset (ti)
d1 := 0
Voiceinact(ch) TO RAGPID
t1 := NOW
Reset (ti)
Voice_ inactive_ hold
Voice_ active_ hold
Reset (tia)
Transp(ch) TO RAGPID
Transparent
/* Voice_inactive state with hold condition */
Voice_ inactive_ hold
tia
Act
Transpreq
Voice_ inactive
t1 := NOW
Reset (tia)
Voice(ch) TO RAGPID
Transp(ch) TO RAGPID
Voice_ active_ hold
Transparent
T1509780-92

PROCESS HSC 7 (15)

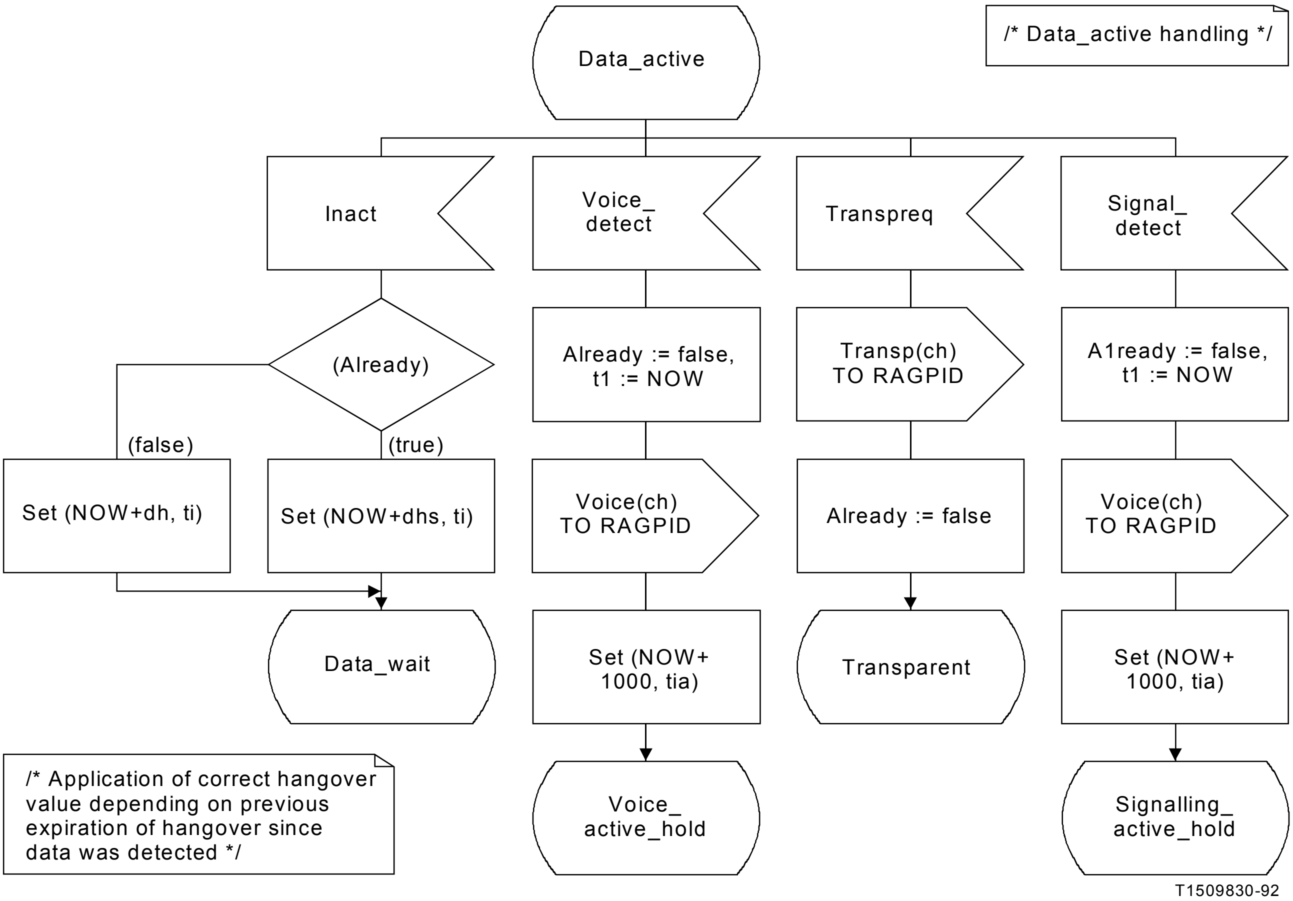


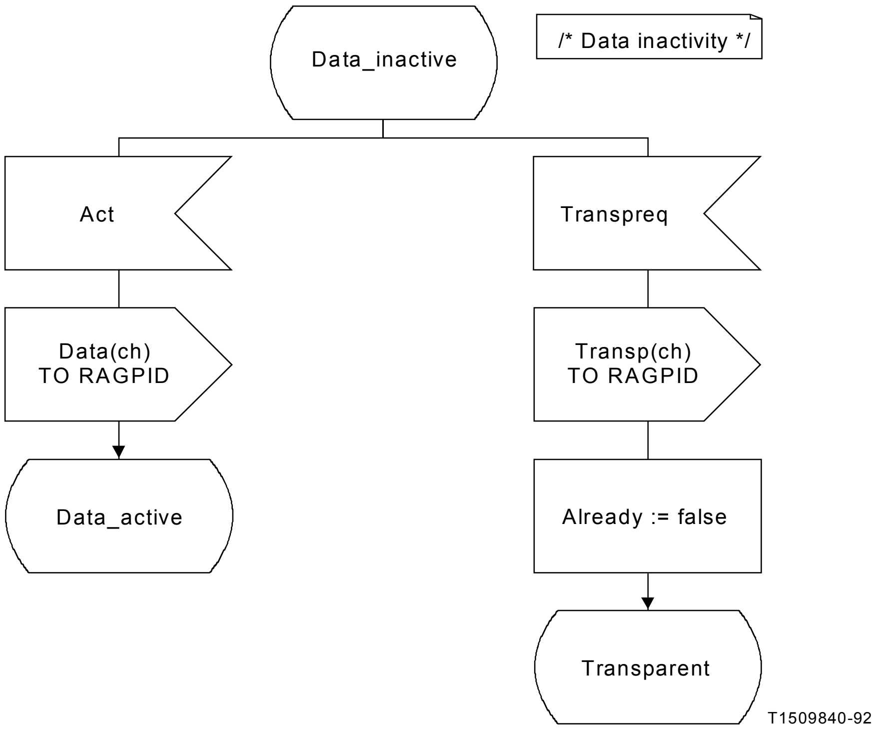

Signalling_
wait
/* Normal extended signalling
hangover handling */
Rxdata
ti
Act
Transpreq
Default_data
d1 := 0
Reset (ti)
d1 := 0
d1 := 0
Voiceinact(ch)
TO RAGPID
t1 := NOW
Reset (ti)
Wait_for_
data
Voice_
inactive
Signalling_
active
Transp(ch)
TO RAGPID
Transparent
T1509800-92

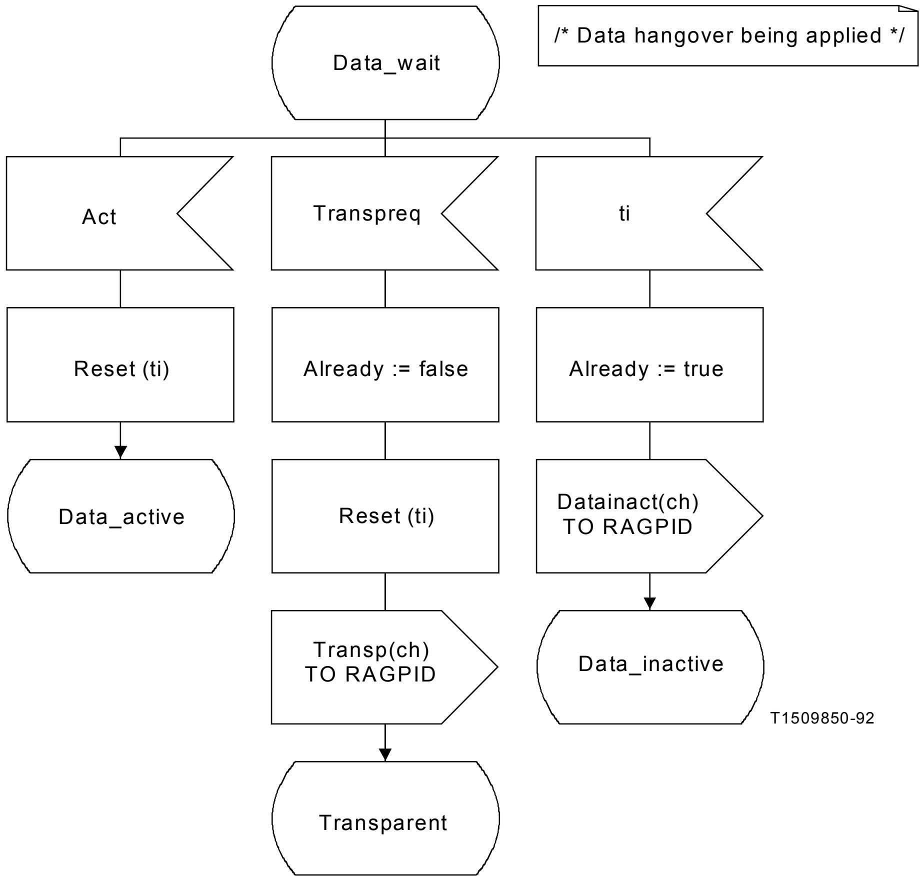

Signalling_
active_hold
/* Signalling detection and
hold condition */
Data_
detect
Inact
tia
Transpreq
Data(ch)
TO RAGPID
t2 := NOW,
d1 :=
t2–t1+d1
Signalling_
active
Transp(ch)
TO RAGPID
Reset (tia)
(d1 > bhot)
Reset (tia)
(true)
(false)
d1 := 0
Set (NOW+blh, ti)
Set (NOW+bsh, ti)
d1 := 0
Data_active
Signalling_
wait_hold
Transparent
T1509810-92

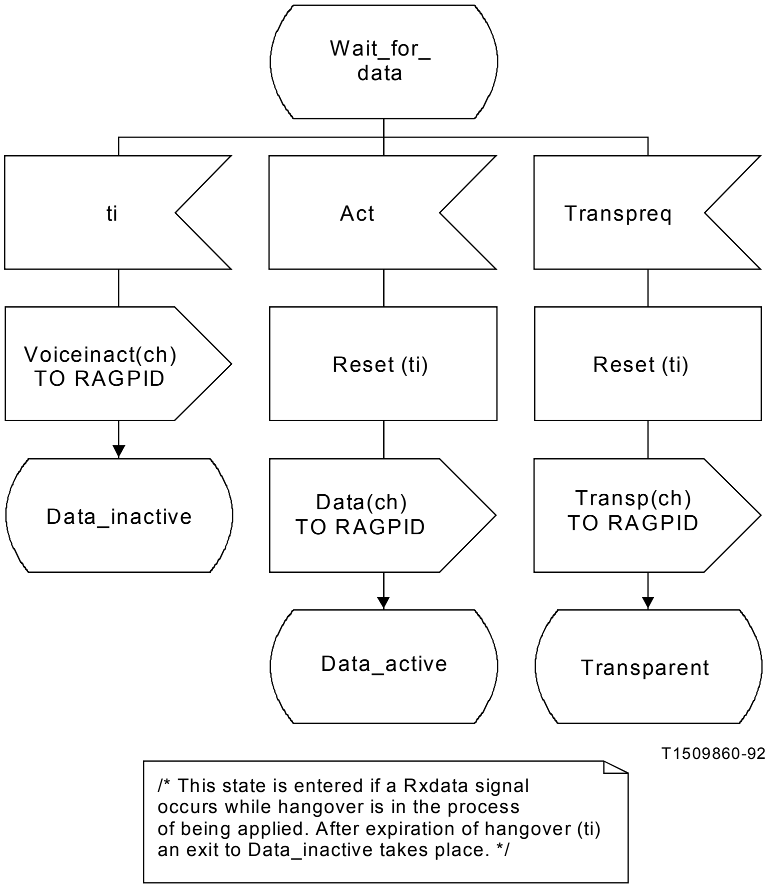


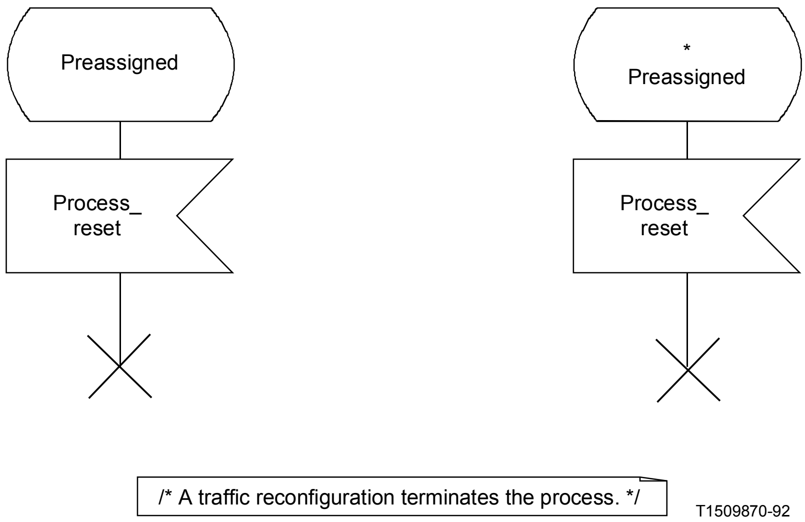
Signalling_ wait_hold
/* Signalling hangover and hold condition */
ti
tia
Act
Transpreq
d1 := 0
Signalling_ wait
Reset (ti)
d1 := 0
Voiceinact(ch) TO RAGPID
t1 := NOW
Reset (ti)
Voice_ inactive_hold
Signalling_ active_hold
Reset (tia)
Transp(ch) TO RAGPID
Transparent
T1509820-92

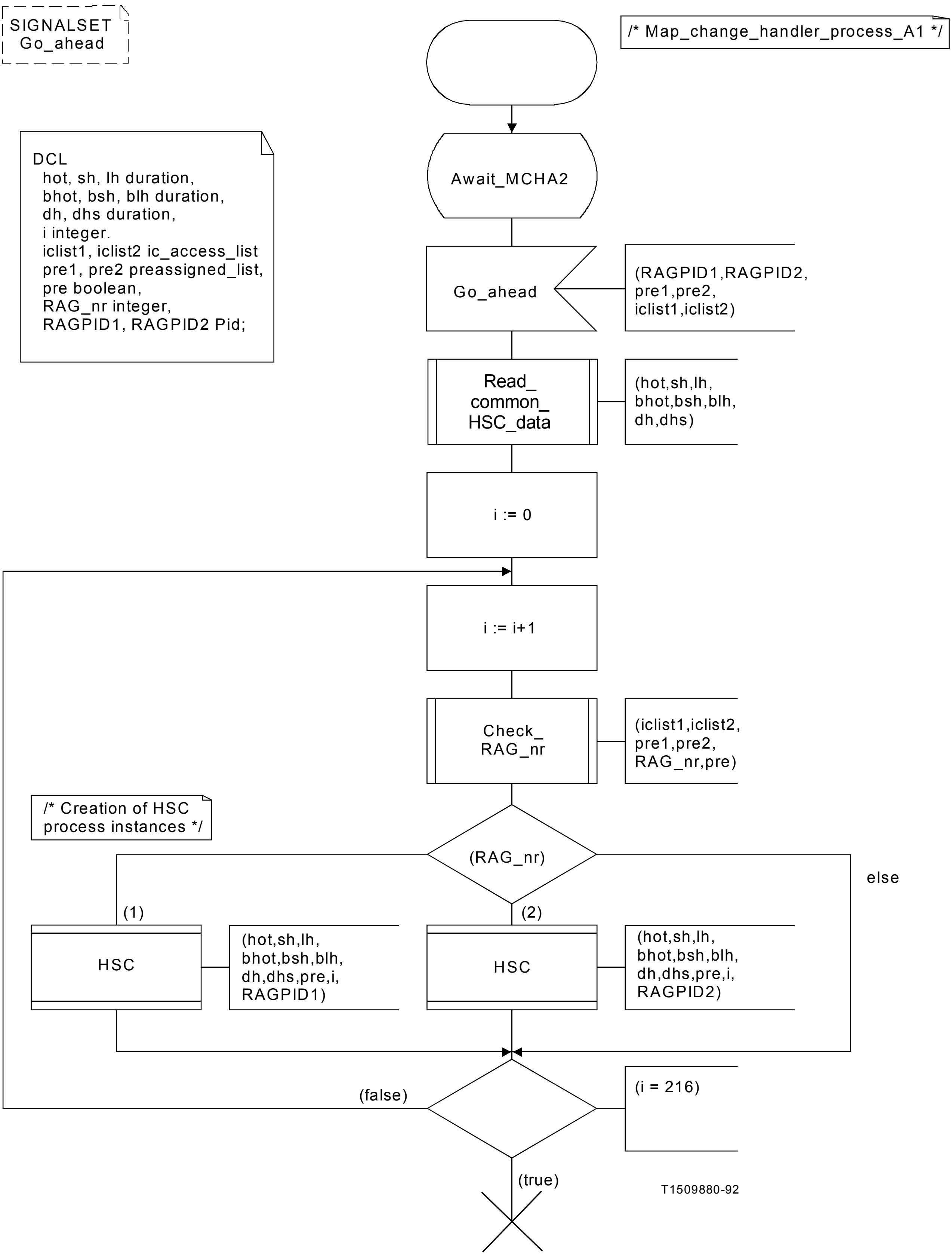

/* Data_active handling */
Data_active
Inact
Voice_
detect
Transpreq
Signal_
detect
(Already)
(false)
(true)
Already := false,
t1 := NOW
Transp(ch)
TO RAGPID
A1ready := false,
t1 := NOW
Set (NOW+dh, ti)
Set (NOW+dhs, ti)
Voice(ch)
TO RAGPID
Already := false
Voice(ch)
TO RAGPID
Data_wait
Set (NOW+
1000, tia)
Transparent
Set (NOW+
1000, tia)
/* Application of correct hangover
value depending on previous
expiration of hangover since
data was detected */
Voice_
active_hold
Signalling_
active_hold
T1509830-92

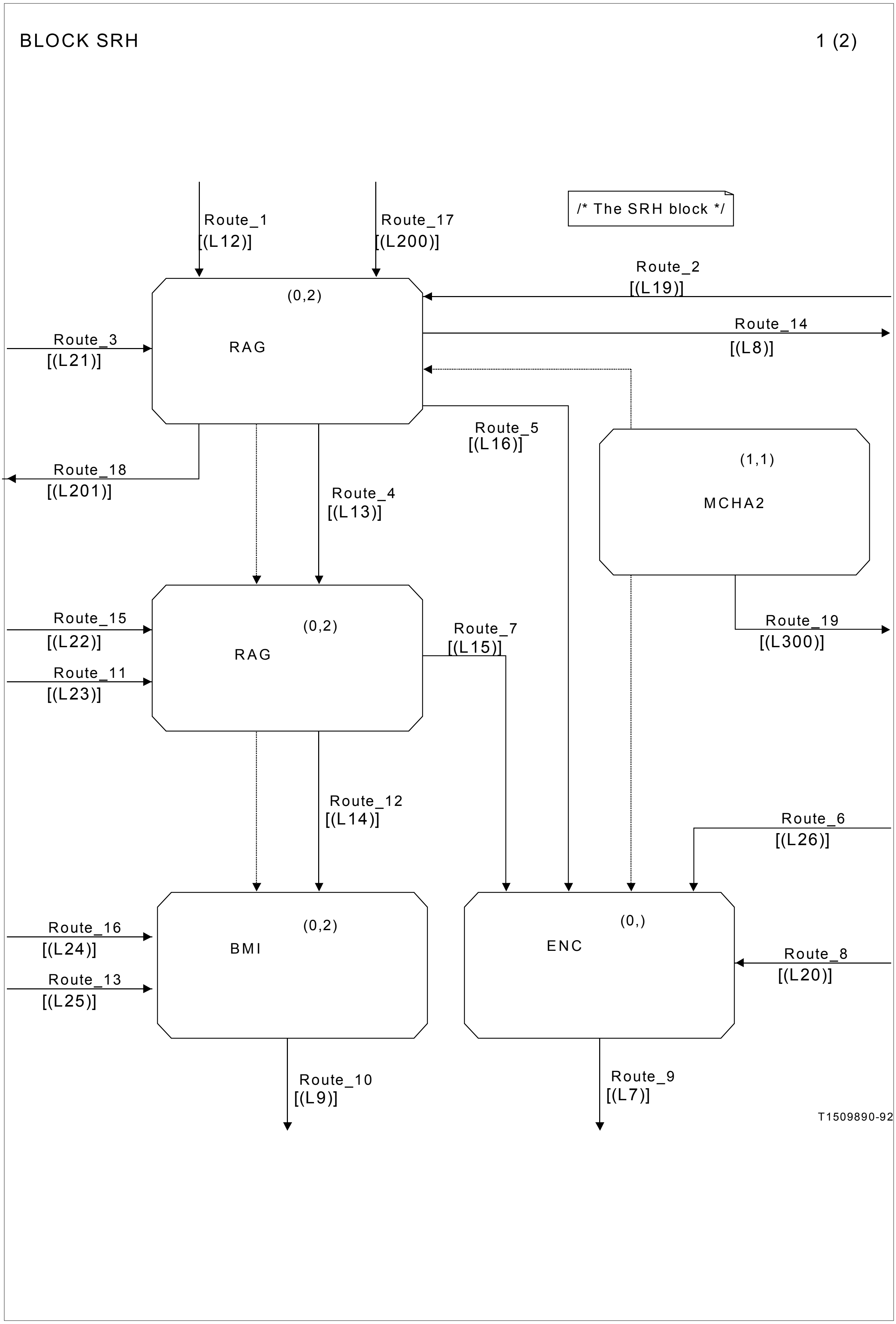

Data_inactive
/* Data inactivity */
Act
Transpreq
Data(ch)
TO RAGPID
Transp(ch)
TO RAGPID
Data_active
Already := false
Transparent
T1509840-92


/* Data hangover being applied */
Data_wait
Act
Transpreq
ti
Reset (ti)
Already := false
Already := true
Data_active
Reset (ti)
Datainact(ch)
TO RAGPID
Transp(ch)
TO RAGPID
Data_inactive
T1509850-92
Transparent


Wait_for_
data
ti
Act
Transpreq
Voiceinact(ch)
TO RAGPID
Reset (ti)
Reset (ti)
Data_inactive
Data(ch)
TO RAGPID
Transp(ch)
TO RAGPID
Data_active
Transparent
T1509860-92
/* This state is entered if a Rxdata signal
occurs while hangover is in the process
of being applied. After expiration of hangover (ti)
an exit to Data_inactive takes place. */

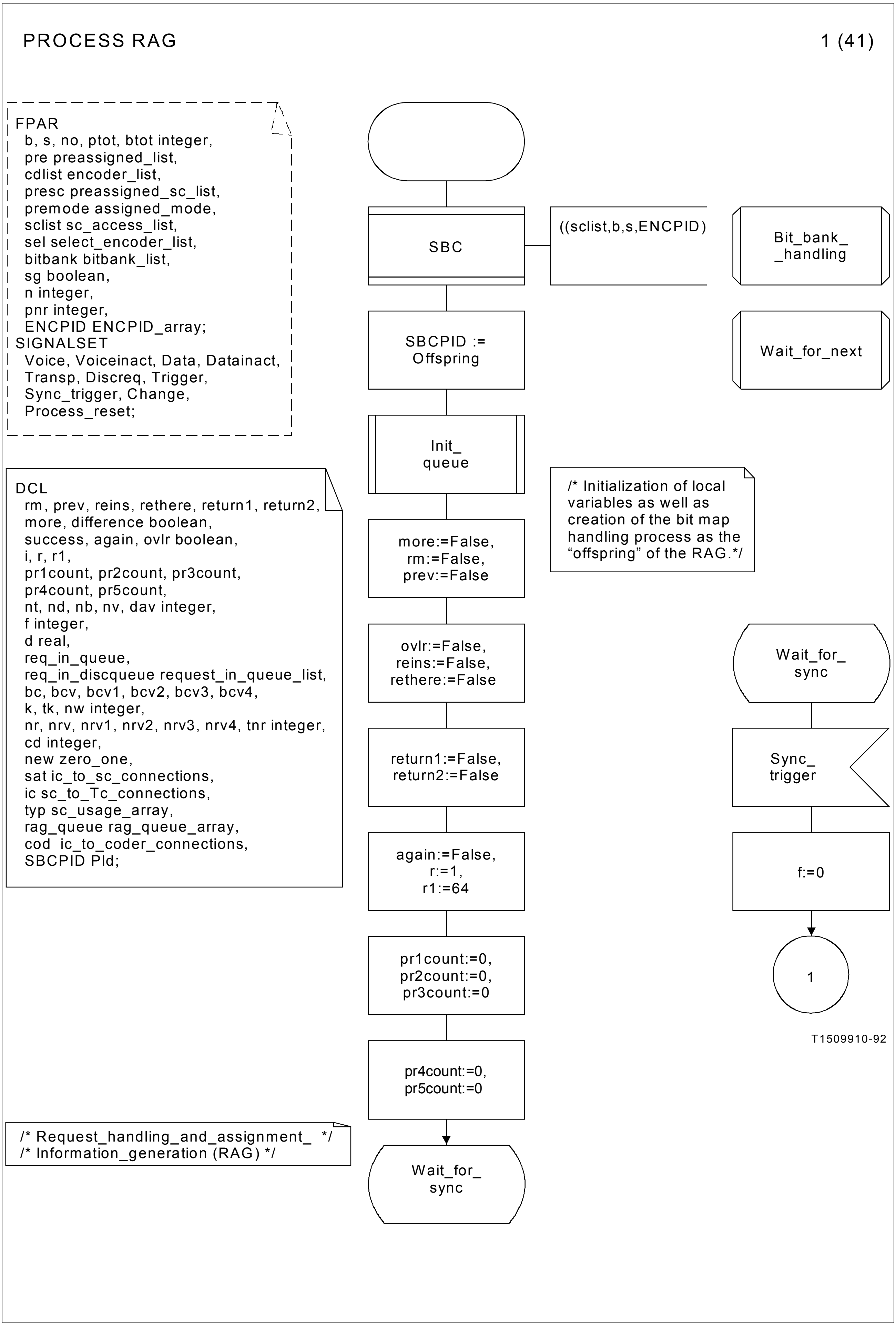

Preassigned
*
Preassigned
Process_
reset
Process_
reset
/* A traffic reconfiguration terminates the process. */
T1509870-92

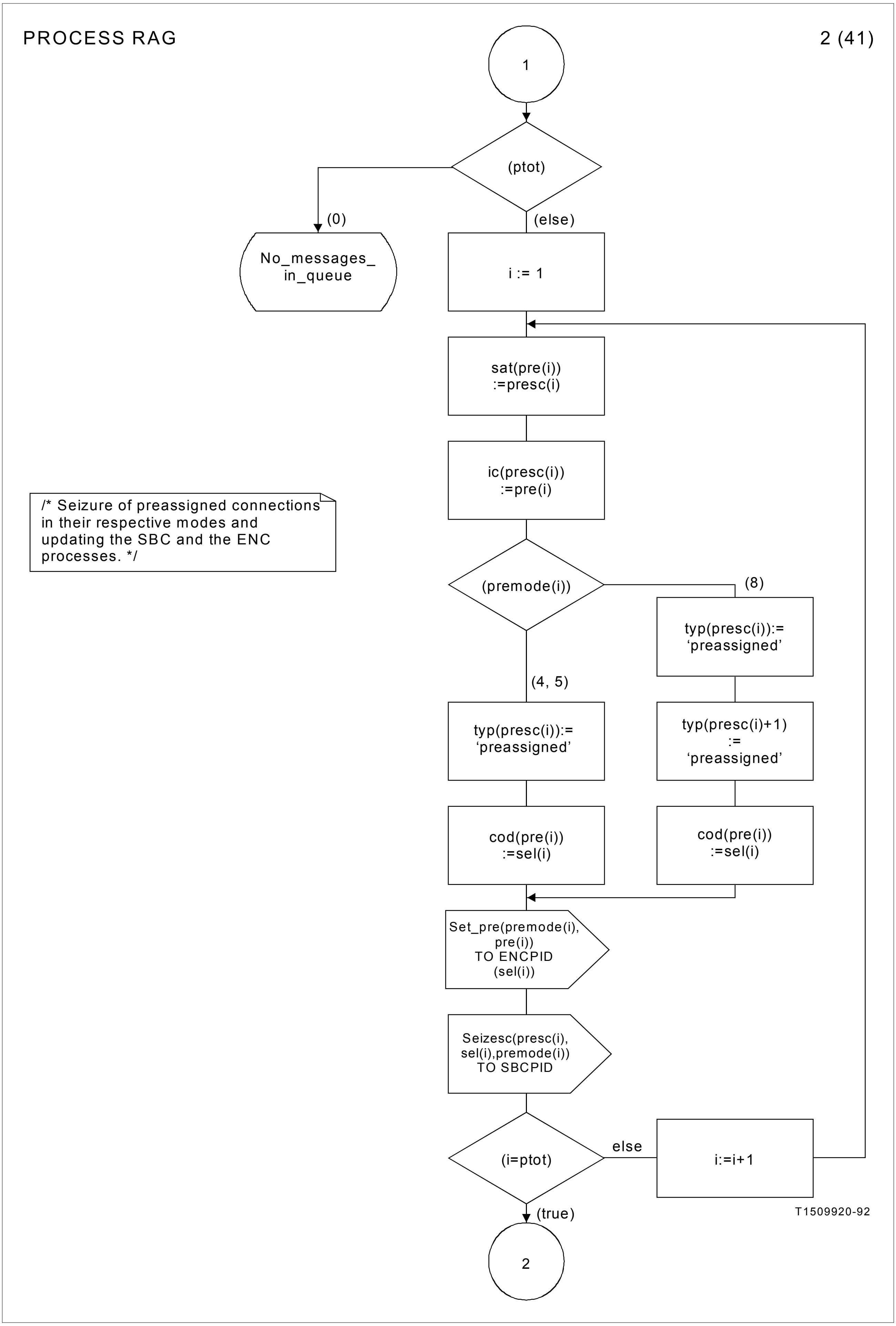

SIGNALSET
Go_ahead
/* Map_change_handler_process_A1 */
DCL
hot, sh, lh duration,
bhot, bsh, blh duration,
dh, dhs duration,
i integer.
iclist1, iclist2 ic_access_list
pre1, pre2 preassigned_list,
pre boolean,
RAG_nr integer,
RAGPID1, RAGPID2 Pid;
Await_MCHA2
Go_ahead
(RAGPID1,RAGPID2,
pre1,pre2,
iclist1,iclist2)
Read_
common_
HSC_data
(hot,sh,lh,
bhot,bsh,blh,
dh,dhs)
i := 0
i := i+1
Check_
RAG_nr
(iclist1,iclist2,
pre1,pre2,
RAG_nr,pre)
/* Creation of HSC
process instances */
(RAG_nr)
else
(1)
(2)
HSC
(hot,sh,lh,
bhot,bsh,blh,
dh,dhs,pre,i,
RAGPID1)
HSC
(hot,sh,lh,
bhot,bsh,blh,
dh,dhs,pre,i,
RAGPID2)
(false)
(i = 216)
(true)
T1509880-92

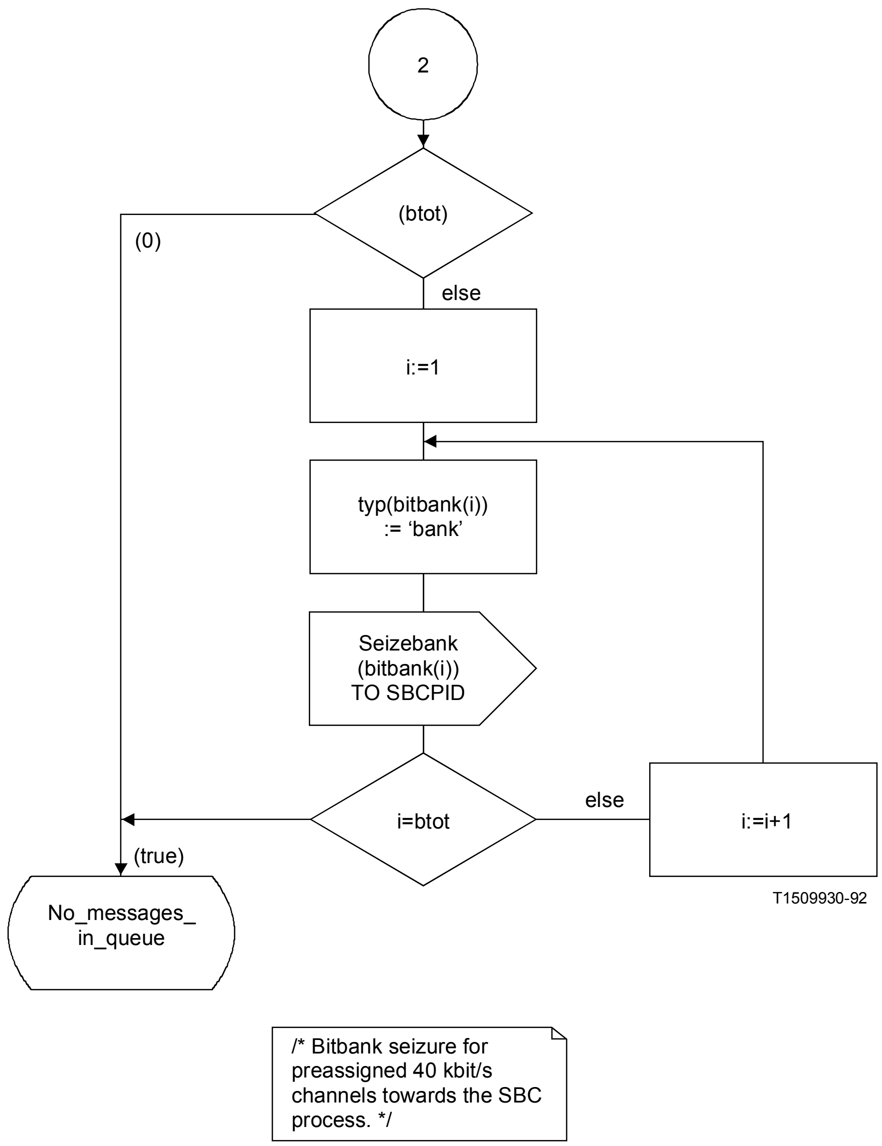

BLOCK SRH
1 (2)
/* The SRH block */
Route_1
[(L12)]
Route_17
[(L200)]
Route_2
[(L19)]
Route_14
[(L8)]
Route_3
[(L21)]
(0,2)
RAG
Route_5
[(L16)]
Route_18
[(L201)]
Route_4
[(L13)]
(1,1)
MCHA2
Route_15
[(L22)]
(0,2)
RAG
Route_11
[(L23)]
Route_7
[(L15)]
Route_19
[(L300)]
Route_12
[(L14)]
Route_6
[(L26)]
Route_16
[(L24)]
(0,2)
BMI
Route_13
[(L25)]
(0,)
ENC
Route_8
[(L20)]
Route_10
[(L9)]
Route_9
[(L7)]
T1509890-92

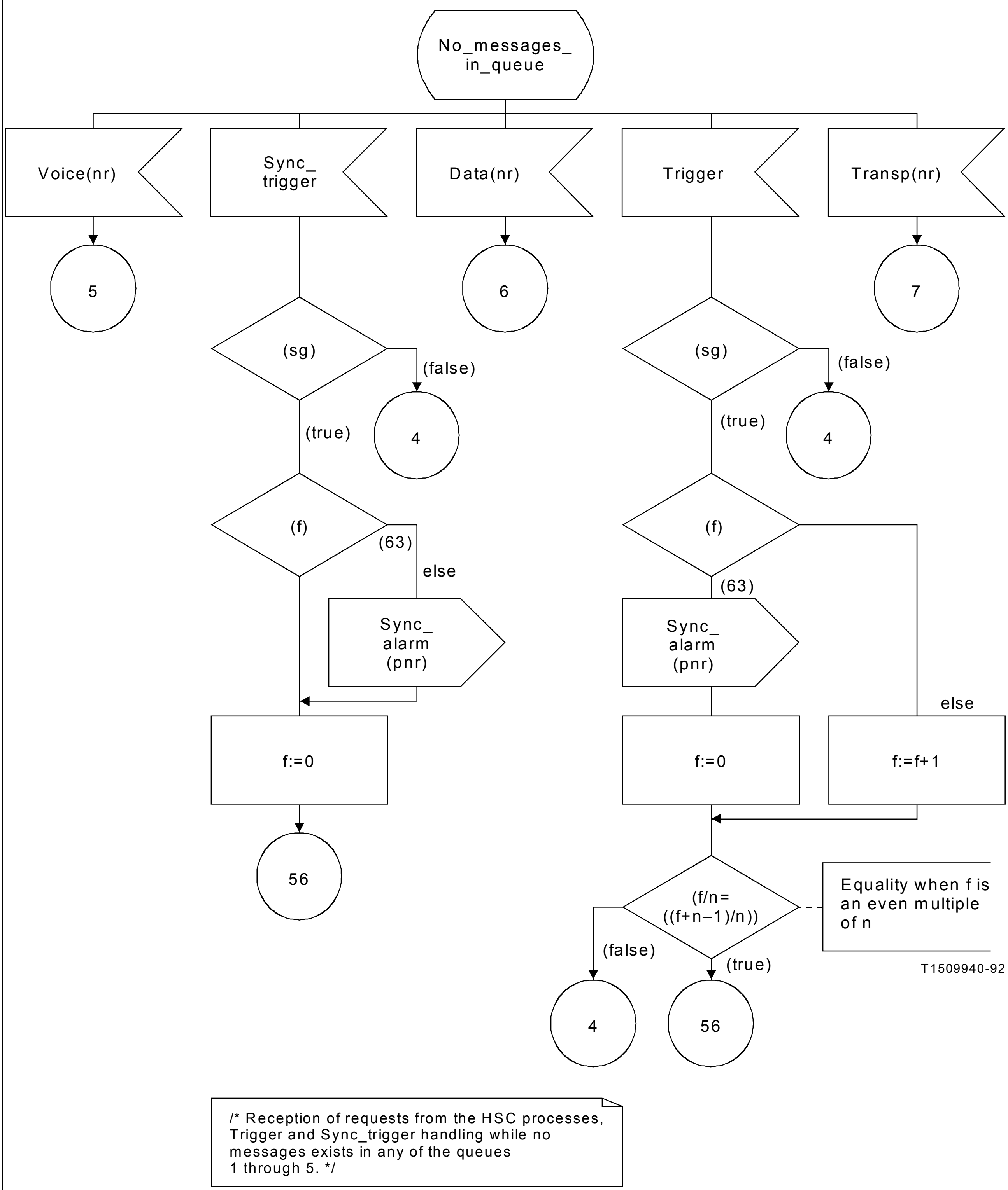
```
/* Signal definitions */
SIGNAL
   Reinsert(Integer), Remove(Integer),
   Seizesc(Integer,Integer,Integer),
   Release(Integer), Releasesc(Integer),
   Seizebank(Integer),
   SC_bitmap(bit_mode_matrix),
   Mode_map(Integer),
   Assign_enc(Integer,Integer,Call_Type),
   Release_enc,
   Set_pre(Integer,Integer);

/* Signallist definitions */
SIGNALLIST L13 = Assign, Reinsert, Remove, Seizesc,
          Release, Releasesc, Seizebank;
SIGNALLIST L14 = SC_bitmap;
SIGNALLIST L15 = Mode_map;
SIGNALLIST L16 = Assign_enc, Release_enc, Set_pre;

connect c12 and Route_1;
connect c19 and Route_2;
connect c21 and Route_3;
connect c26 and Route_6;
connect c20 and Route_8;
connect c7 and Route_9;
connect c9 and Route_10;
connect c23 and Route_11;
connect c25 and Route_13;
connect c8 and Route_14;
connect c22 and Route_15;
connect c24 and Route_16;
connect c200 and Route_17;
connect c201 and Route_18;
connect c300 and Route_19;
```

T1509900-92


PROCESS RAG
1 (41)
FPAR
b, s, no, ptot, btot integer,
pre preassigned_list,
cdlist encoder_list,
presc preassigned_sc_list,
premode assigned_mode,
sclist sc_access_list,
sel select_encoder_list,
bitbank bitbank_list,
sg boolean,
n integer,
pnr integer,
ENCPID ENCPID_array;
SIGNALSET
Voice, Voiceinact, Data, Datainact,
Transp, Discreq, Trigger,
Sync_trigger, Change,
Process_reset;
DCL
rm, prev, reins, rethere, return1, return2,
more, difference boolean,
success, again, ovlr boolean,
i, r, r1,
pr1count, pr2count, pr3count,
pr4count, pr5count,
nt, nd, nb, nv, dav integer,
f integer,
d real,
req_in_queue,
req_in_discqueue request_in_queue_list,
bc, bcv, bcv1, bcv2, bcv3, bcv4,
k, tk, nw integer,
nr, nrv, nrv1, nrv2, nrv3, nrv4, tnr integer,
cd integer,
new zero_one,
sat ic_to_sc_connections,
ic sc_to_Tc_connections,
typ sc_usage_array,
rag_queue rag_queue_array,
cod ic_to_coder_connections,
SBCPID PId;
/* Request_handling_and_assignment_ */
/* Information_generation (RAG) */
SBC
((sclist,b,s,ENCPID)
Bit_bank_
_handling
SBCPID :=
Offspring
Wait_for_next
Init_
queue
/* Initialization of local
variables as well as
creation of the bit map
handling process as the
“offspring” of the RAG.*/
more:=False,
rm:=False,
prev:=False
ovlr:=False,
reins:=False,
rethere:=False
return1:=False,
return2:=False
again:=False,
r:=1,
r1:=64
pr1count:=0,
pr2count:=0,
pr3count:=0
pr4count:=0,
pr5count:=0
Wait_for_
sync
Wait_for_
sync
Sync_
trigger
f:=0
1
T1509910-92

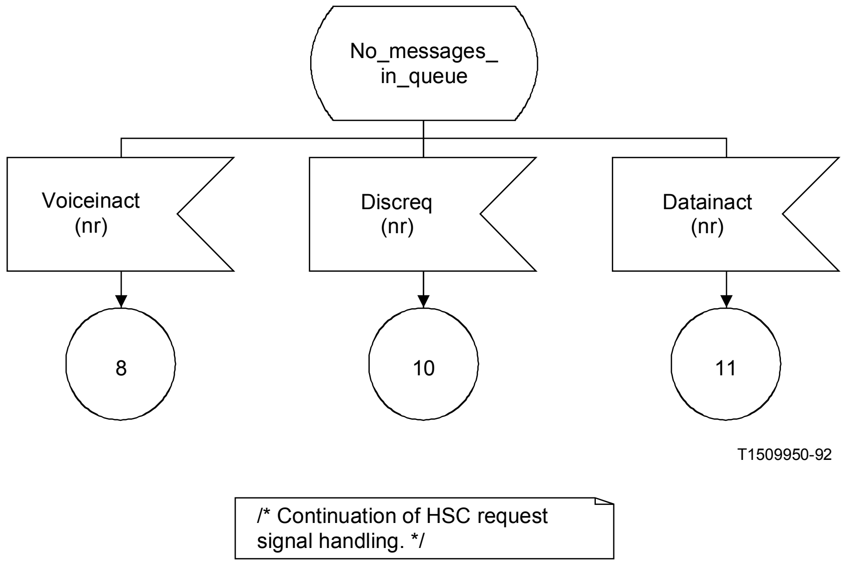

PROCESS RAG
2 (41)
1
(ptot)
(0)
(else)
No_messages_
in_queue
i := 1
sat(pre(i))
:=presc(i)
ic(presc(i))
:=pre(i)
/* Seizure of preassigned connections
in their respective modes and
updating the SBC and the ENC
processes. */
(premode(i))
(8)
typ(presc(i)):=
'preassigned'
(4, 5)
typ(presc(i)):=
'preassigned'
typ(presc(i)+1)
:=
'preassigned'
cod(pre(i))
:=sel(i)
cod(pre(i))
:=sel(i)
Set_pre(premode(i),
pre(i))
TO ENCPID
(sel(i))
Seizesc(presc(i),
sel(i),premode(i))
TO SBCPID
(i=ptot)
else
i:=i+1
(true)
2
T1509920-92

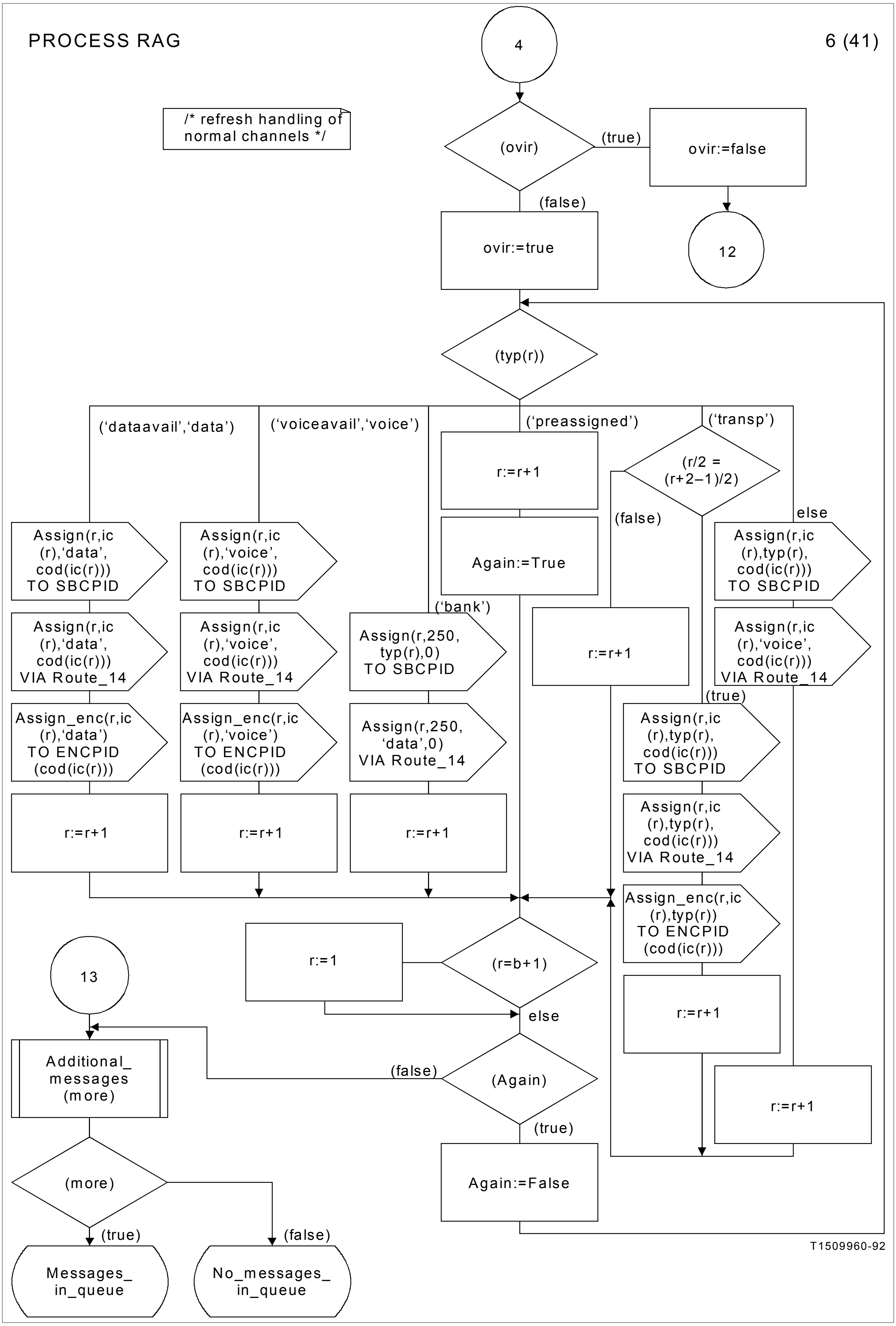

2
(btot)
(0)
else
i:=1
typ(bitbank(i))
:= ‘bank’
Seizebank
(bitbank(i))
TO SBCPID
i=btot
else
i:=i+1
(true)
No_messages_
in_queue
T1509930-92
/* Bitbank seizure for
preassigned 40 kbit/s
channels towards the SBC
process. */



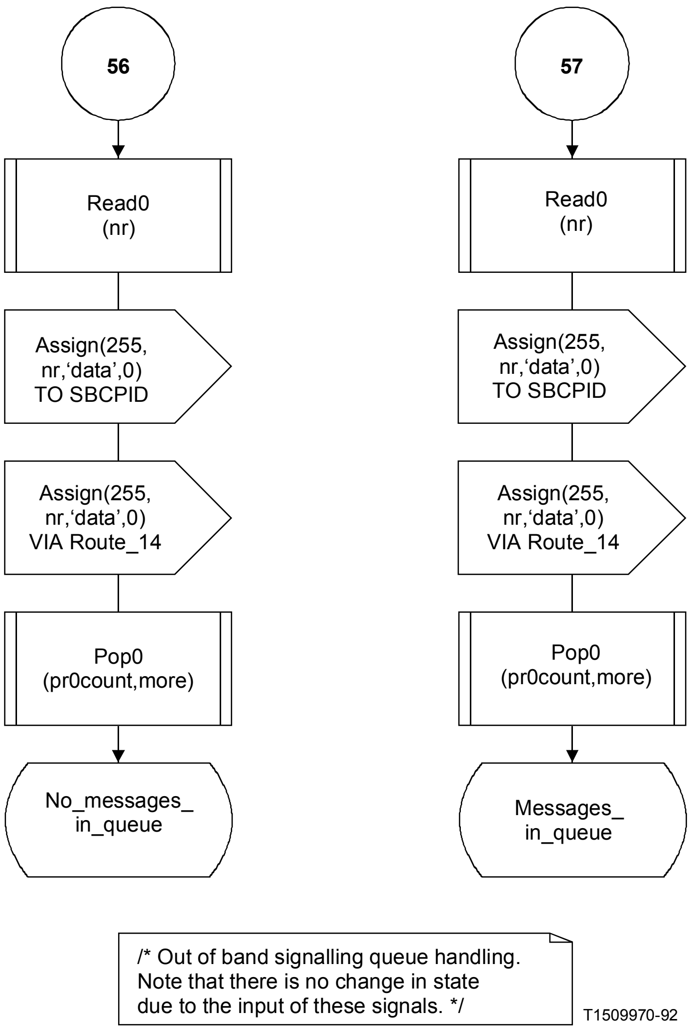
No_messages_ in_queue
Voice(nr)
Sync_ trigger
Data(nr)
Trigger
Transp(nr)
5
6
7
(sg)
(false)
(true)
4
(sg)
(false)
(true)
4
(f)
(63)
else
Sync_ alarm (pnr)
(f)
(63)
Sync_ alarm (pnr)
else
f:=0
f:=0
f:=f+1
56
(f/n= ((f+n–1)/n))
Equality when f is an even multiple of n
(false)
(true)
4
56
/* Reception of requests from the HSC processes, Trigger and Sync_trigger handling while no messages exists in any of the queues 1 through 5. */

T1509940-92


No_messages_
in_queue
Voiceinact
(nr)
Discreq
(nr)
Datainact
(nr)
8
10
11
T1509950-92
/* Continuation of HSC request
signal handling. */

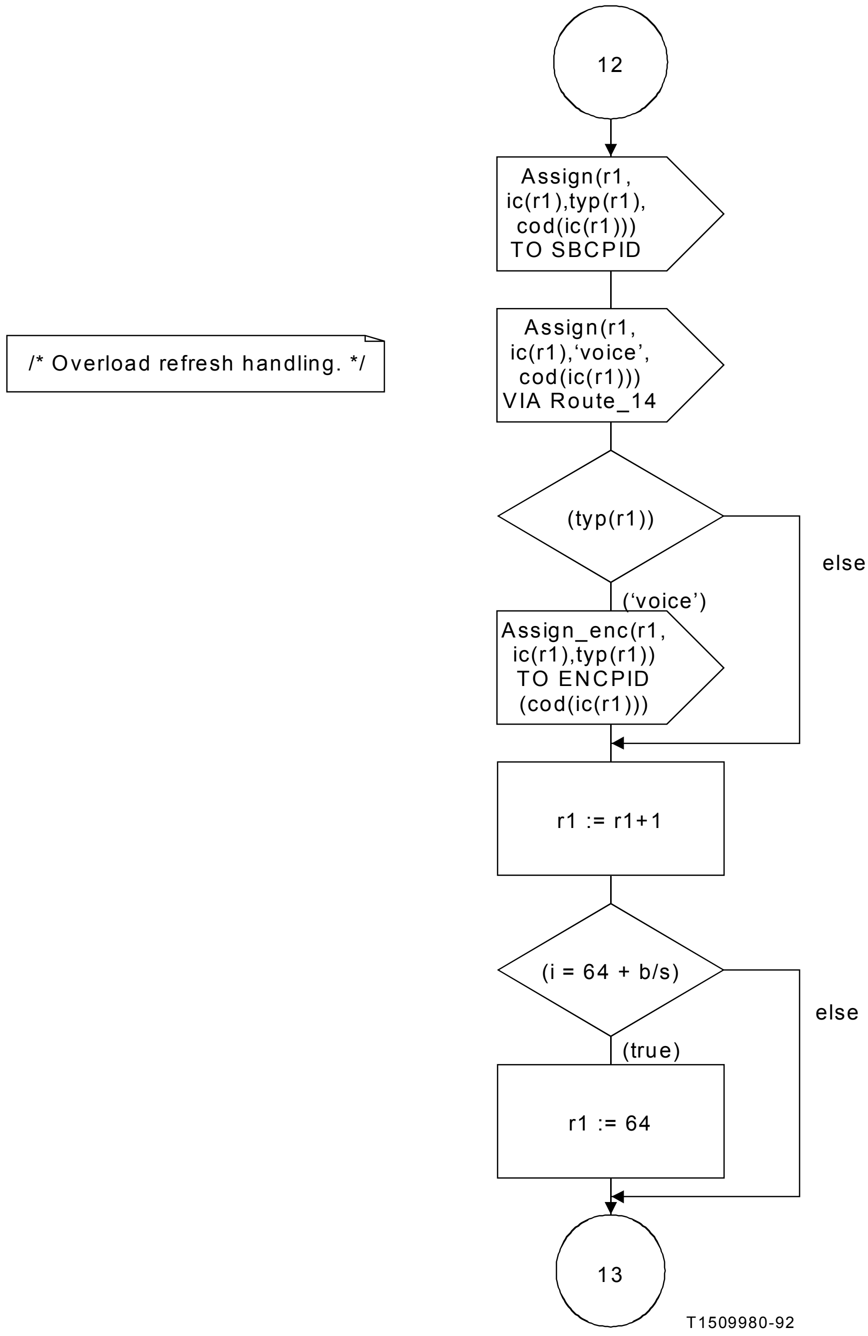


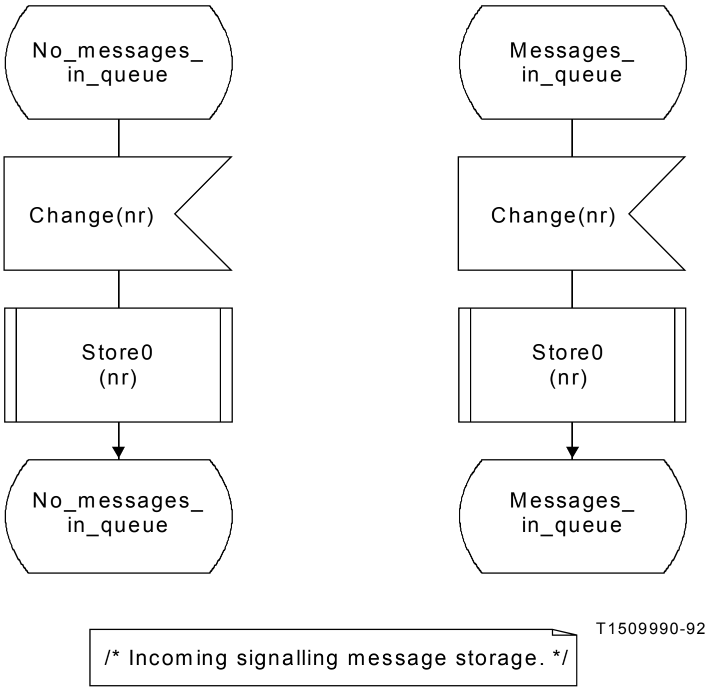
PROCESS RAG
6 (41)
4
/* refresh handling of normal channels */
(ovir)
(true)
ovir:=false
12
(false)
ovir:=true
(typ(r))
('dataavail','data')
('voiceavail','voice')
('preassigned')
('transp')
r:=r+1
(r/2 = (r+2–1)/2)
(false)
else
Assign(r,ic (r),'data', cod(ic(r))) TO SBCPID
Assign(r,ic (r),'voice', cod(ic(r))) TO SBCPID
Again:=True
Assign(r,ic (r),typ(r), cod(ic(r))) TO SBCPID
('bank')
Assign(r,ic (r),'data', cod(ic(r))) VIA Route_14
Assign(r,ic (r),'voice', cod(ic(r))) VIA Route_14
Assign(r,250, typ(r),0) TO SBCPID
r:=r+1
Assign(r,ic (r),'voice', cod(ic(r))) VIA Route_14
(true)
Assign_enc(r,ic (r),'data') TO ENCPID (cod(ic(r)))
Assign_enc(r,ic (r),'voice') TO ENCPID (cod(ic(r)))
Assign(r,250, 'data',0) VIA Route_14
Assign(r,ic (r),typ(r), cod(ic(r))) TO SBCPID
r:=r+1
r:=r+1
r:=r+1
Assign(r,ic (r),typ(r), cod(ic(r))) VIA Route_14
Assign_enc(r,ic (r),typ(r)) TO ENCPID (cod(ic(r)))
13
r:=1
(r=b+1)
else
r:=r+1
(false)
(Again)
r:=r+1
Additional_ messages (more)
(true)
(more)
Again:=False
(true)
(false)
T1509960-92
Messages_ in_queue
No_messages_ in_queue

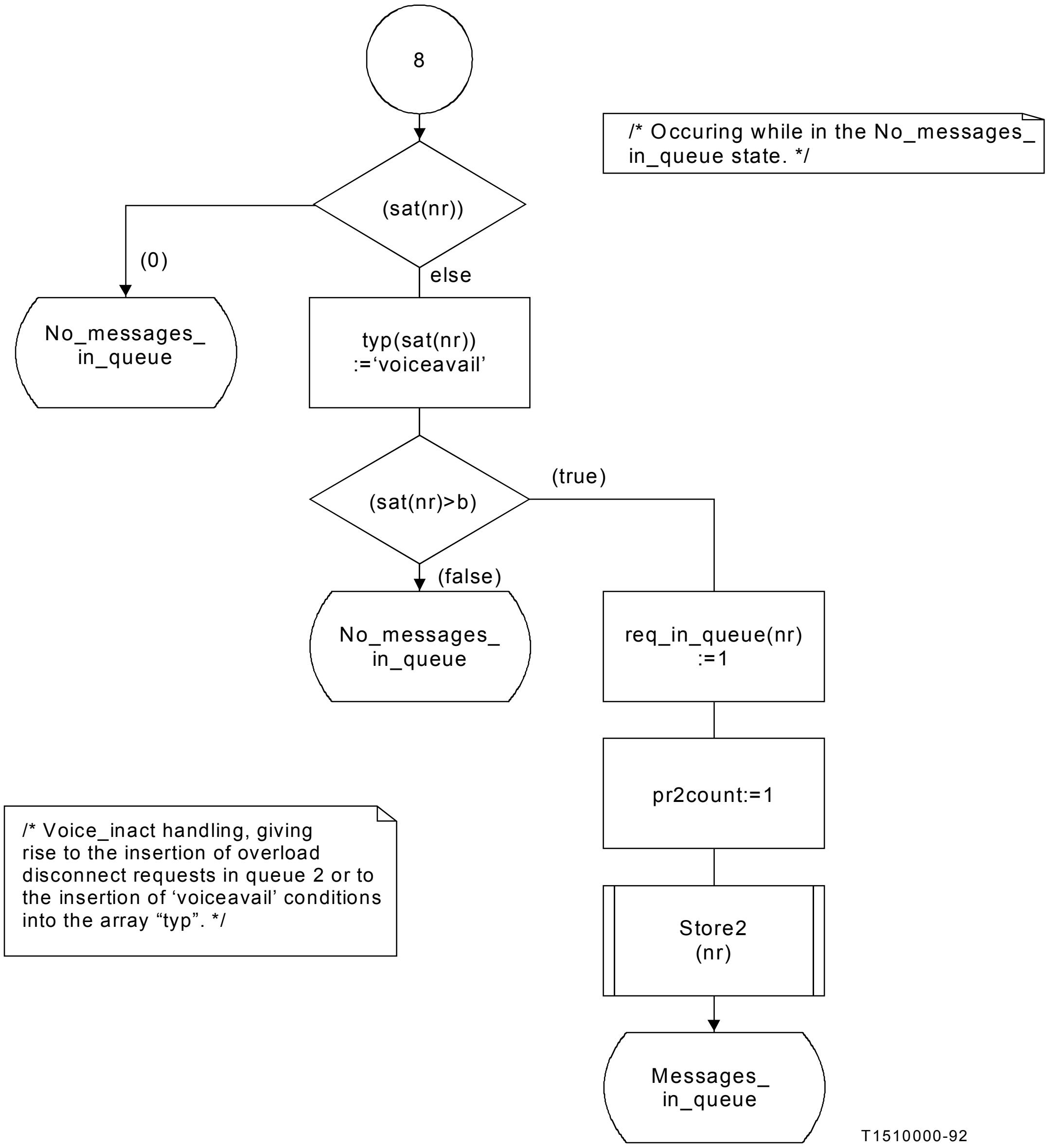


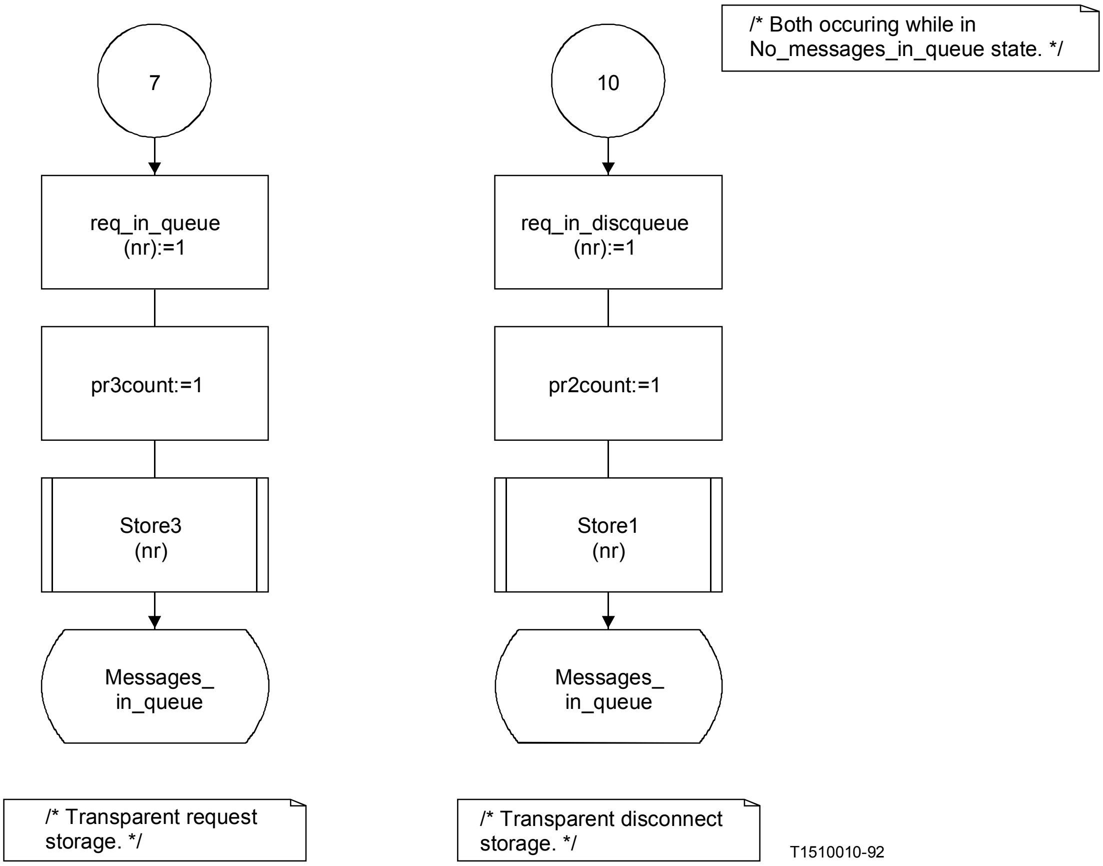
56
Read0
(nr)
Assign(255,
nr,'data',0)
TO SBCPID
Assign(255,
nr,'data',0)
VIA Route_14
Pop0
(pr0count,more)
No_messages_
in_queue
57
Read0
(nr)
Assign(255,
nr,'data',0)
TO SBCPID
Assign(255,
nr,'data',0)
VIA Route_14
Pop0
(pr0count,more)
Messages_
in_queue
/* Out of band signalling queue handling.
Note that there is no change in state
due to the input of these signals. */

T1509970-92

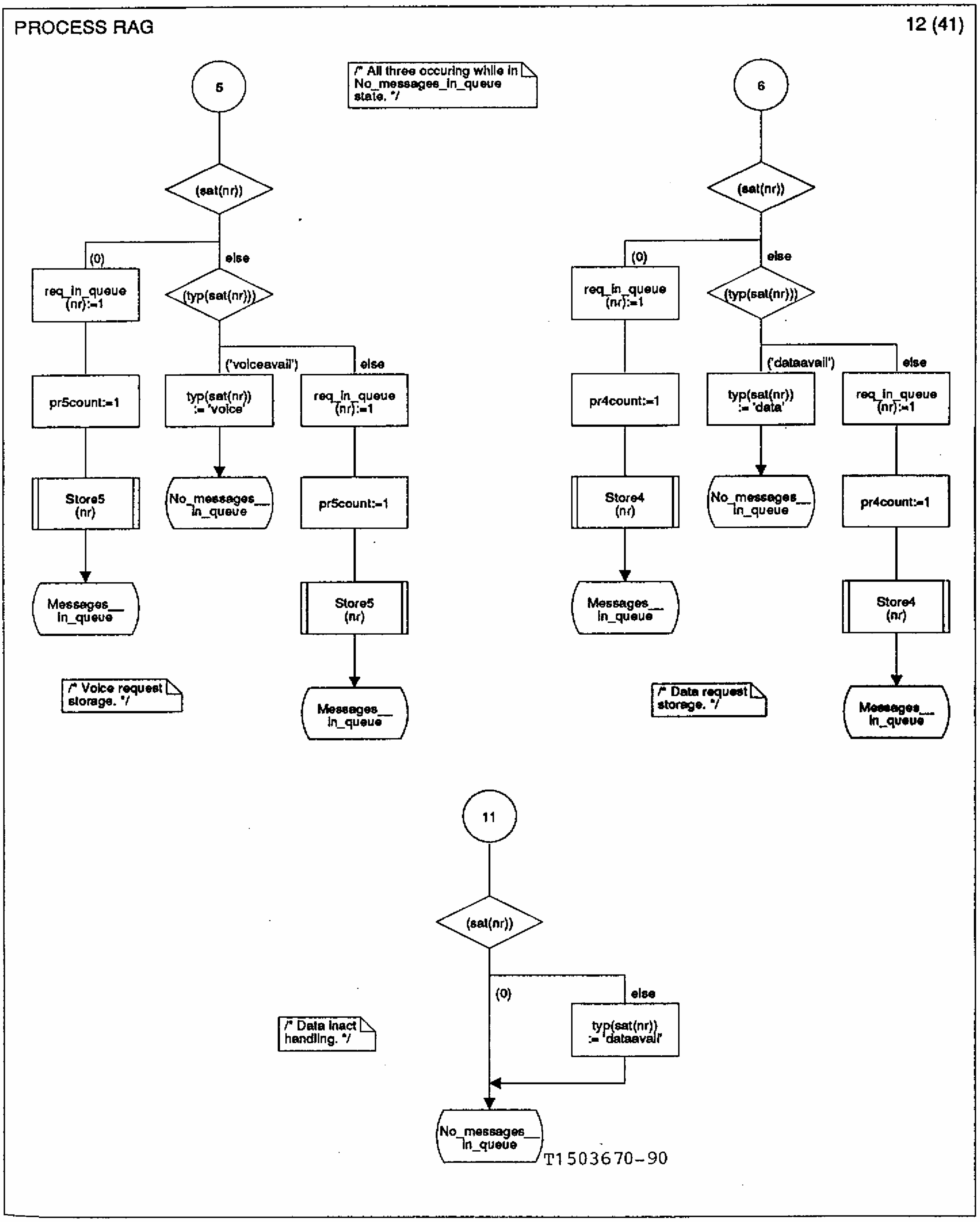

12
Assign(r1,
ic(r1),typ(r1),
cod(ic(r1)))
TO SBCPID
/* Overload refresh handling. */
Assign(r1,
ic(r1),'voice',
cod(ic(r1)))
VIA Route_14
(typ(r1))
else
('voice')
Assign_enc(r1,
ic(r1),typ(r1))
TO ENCPID
(cod(ic(r1)))
r1 := r1+1
(i = 64 + b/s)
else
(true)
r1 := 64
13
T1509980-92

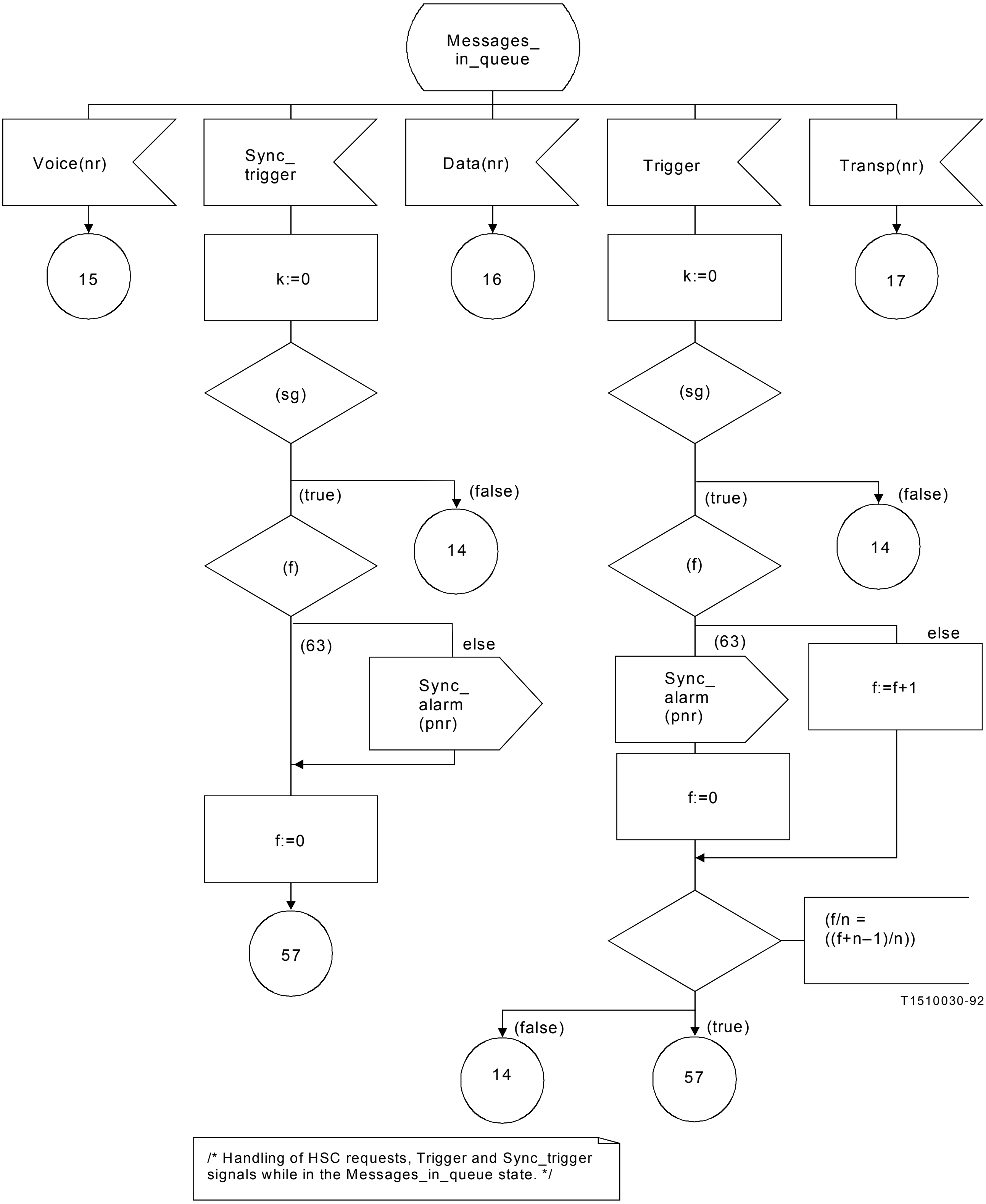

No_messages_
in_queue
Change(nr)
Store0
(nr)
No_messages_
in_queue
Messages_
in_queue
Change(nr)
Store0
(nr)
Messages_
in_queue
T1509990-92
/* Incoming signalling message storage. */

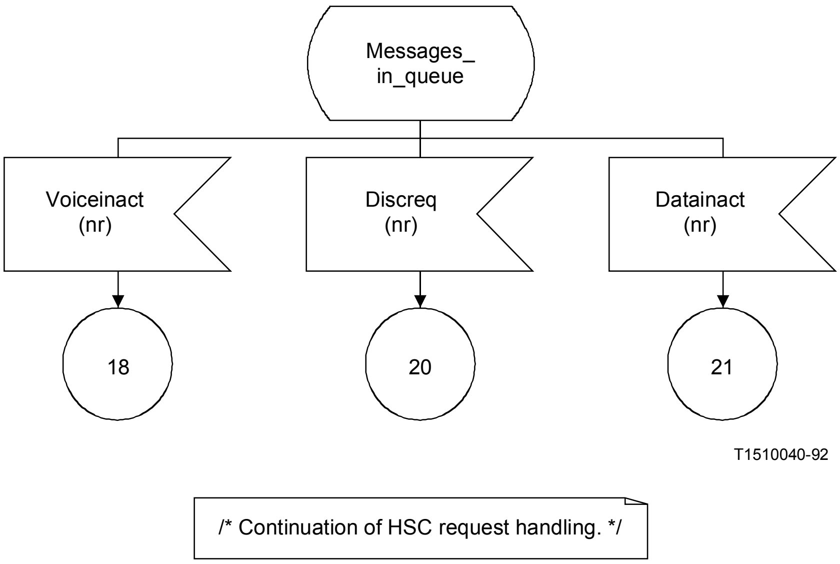

8
/* Occuring while in the No_messages_ in_queue state. */
(sat(nr))
(0)
else
No_messages_ in_queue
typ(sat(nr)) :='voiceavail'
(sat(nr)>b)
(true)
(false)
No_messages_ in_queue
req_in_queue(nr) :=1
pr2count:=1
/* Voice_inact handling, giving rise to the insertion of overload disconnect requests in queue 2 or to the insertion of 'voiceavail' conditions into the array "typ". */
Store2 (nr)
Messages_ in_queue
T1510000-92

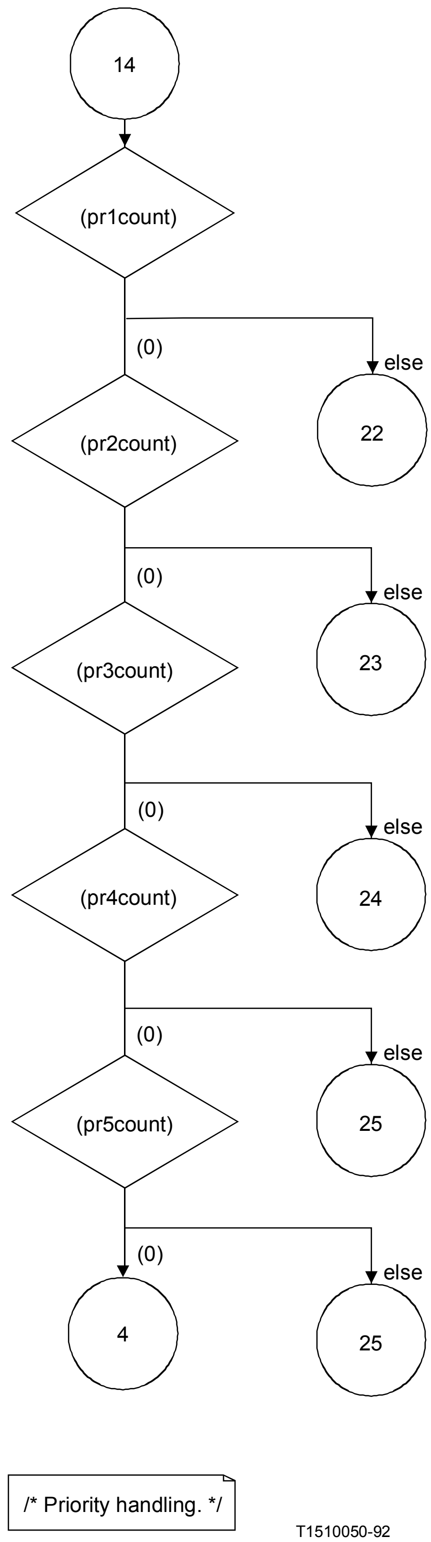


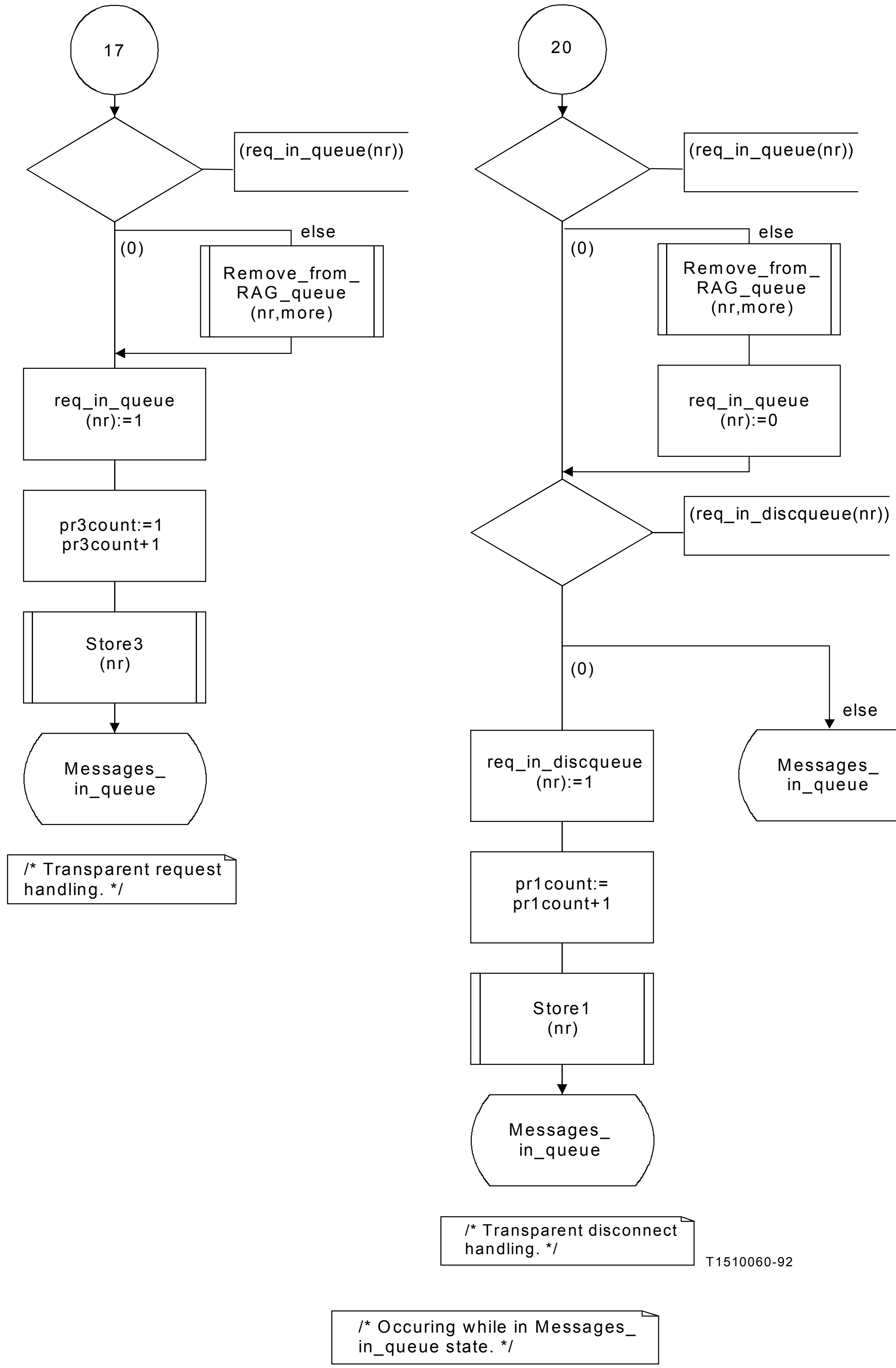
/* Both occuring while in No_messages_in_queue state. */
7
10
req_in_queue (nr):=1
req_in_discqueue (nr):=1
pr3count:=1
pr2count:=1
Store3 (nr)
Store1 (nr)
Messages_ in_queue
Messages_ in_queue
/* Transparent request storage. */
/* Transparent disconnect storage. */
T1510010-92

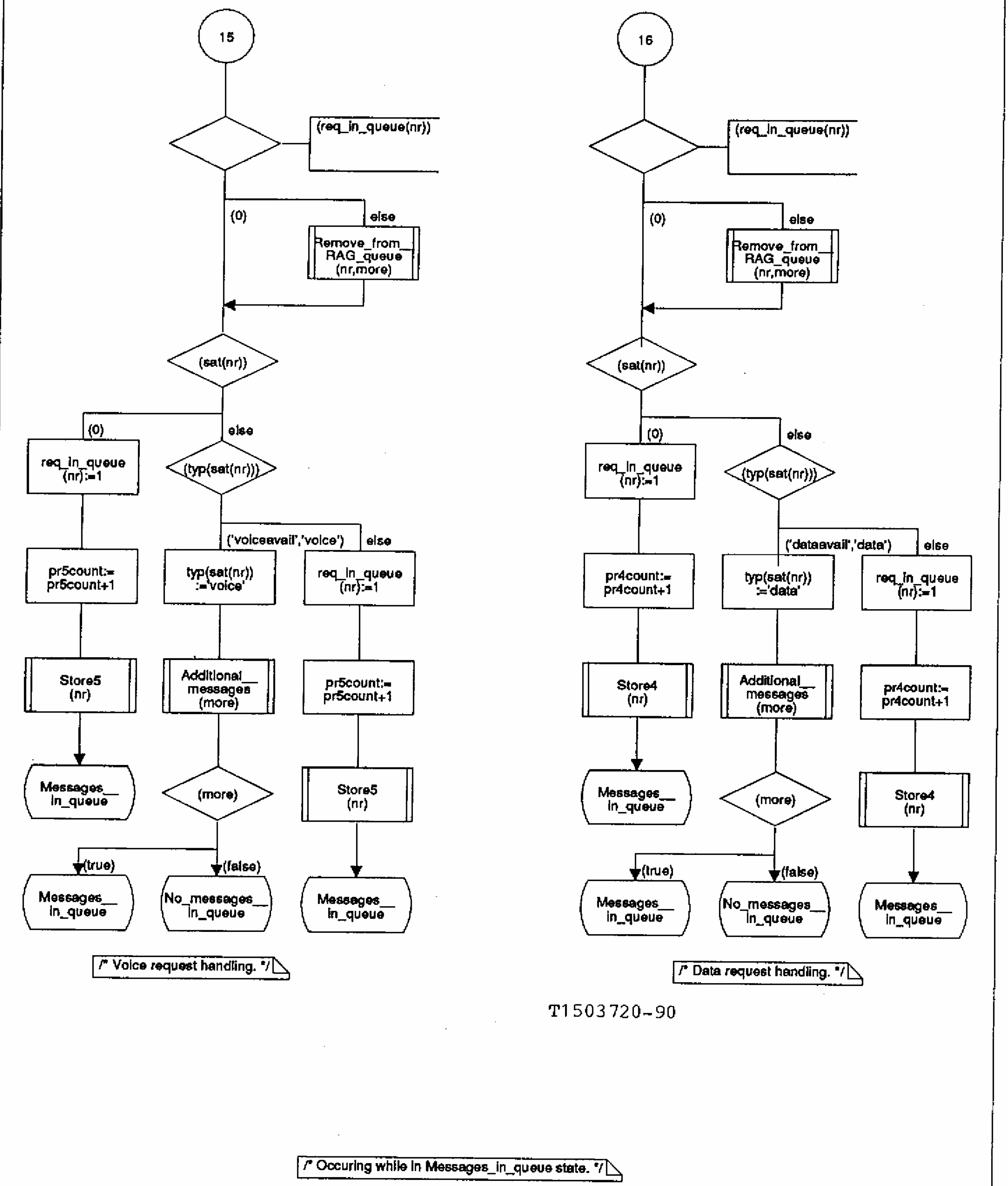

PROCESS RAG
12 (41)
/* All three occuring while in No_messages_in_queue state. */
5
(sat(nr))
(0)
else
req_in_queue (nr):=1
(typ(sat(nr)))
('voiceavail')
else
pr5count:=1
typ(sat(nr)) := 'voice'
req_in_queue (nr):=1
Store5 (nr)
No_messages__ in_queue
pr5count:=1
Messages__ in_queue
Store5 (nr)
/* Voice request storage. */
Messages__ in_queue
6
(sat(nr))
(0)
else
req_in_queue (nr):=1
(typ(sat(nr)))
('dataavail')
else
pr4count:=1
typ(sat(nr)) := 'data'
req_in_queue (nr):=1
Store4 (nr)
No_messages__ in_queue
pr4count:=1
Messages__ in_queue
Store4 (nr)
/* Data request storage. */
Messages__ in_queue
11
(sat(nr))
(0)
else
/* Data inact handling. */
typ(sat(nr)) := 'dataavail'
No_messages__ in_queue
T1503670-90


Messages_
in_queue
Voice(nr)
Sync_
trigger
Data(nr)
Trigger
Transp(nr)
15
k:=0
16
k:=0
17
(sg)
(sg)
(true)
(false)
(true)
(false)
14
14
(f)
(f)
(63)
else
(63)
else
Sync_
alarm
(pnr)
Sync_
alarm
(pnr)
f:=f+1
f:=0
f:=0
57
(f/n =
((f+n–1)/n))
T1510030-92
(false)
(true)
14
57
/* Handling of HSC requests, Trigger and Sync_trigger
signals while in the Messages_in_queue state. */

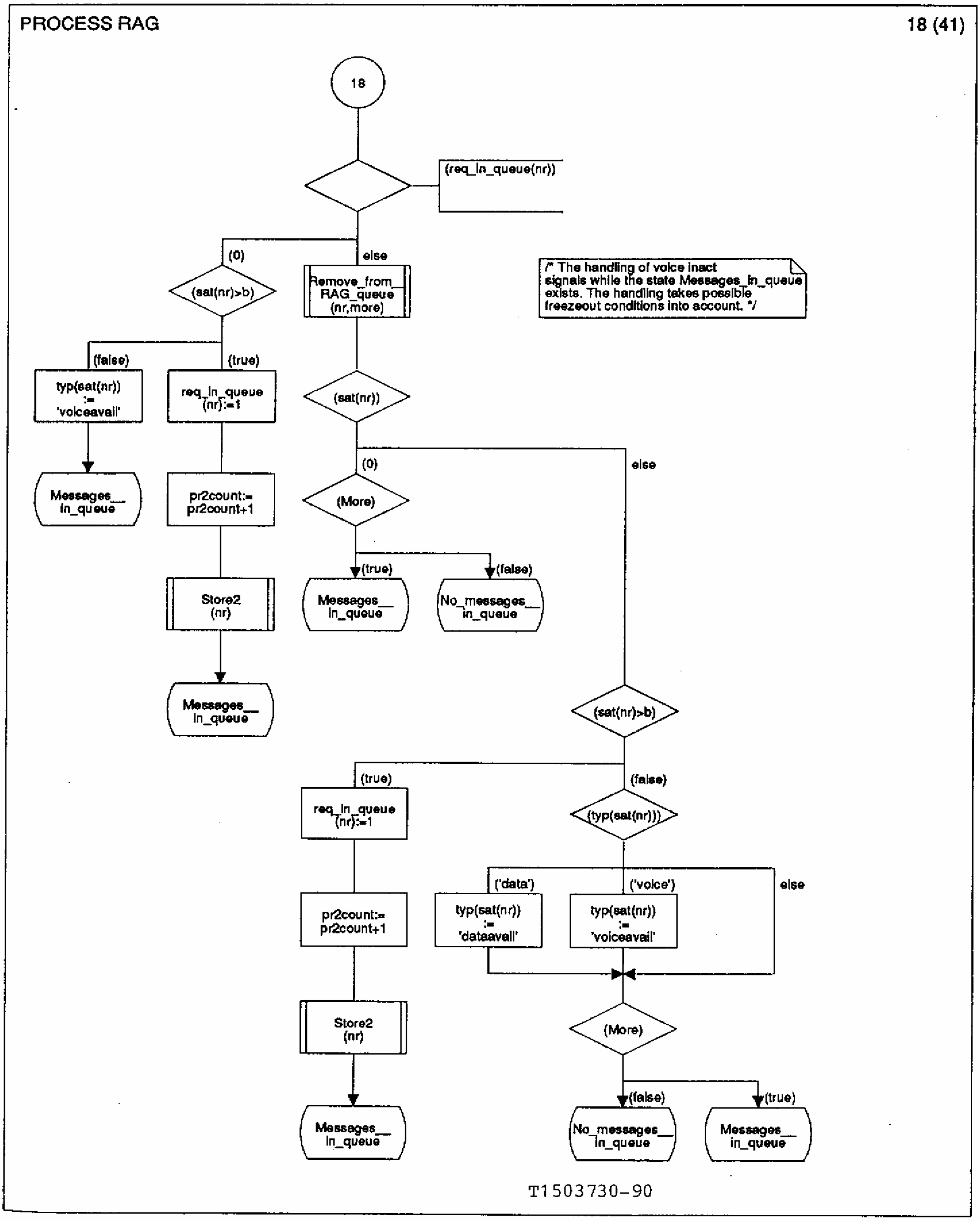

Messages_
in_queue
Voiceinact
(nr)
Discreq
(nr)
Datainact
(nr)
18
20
21
T1510040-92
/* Continuation of HSC request handling. */

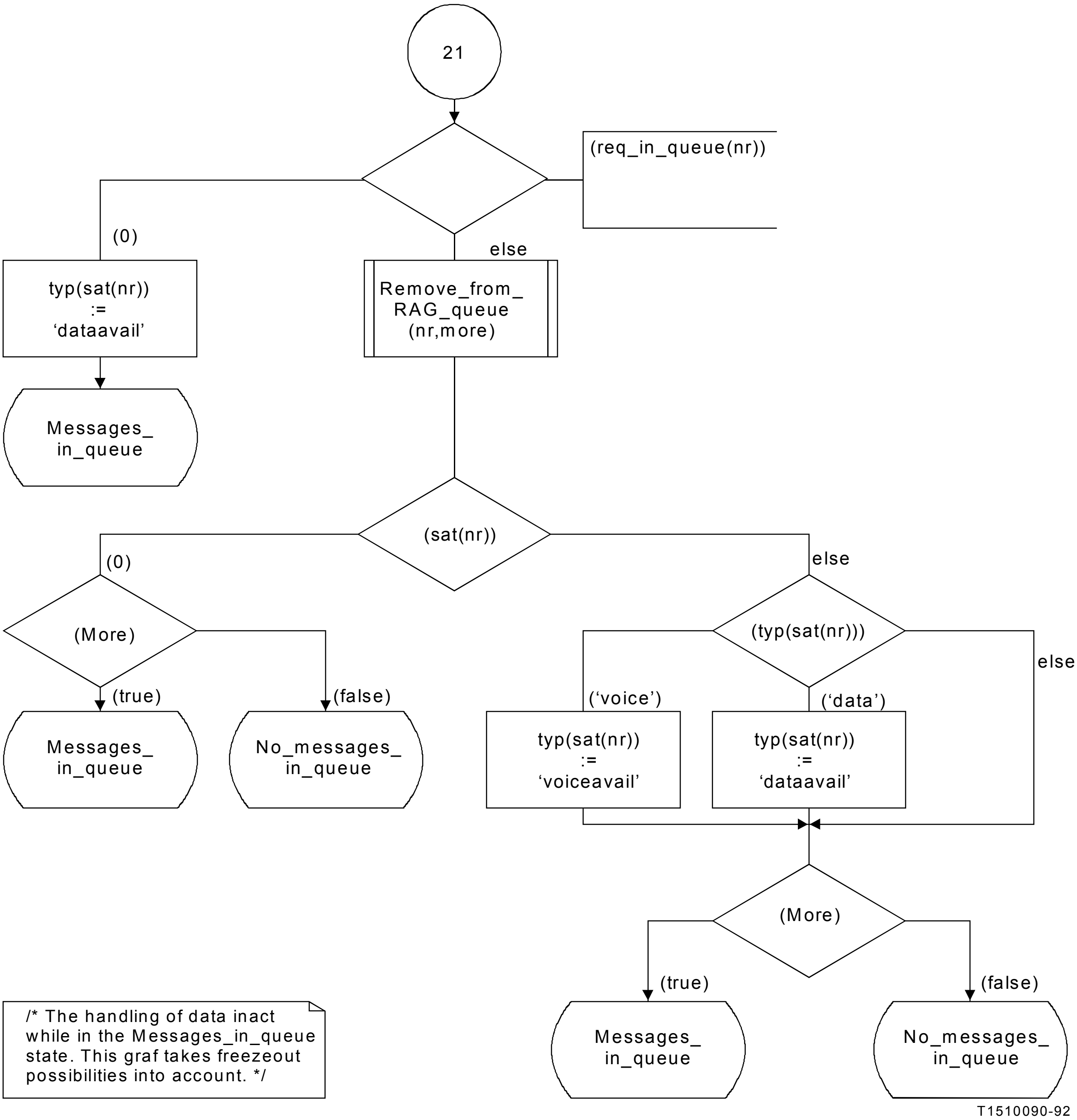

14
(pr1count)
(0)
else
22
(pr2count)
(0)
else
23
(pr3count)
(0)
else
24
(pr4count)
(0)
else
25
(pr5count)
(0)
else
4
25
/* Priority handling. */
T1510050-92


17
(req_in_queue(nr))
(0)
else
Remove_from_
RAG_queue
(nr,more)
req_in_queue
(nr):=1
pr3count:=1
pr3count+1
Store3
(nr)
Messages_
in_queue
/* Transparent request
handling. */
20
(req_in_queue(nr))
(0)
else
Remove_from_
RAG_queue
(nr,more)
req_in_queue
(nr):=0
(req_in_discqueue(nr))
(0)
else
req_in_discqueue
(nr):=1
Messages_
in_queue
pr1count:=
pr1count+1
Store1
(nr)
Messages_
in_queue
/* Transparent disconnect
handling. */
T1510060-92
/* Occuring while in Messages_
in_queue state. */

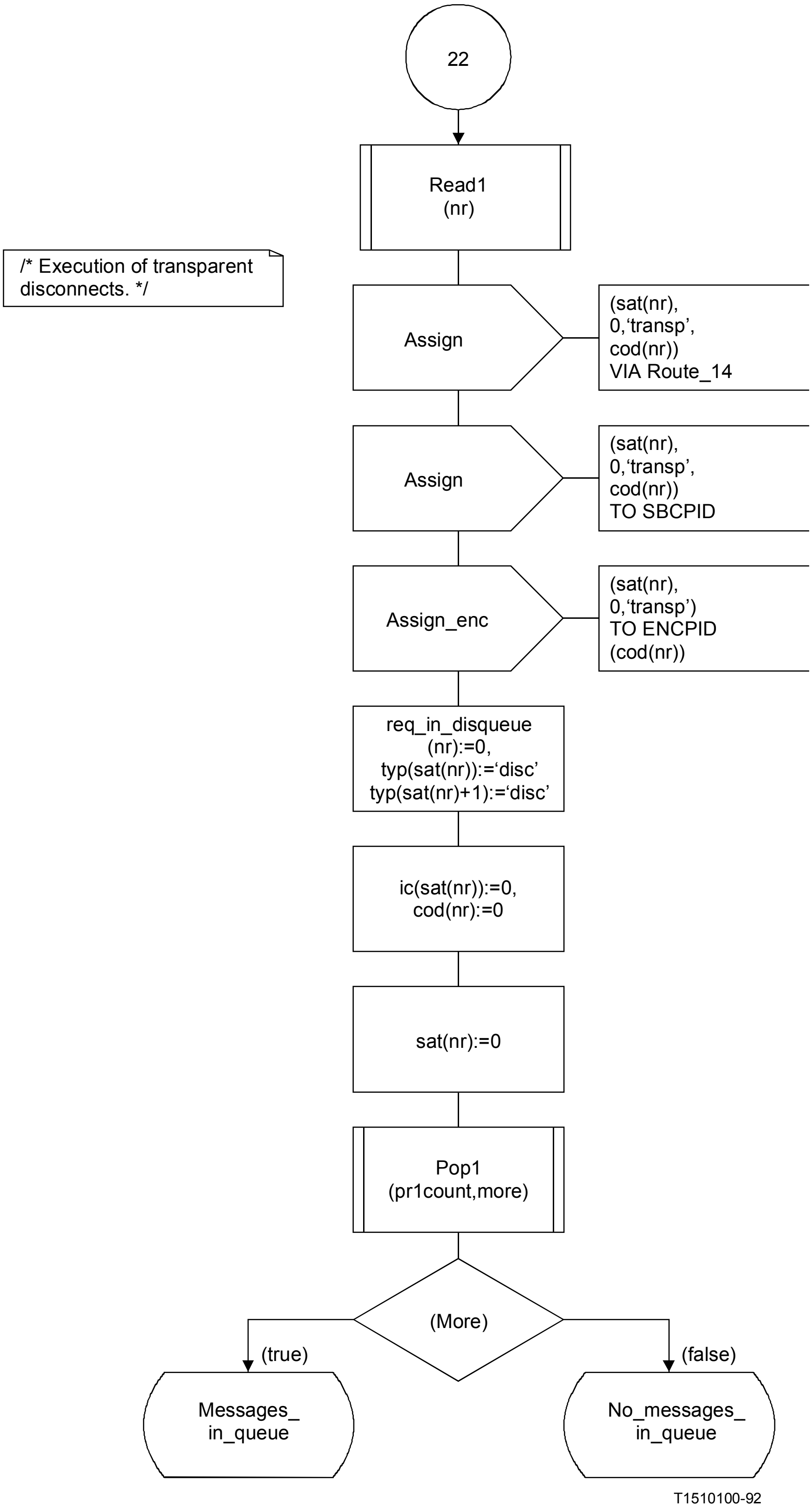


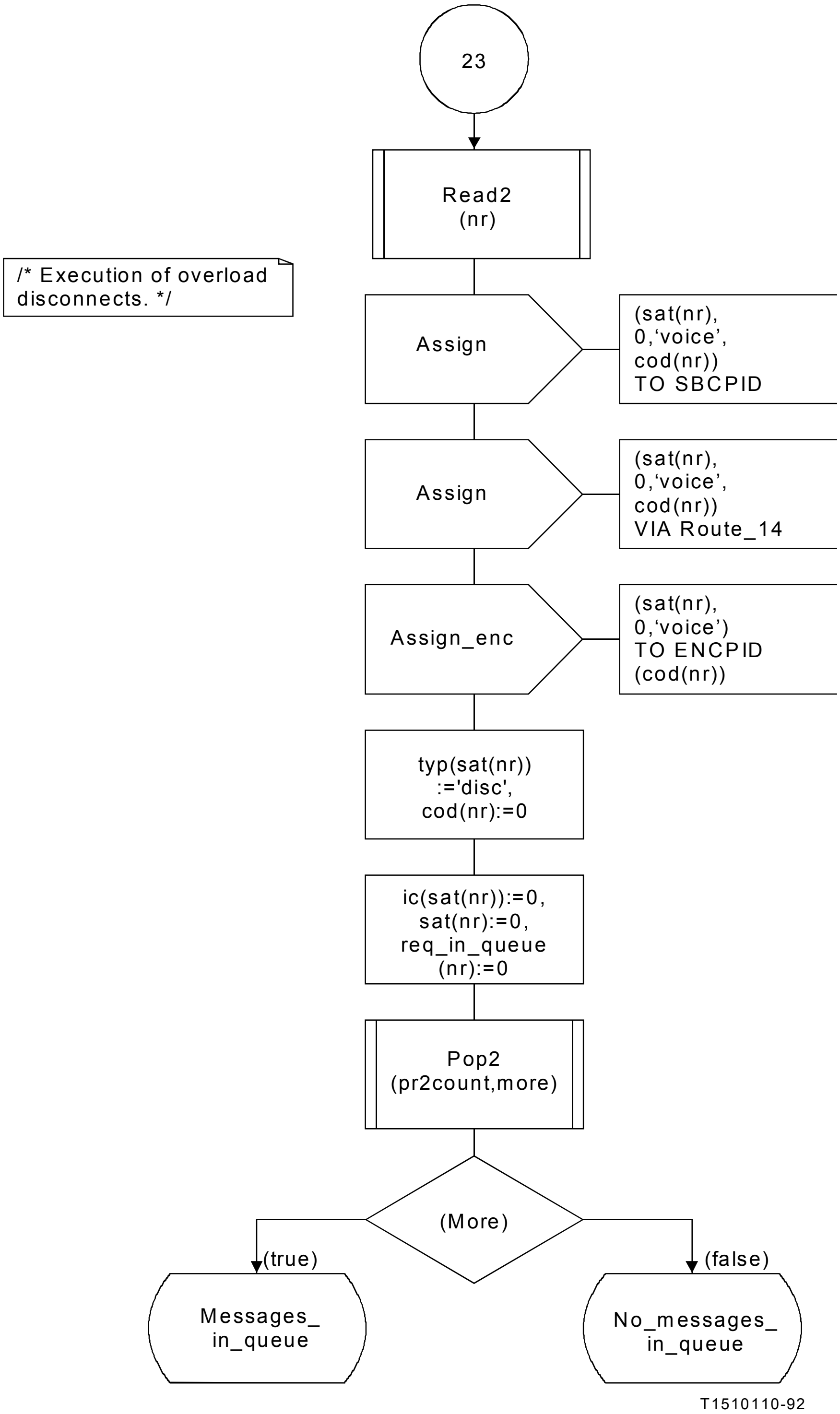
15
(req_in_queue(nr))
(0)
else
Remove_from_RAG_queue (nr,more)
(sat(nr))
(0)
else
req_in_queue (nr):=1
(typ(sat(nr)))
('voiceavail','voice')
else
pr5count:= pr5count+1
typ(sat(nr)) :='voice'
req_in_queue (nr):=1
Store5 (nr)
Additional_messages (more)
pr5count:= pr5count+1
Messages_in_queue
(more)
Store5 (nr)
(true)
(false)
Messages_in_queue
No_messages_in_queue
Messages_in_queue
/* Voice request handling. */
16
(req_in_queue(nr))
(0)
else
Remove_from_RAG_queue (nr,more)
(sat(nr))
(0)
else
req_in_queue (nr):=1
(typ(sat(nr)))
('dataavail','data')
else
pr4count:= pr4count+1
typ(sat(nr)) :='data'
req_in_queue (nr):=1
Store4 (nr)
Additional_messages (more)
pr4count:= pr4count+1
Messages_in_queue
(more)
Store4 (nr)
(true)
(false)
Messages_in_queue
No_messages_in_queue
Messages_in_queue
/* Data request handling. */
T1503720-90
/* Occuring while in Messages_in_queue state. */

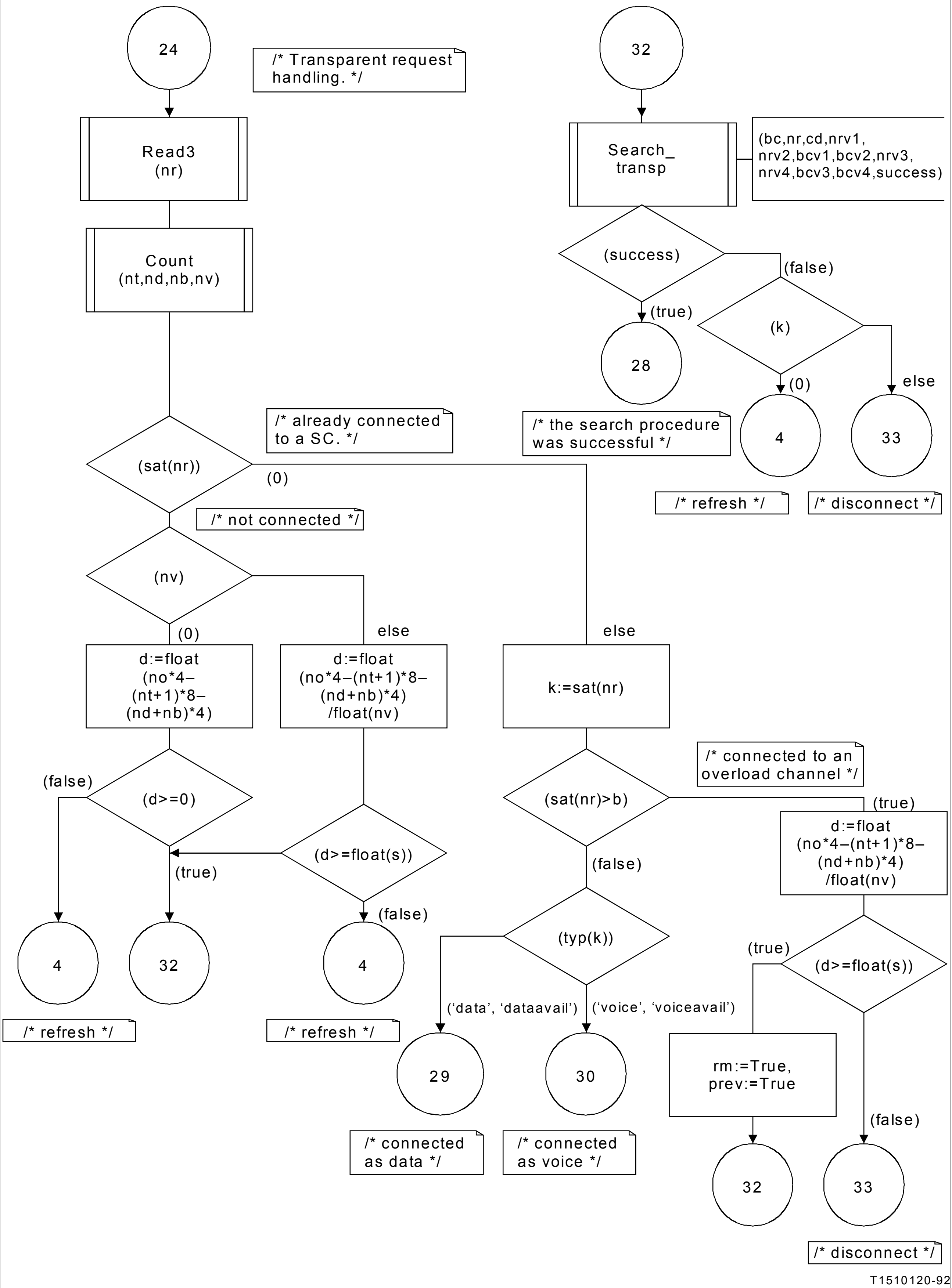

PROCESS RAG
18 (41)
18
(req_in_queue(nr))
(0)
else
(sat(nr)>b)
Remove_from_
RAG_queue
(nr,more)
/* The handling of voice inact signals while the state Messages_in_queue exists. The handling takes possible freezeout conditions into account. */
(false)
(true)
typ(sat(nr))
:=
'voiceavail'
req_in_queue
(nr):=1
(sat(nr))
Messages_
in_queue
pr2count:=
pr2count+1
(0)
else
(More)
Store2
(nr)
(true)
(false)
Messages_
in_queue
No_messages_
in_queue
Messages_
in_queue
(sat(nr)>b)
(true)
(false)
req_in_queue
(nr):=1
(typ(sat(nr)))
('data')
('voice')
else
pr2count:=
pr2count+1
typ(sat(nr))
:=
'dataavail'
typ(sat(nr))
:=
'voiceavail'
Store2
(nr)
(More)
(false)
(true)
Messages_
in_queue
No_messages_
in_queue
Messages_
in_queue
T1503730-90

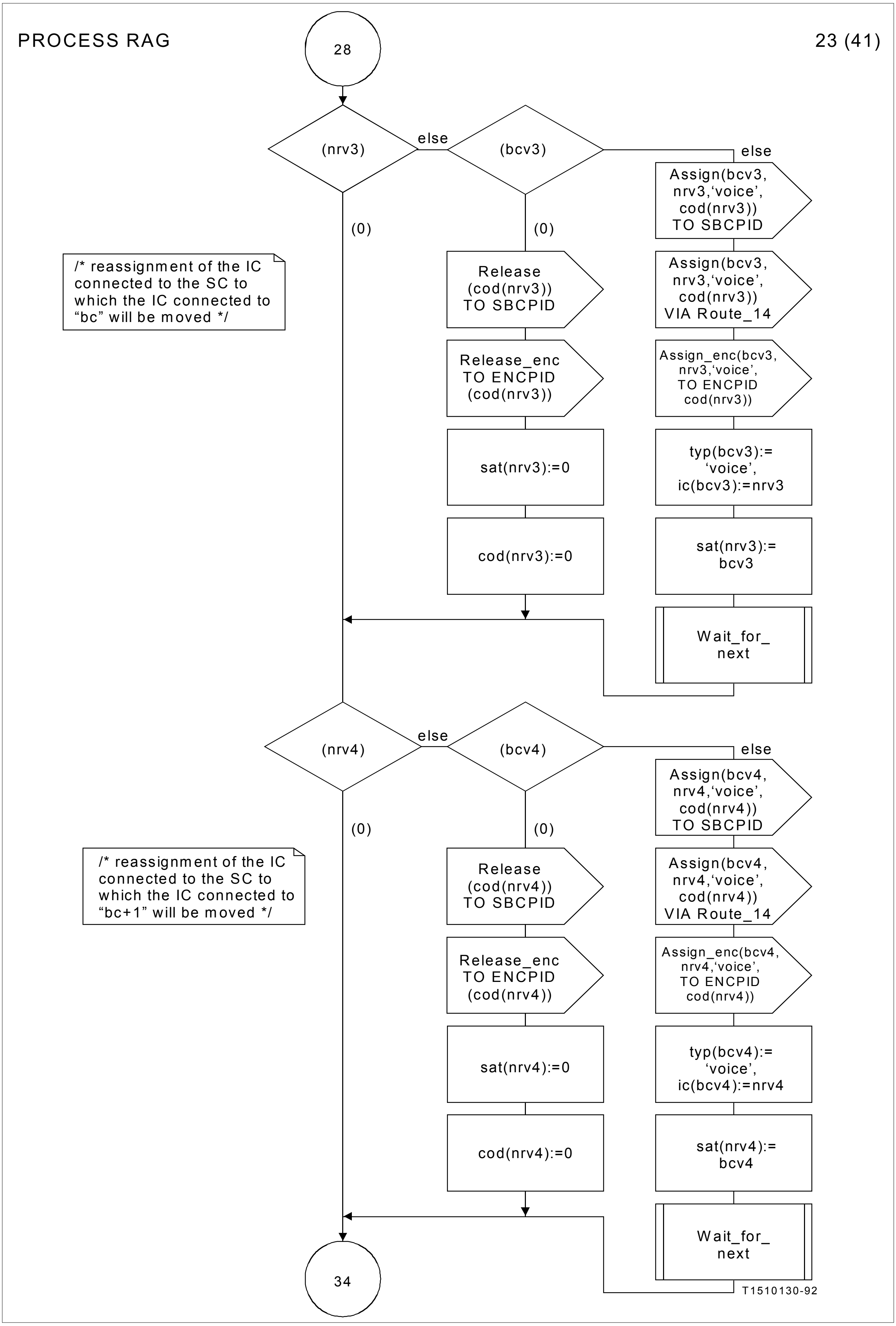


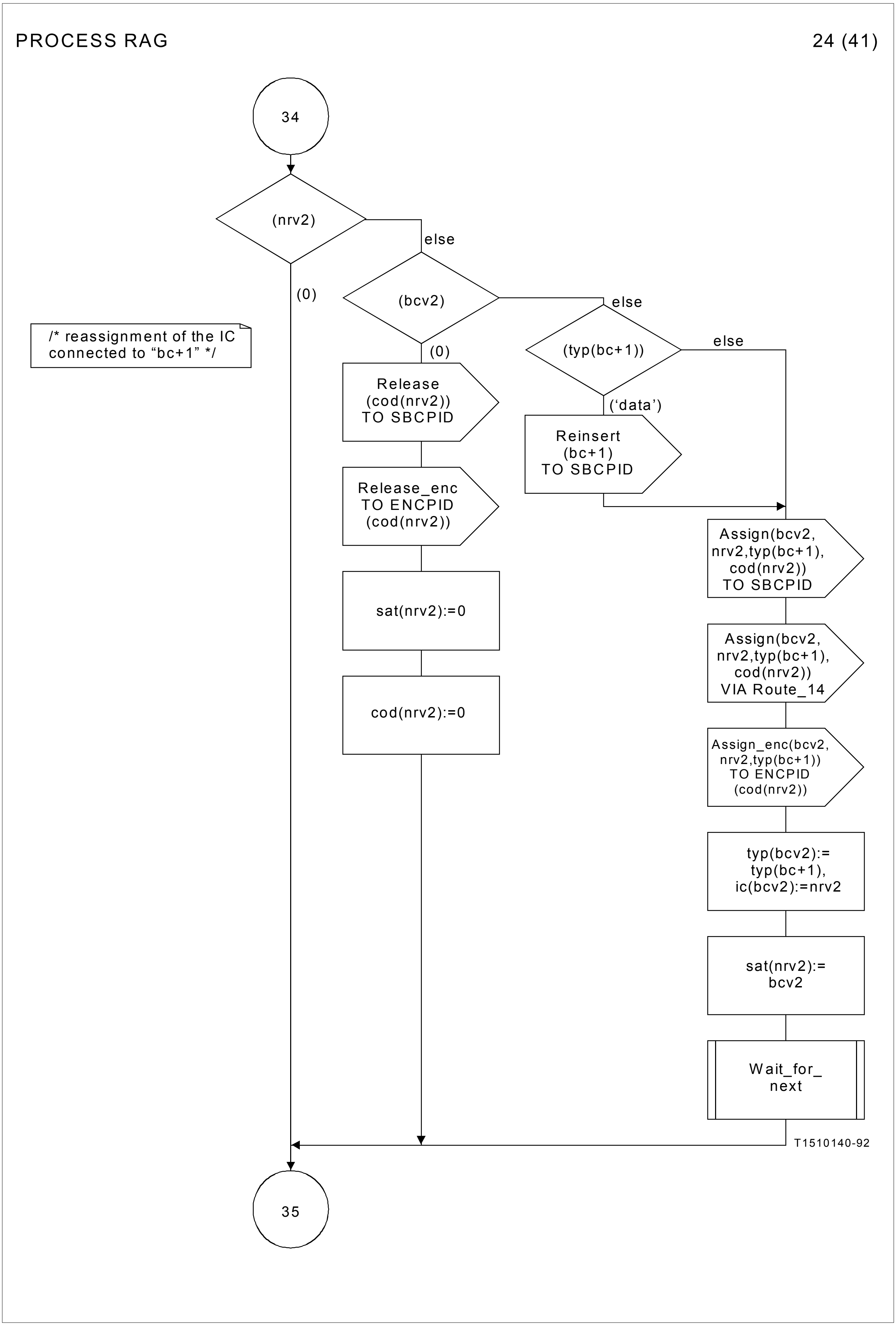
21
(req_in_queue(nr))
(0)
else
typ(sat(nr))
:=
'dataavail'
Remove_from_
RAG_queue
(nr,more)
Messages_
in_queue
(sat(nr))
(0)
else
(More)
(typ(sat(nr)))
else
(true)
(false)
('voice')
('data')
Messages_
in_queue
No_messages_
in_queue
typ(sat(nr))
:=
'voiceavail'
typ(sat(nr))
:=
'dataavail'
(More)
(true)
(false)
/* The handling of data inact
while in the Messages_in_queue
state. This graf takes freezeout
possibilities into account. */
Messages_
in_queue
No_messages_
in_queue
T1510090-92

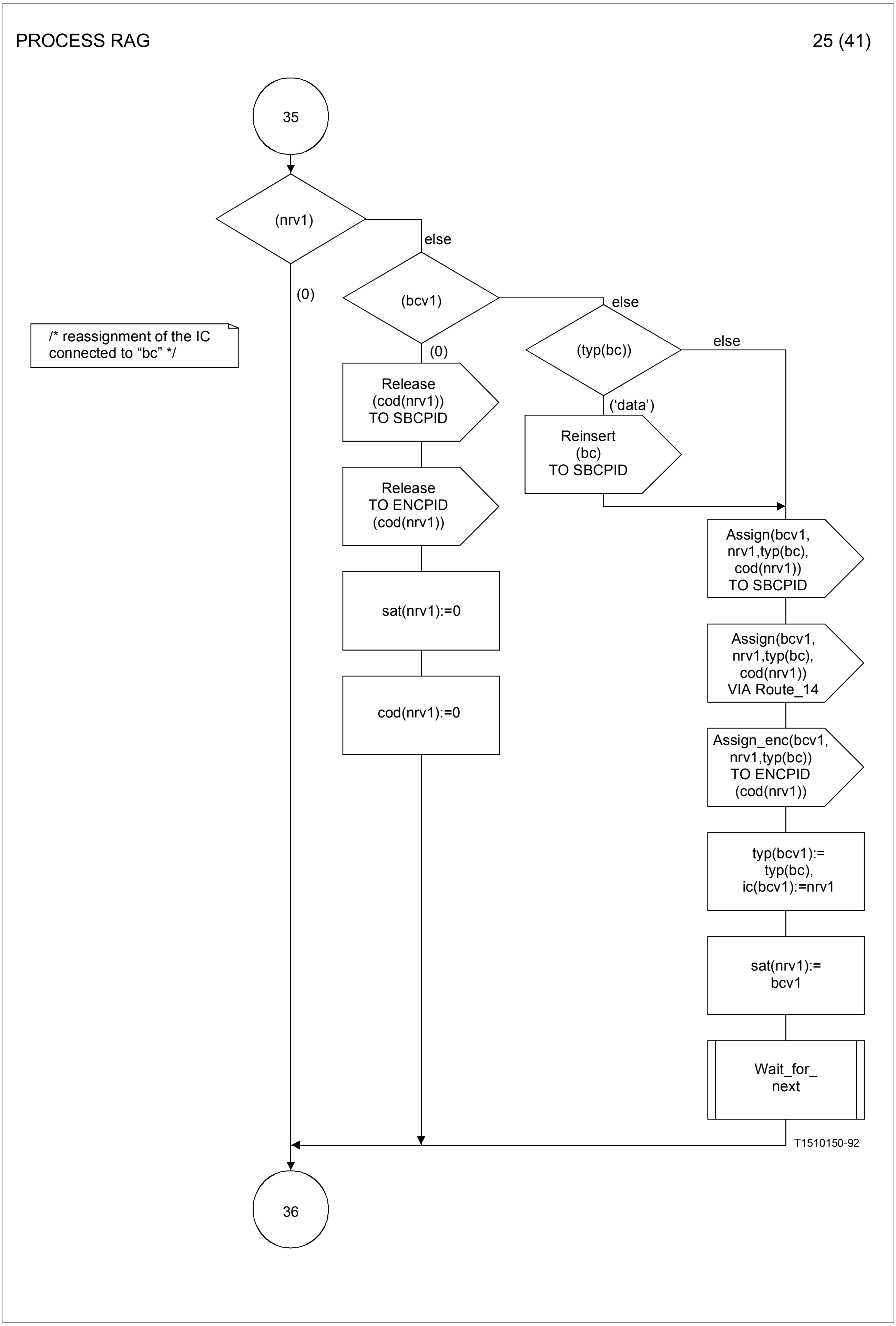

22
Read1
(nr)
/* Execution of transparent
disconnects. */
Assign
(sat(nr),
0,'transp',
cod(nr))
VIA Route_14
Assign
(sat(nr),
0,'transp',
cod(nr))
TO SBCPID
Assign_enc
(sat(nr),
0,'transp')
TO ENCPID
(cod(nr))
req_in_disqueue
(nr):=0,
typ(sat(nr)):='disc'
typ(sat(nr)+1):='disc'
ic(sat(nr)):=0,
cod(nr):=0
sat(nr):=0
Pop1
(pr1count,more)
(More)
(true)
(false)
Messages_
in_queue
No_messages_
in_queue
T1510100-92

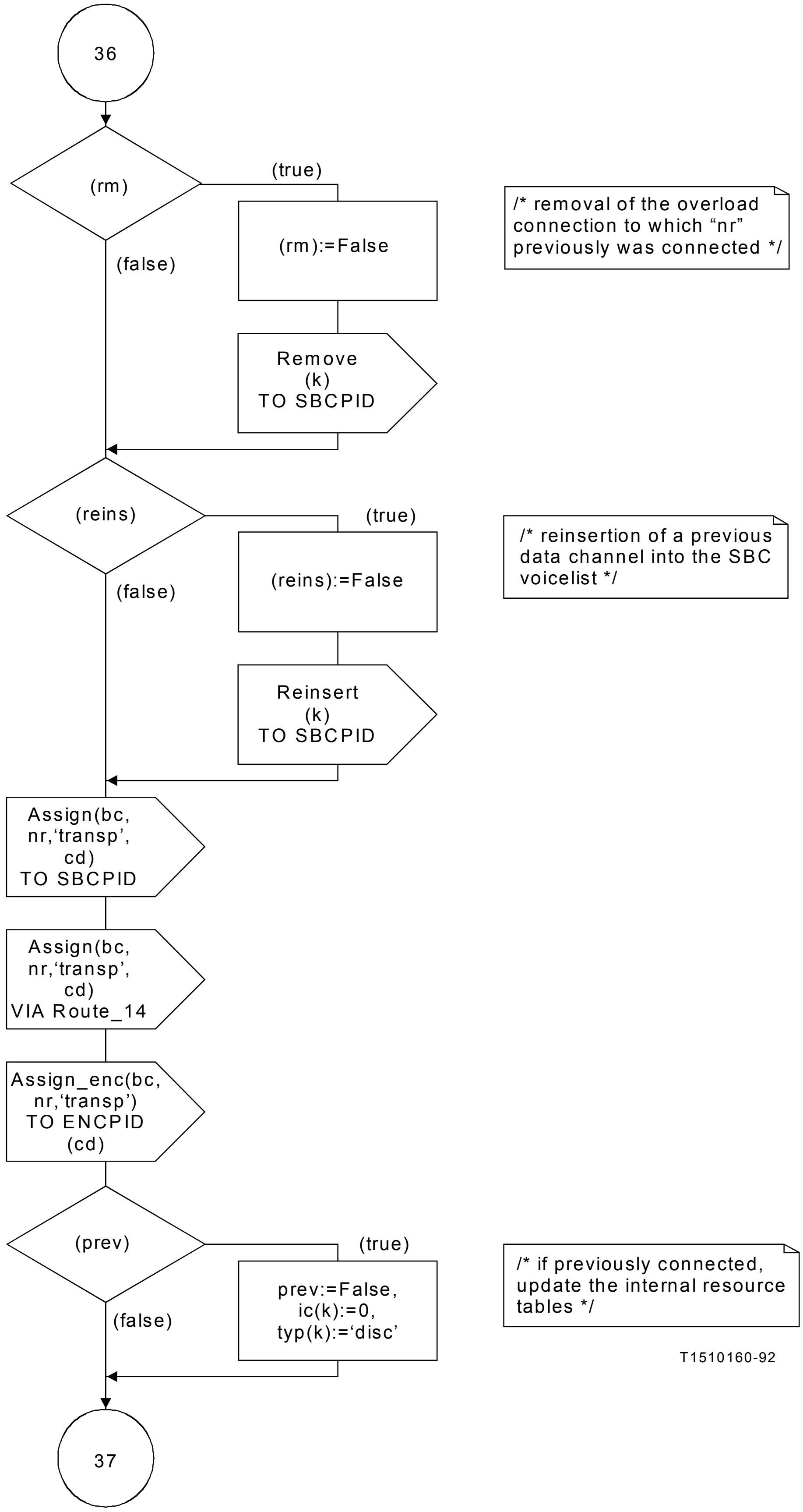

23
Read2 (nr)
/* Execution of overload disconnects. */
Assign
(sat(nr), 0,'voice', cod(nr)) TO SBCPID
Assign
(sat(nr), 0,'voice', cod(nr)) VIA Route_14
Assign_enc
(sat(nr), 0,'voice') TO ENCPID (cod(nr))
typ(sat(nr)) :='disc', cod(nr):=0
ic(sat(nr)):=0, sat(nr):=0, req_in_queue (nr):=0
Pop2 (pr2count,more)
(More)
(true)
(false)
Messages_ in_queue
No_messages_ in_queue
T1510110-92


24
/* Transparent request handling. */
Read3 (nr)
Count (nt,nd,nb,nv)
/* already connected to a SC. */
(sat(nr))
(0)
/* not connected */
(nv)
(0)
else
d:=float (no*4–(nt+1)*8–(nd+nb)*4)
d:=float (no*4–(nt+1)*8–(nd+nb)*4) /float(nv)
(false)
(d>=0)
(true)
(d>=float(s))
(false)
4
32
4
/* refresh */
/* refresh */
else
k:=sat(nr)
/* connected to an overload channel */
(sat(nr)>b)
(true)
(false)
d:=float (no*4–(nt+1)*8–(nd+nb)*4) /float(nv)
(typ(k))
('data', 'dataavail')
('voice', 'voiceavail')
29
30
/* connected as data */
/* connected as voice */
(true)
(d>=float(s))
rm:=True, prev:=True
(false)
32
33
/* disconnect */
32
Search_ transp
(bc,nr,cd,nrv1, nrv2,bcv1,bcv2,nrv3, nrv4,bcv3,bcv4,success)
(success)
(false)
(true)
(k)
28
(0)
else
/* the search procedure was successful */
4
33
/* refresh */
/* disconnect */
T1510120-92

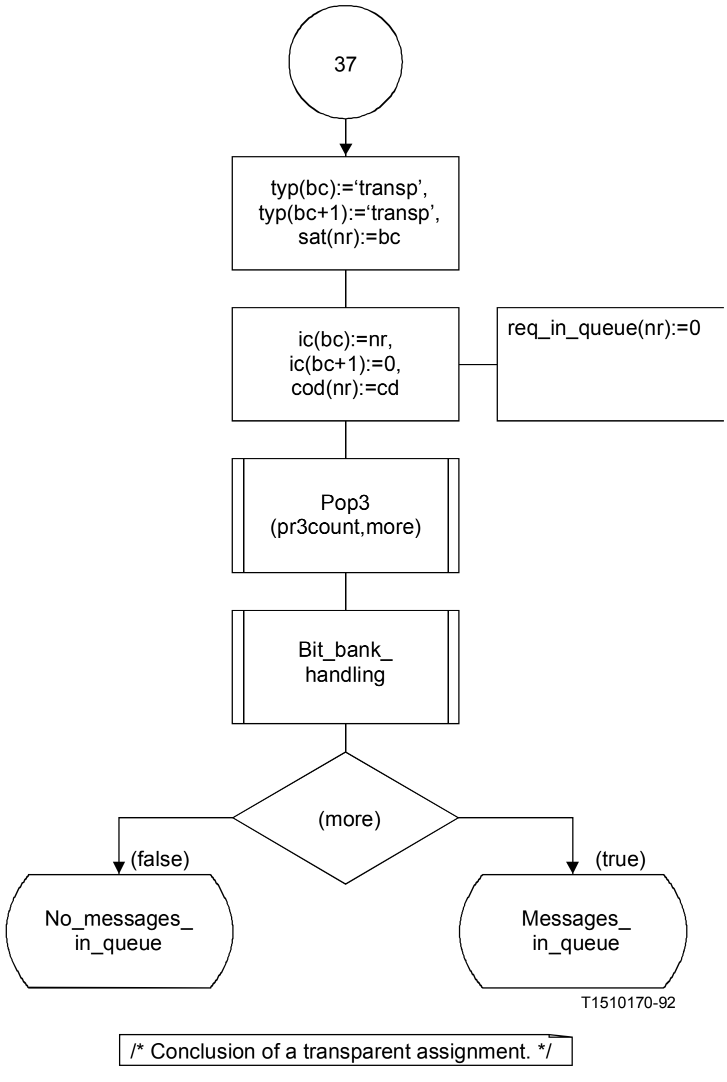

PROCESS RAG
23 (41)
28
(nrv3)
else
(bcv3)
else
Assign(bcv3, nrv3,'voice', cod(nrv3)) TO SBCPID
(0)
(0)
/* reassignment of the IC connected to the SC to which the IC connected to "bc" will be moved */
Release (cod(nrv3)) TO SBCPID
Assign(bcv3, nrv3,'voice', cod(nrv3)) VIA Route_14
Release_enc TO ENCPID (cod(nrv3))
Assign_enc(bcv3, nrv3,'voice', TO ENCPID cod(nrv3))
sat(nrv3):=0
typ(bcv3):= 'voice', ic(bcv3):=nrv3
cod(nrv3):=0
sat(nrv3):= bcv3
Wait_for_ next
(nrv4)
else
(bcv4)
else
Assign(bcv4, nrv4,'voice', cod(nrv4)) TO SBCPID
(0)
(0)
/* reassignment of the IC connected to the SC to which the IC connected to "bc+1" will be moved */
Release (cod(nrv4)) TO SBCPID
Assign(bcv4, nrv4,'voice', cod(nrv4)) VIA Route_14
Release_enc TO ENCPID (cod(nrv4))
Assign_enc(bcv4, nrv4,'voice', TO ENCPID cod(nrv4))
sat(nrv4):=0
typ(bcv4):= 'voice', ic(bcv4):=nrv4
cod(nrv4):=0
sat(nrv4):= bcv4
Wait_for_ next
34
T1510130-92

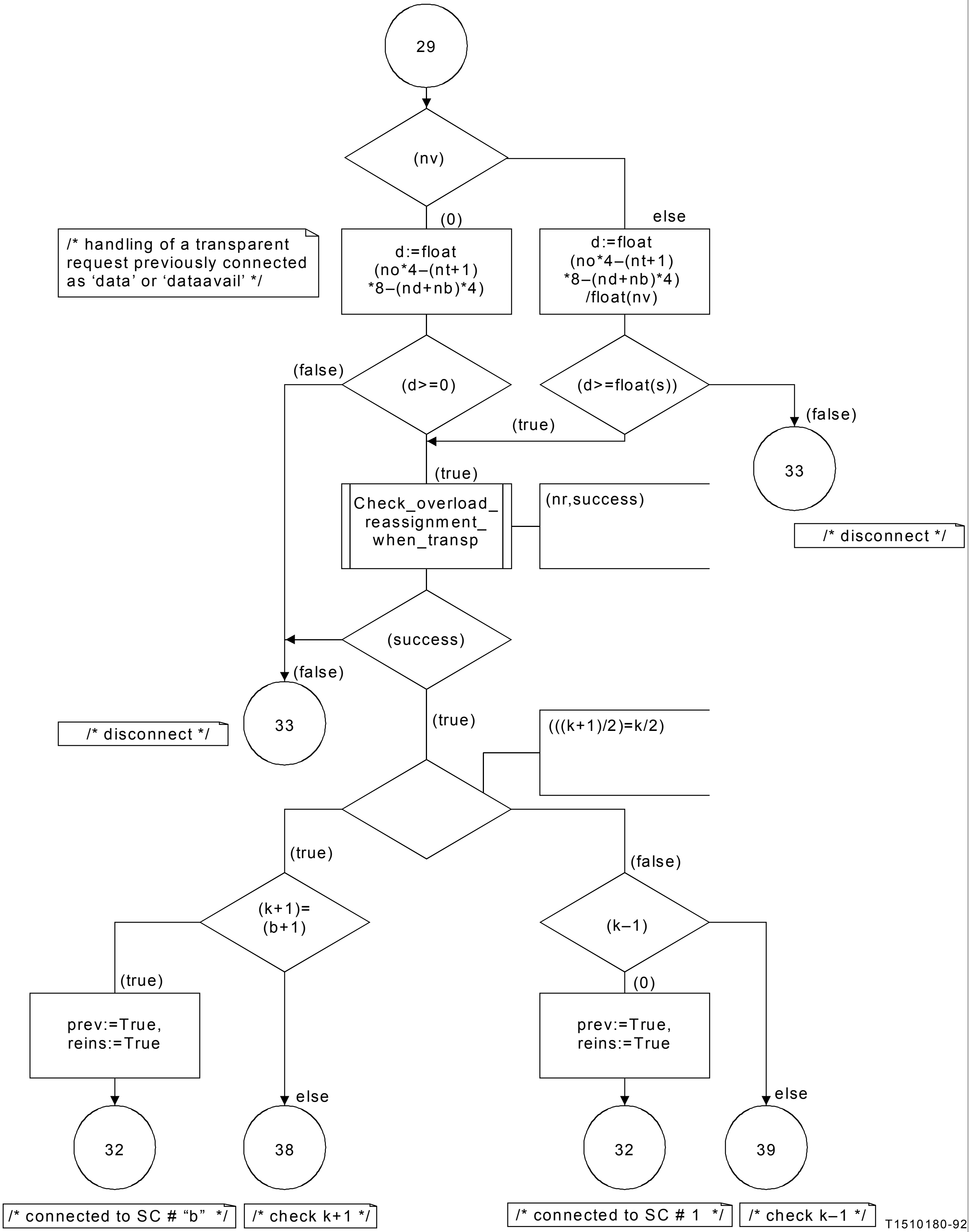

PROCESS RAG
24 (41)
34
(nrv2)
else
(0)
(bcv2)
else
/* reassignment of the IC connected to “bc+1” */
(0)
(typ(bc+1))
else
Release (cod(nrv2)) TO SBCPID
(‘data’)
Reinsert (bc+1) TO SBCPID
Release_enc TO ENCPID (cod(nrv2))
Assign(bcv2, nrv2,typ(bc+1), cod(nrv2)) TO SBCPID
sat(nrv2):=0
Assign(bcv2, nrv2,typ(bc+1), cod(nrv2)) VIA Route_14
cod(nrv2):=0
Assign_enc(bcv2, nrv2,typ(bc+1)) TO ENCPID (cod(nrv2))
typ(bcv2):= typ(bc+1), ic(bcv2):=nrv2
sat(nrv2):= bcv2
Wait_for_ next
T1510140-92
35


PROCESS RAG
25 (41)
35
(nrv1)
else
(0)
(bcv1)
else
/* reassignment of the IC connected to "bc" */
(0)
(typ(bc))
else
Release (cod(nrv1)) TO SBCPID
('data')
Reinsert (bc) TO SBCPID
Release TO ENCPID (cod(nrv1))
Assign(bcv1, nrv1,typ(bc), cod(nrv1)) TO SBCPID
sat(nrv1):=0
Assign(bcv1, nrv1,typ(bc), cod(nrv1)) VIA Route_14
cod(nrv1):=0
Assign_enc(bcv1, nrv1,typ(bc)) TO ENCPID (cod(nrv1))
typ(bcv1):= typ(bc), ic(bcv1):=nrv1
sat(nrv1):= bcv1
Wait_for_ next
T1510150-92
36



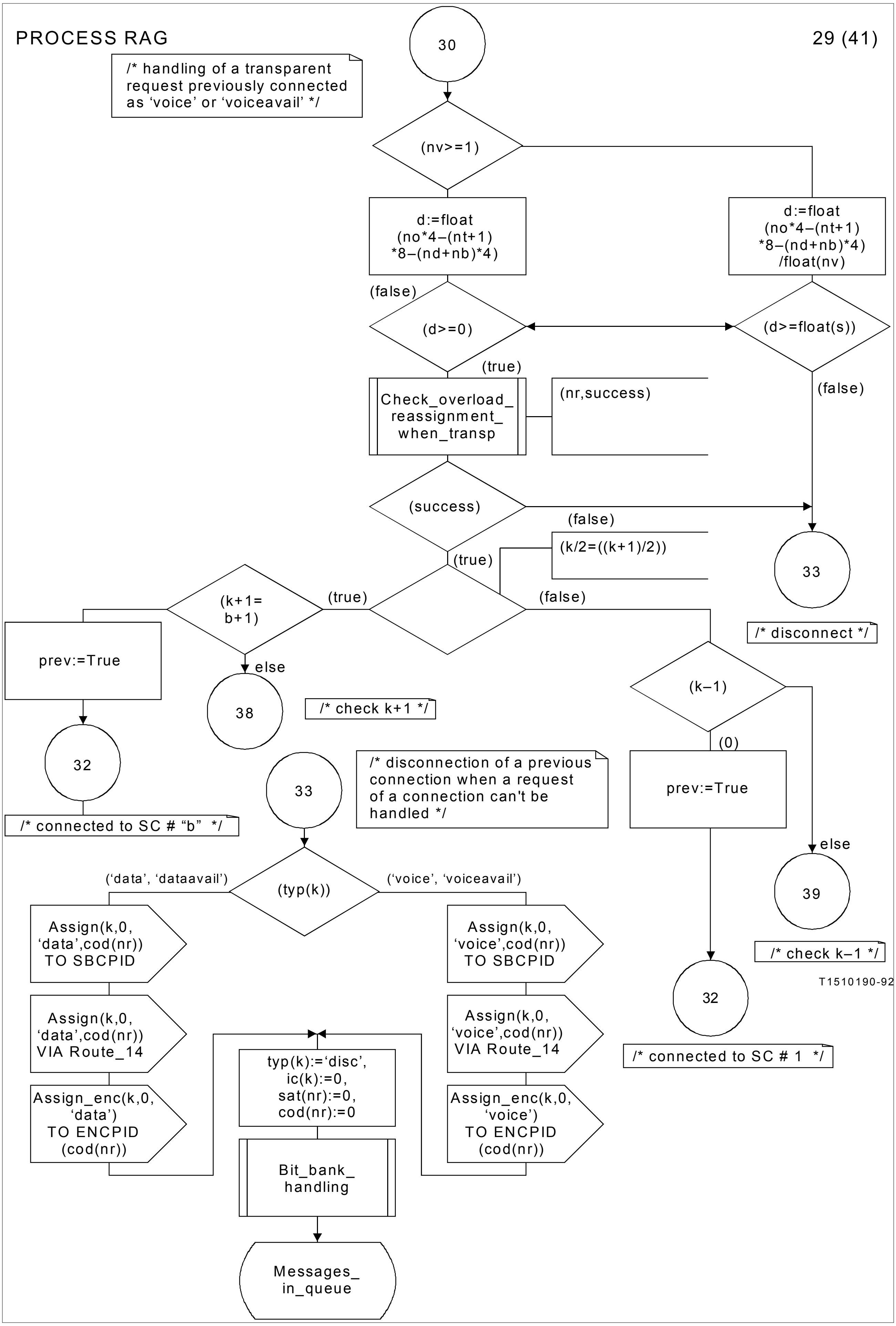
36
(rm)
(true)
(false)
(rm):=False
Remove (k) TO SBCPID
/* removal of the overload connection to which "nr" previously was connected */
(reins)
(true)
(false)
(reins):=False
Reinsert (k) TO SBCPID
/* reinsertion of a previous data channel into the SBC voicelist */
Assign(bc, nr,'transp', cd) TO SBCPID
Assign(bc, nr,'transp', cd) VIA Route_14
Assign_enc(bc, nr,'transp') TO ENCPID (cd)
(prev)
(true)
(false)
prev:=False, ic(k):=0, typ(k):='disc'
/* if previously connected, update the internal resource tables */
T1510160-92
37

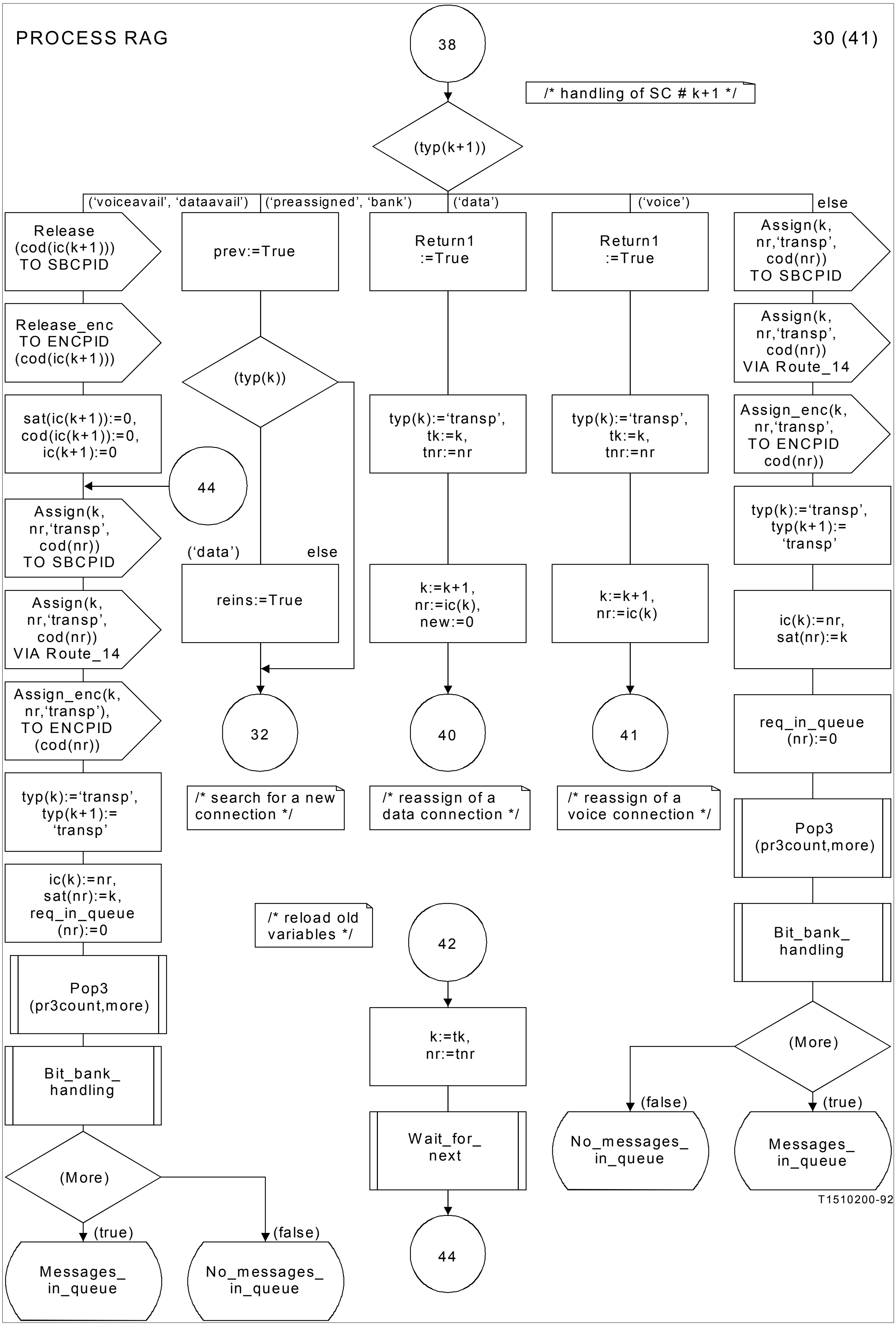

37
typ(bc):='transp',
typ(bc+1):='transp',
sat(nr):=bc
ic(bc):=nr,
ic(bc+1):=0,
cod(nr):=cd
req_in_queue(nr):=0
Pop3
(pr3count,more)
Bit_bank_
handling
(more)
(false)
(true)
No_messages_
in_queue
Messages_
in_queue
T1510170-92
/* Conclusion of a transparent assignment. */

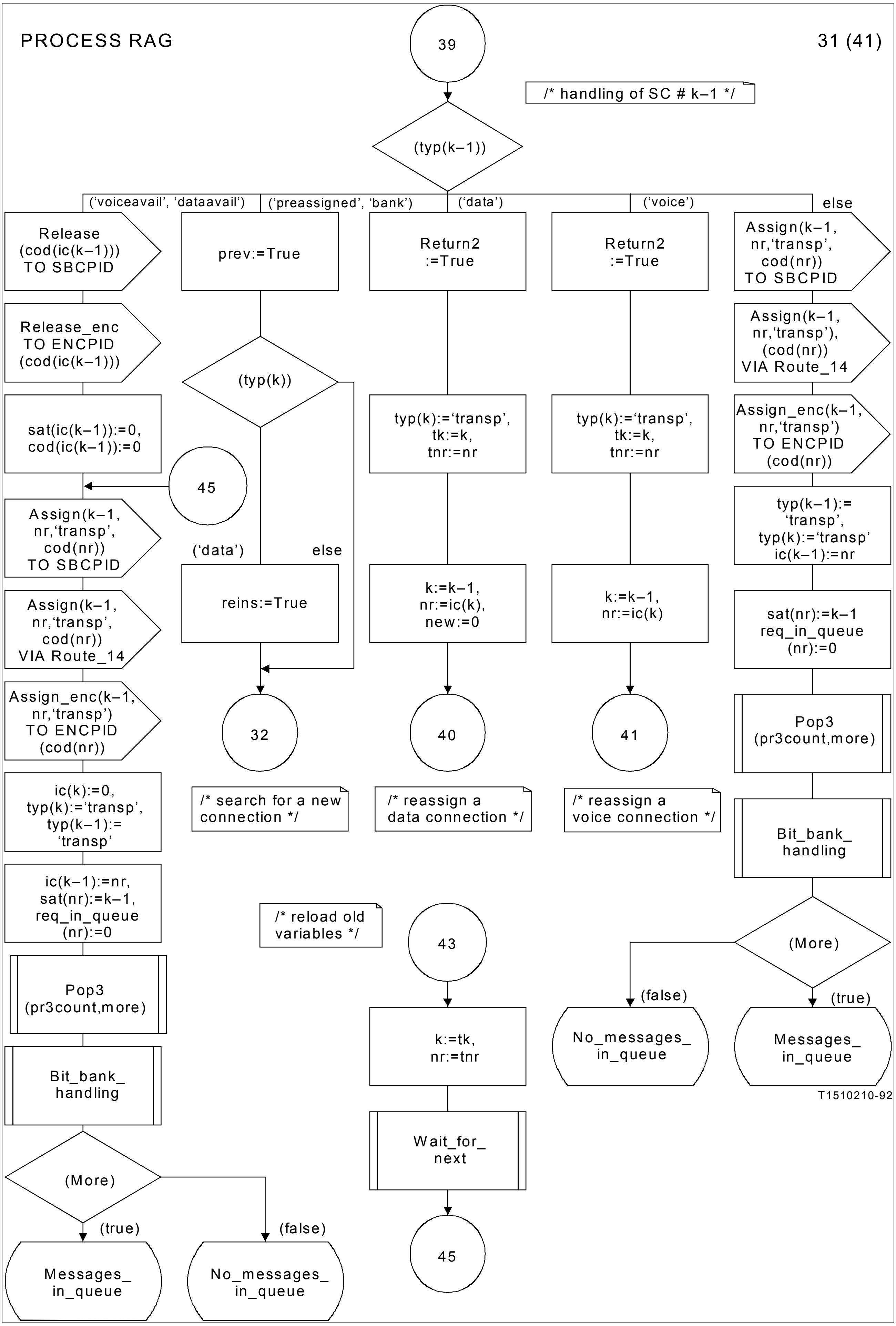


29
(nv)
(0)
else
/* handling of a transparent request previously connected as 'data' or 'dataavail' */
d:=float (no*4–(nt+1) *8–(nd+nb)*4)
d:=float (no*4–(nt+1) *8–(nd+nb)*4) /float(nv)
(false)
(d>=0)
(d>=float(s))
(true)
(false)
33
(true)
Check_overload_ reassignment_ when_transp
(nr,success)
/* disconnect */
(success)
(false)
33
/* disconnect */
(true)
(((k+1)/2)=k/2)
(true)
(false)
(k+1)= (b+1)
(k–1)
(true)
(0)
prev:=True, reins:=True
prev:=True, reins:=True
else
else
32
38
32
39
/* connected to SC # "b" */
/* check k+1 */
/* connected to SC # 1 */
/* check k–1 */

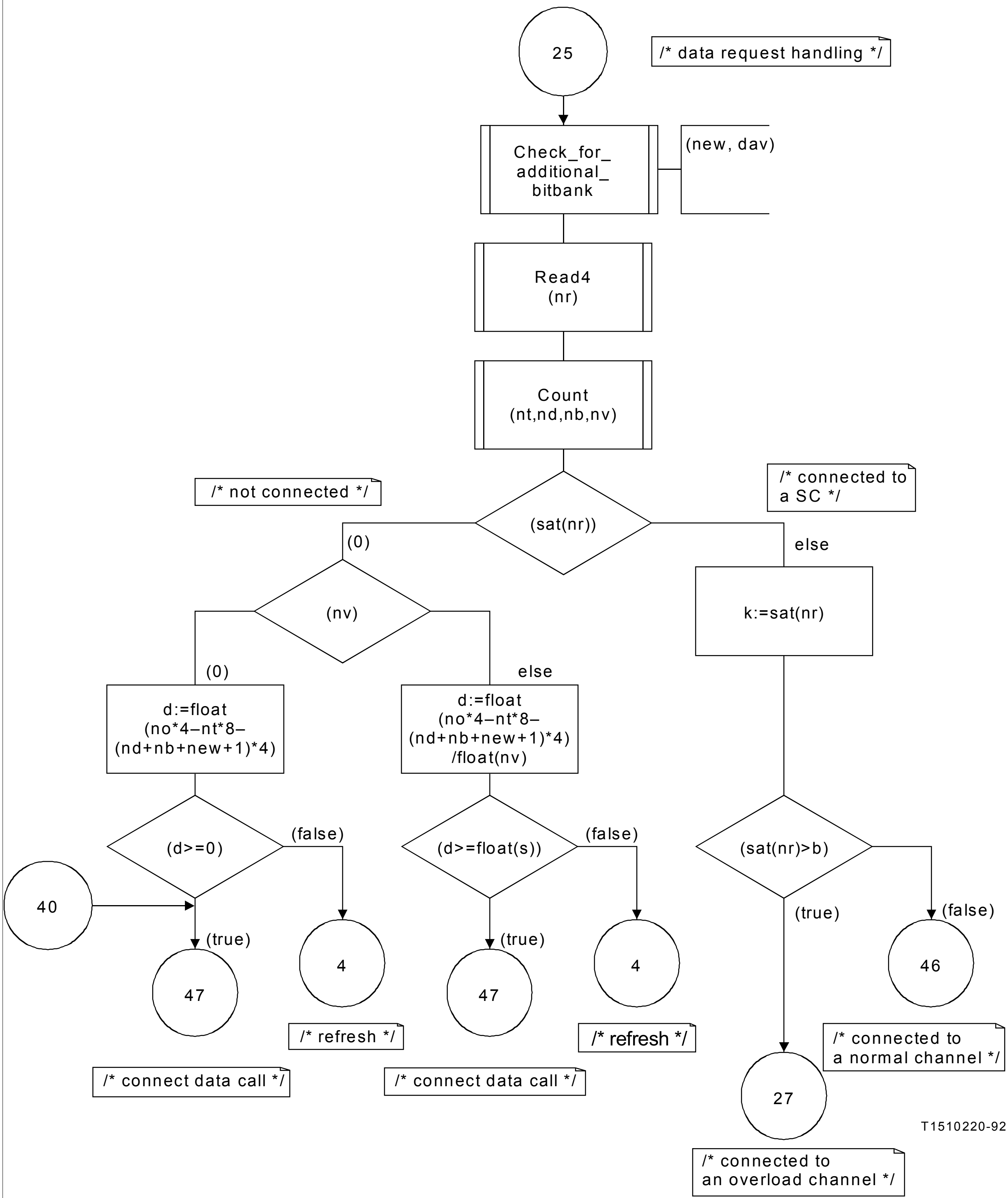
T1510180-92

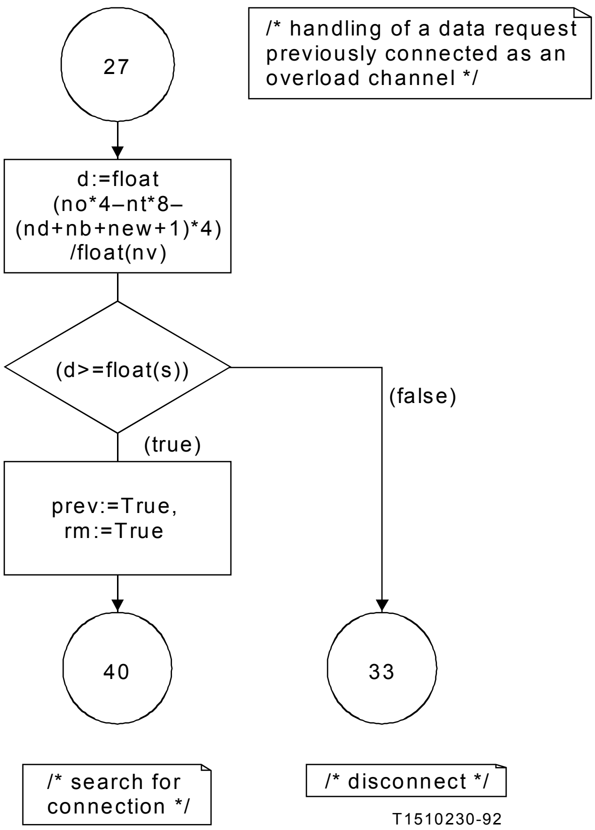

PROCESS RAG
29 (41)
30
/* handling of a transparent request previously connected as 'voice' or 'voiceavail' */
(nv>=1)
d:=float (no*4–(nt+1) *8–(nd+nb)*4)
d:=float (no*4–(nt+1) *8–(nd+nb)*4) /float(nv)
(false)
(d>=0)
(d>=float(s))
(true)
Check_overload_ reassignment_ when_transp
(nr,success)
(false)
(success)
(false)
(k/2=((k+1)/2))
(true)
33
(k+1= b+1)
(true)
(false)
/* disconnect */
prev:=True
else
(k–1)
38
/* check k+1 */
(0)
32
33
/* disconnection of a previous connection when a request of a connection can't be handled */
prev:=True
/* connected to SC # "b" */
else
('data', 'dataavail')
(typ(k))
('voice', 'voiceavail')
39
Assign(k,0, 'data',cod(nr)) TO SBCPID
Assign(k,0, 'voice',cod(nr)) TO SBCPID
/* check k–1 */
T1510190-92
Assign(k,0, 'data',cod(nr)) VIA Route_14
Assign(k,0, 'voice',cod(nr)) VIA Route_14
32
/* connected to SC # 1 */
typ(k):='disc', ic(k):=0, sat(nr):=0, cod(nr):=0
Assign_enc(k,0, 'data') TO ENCPID (cod(nr))
Assign_enc(k,0, 'voice') TO ENCPID (cod(nr))
Bit_bank_ handling
Messages_ in_queue

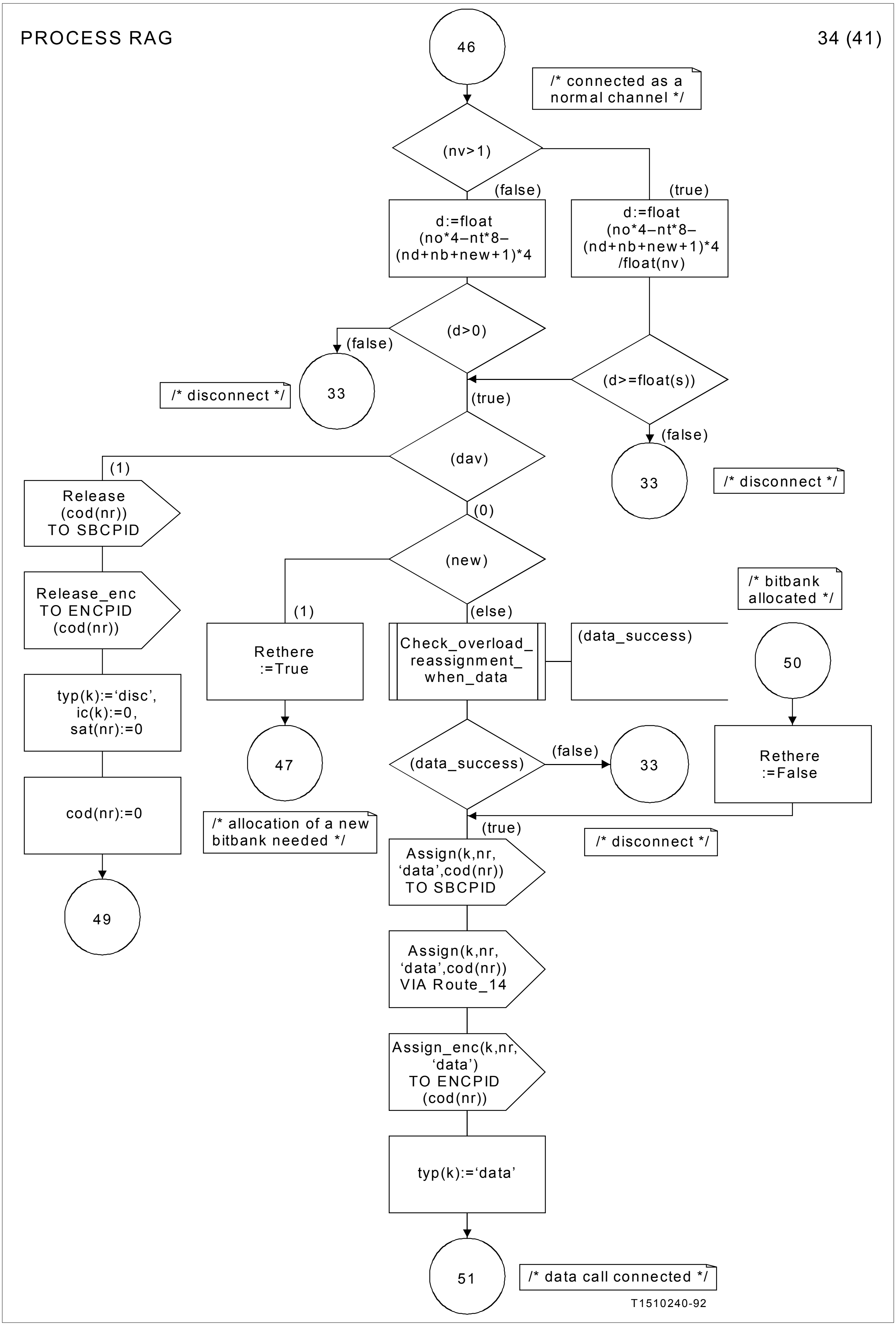

PROCESS RAG
30 (41)
38
/* handling of SC # k+1 */
(typ(k+1))
('voiceavail', 'dataavail')
('preassigned', 'bank')
('data')
('voice')
else
Release (cod(ic(k+1))) TO SBCPID
Release_enc TO ENCPID (cod(ic(k+1)))
sat(ic(k+1)):=0, cod(ic(k+1)):=0, ic(k+1):=0
44
Assign(k, nr,'transp', cod(nr)) TO SBCPID
Assign(k, nr,'transp', cod(nr)) VIA Route_14
Assign_enc(k, nr,'transp'), TO ENCPID (cod(nr))
typ(k):='transp', typ(k+1):= 'transp'
ic(k):=nr, sat(nr):=k, req_in_queue (nr):=0
Pop3 (pr3count,more)
Bit_bank_ handling
(More)
(true)
(false)
Messages_ in_queue
No_messages_ in_queue
prev:=True
(typ(k))
('data')
else
reins:=True
32
/* search for a new connection */
Return1 :=True
typ(k):='transp', tk:=k, tnr:=nr
k:=k+1, nr:=ic(k), new:=0
40
/* reassign of a data connection */
Return1 :=True
typ(k):='transp', tk:=k, tnr:=nr
k:=k+1, nr:=ic(k)
41
/* reassign of a voice connection */
Assign(k, nr,'transp', cod(nr)) TO SBCPID
Assign(k, nr,'transp', cod(nr)) VIA Route_14
Assign_enc(k, nr,'transp', TO ENCPID cod(nr))
typ(k):='transp', typ(k+1):= 'transp'
ic(k):=nr, sat(nr):=k
req_in_queue (nr):=0
Pop3 (pr3count,more)
Bit_bank_ handling
(More)
(false)
(true)
No_messages_ in_queue
Messages_ in_queue
/* reload old variables */
42
k:=tk, nr:=tnr
Wait_for_ next
44
T1510200-92

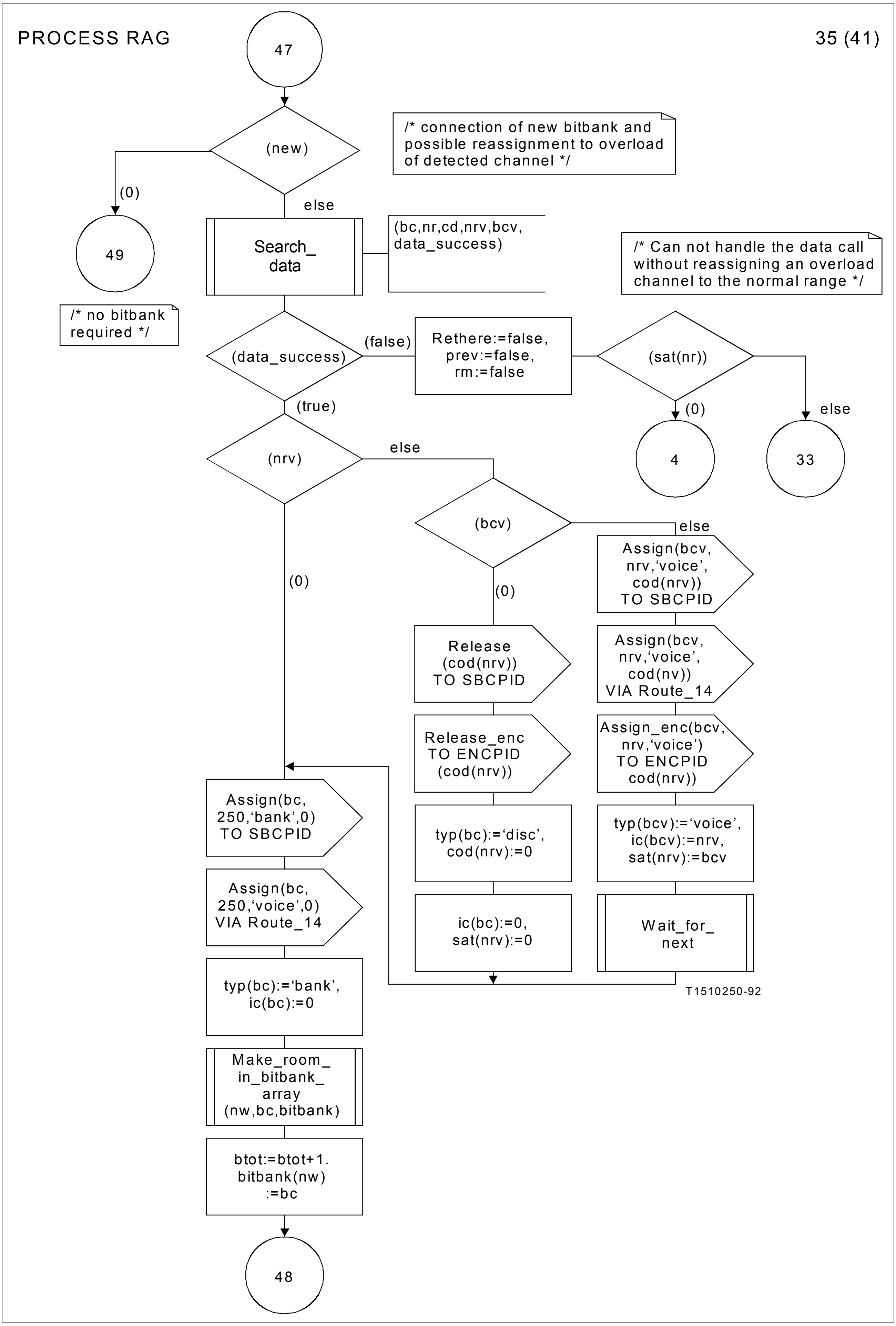


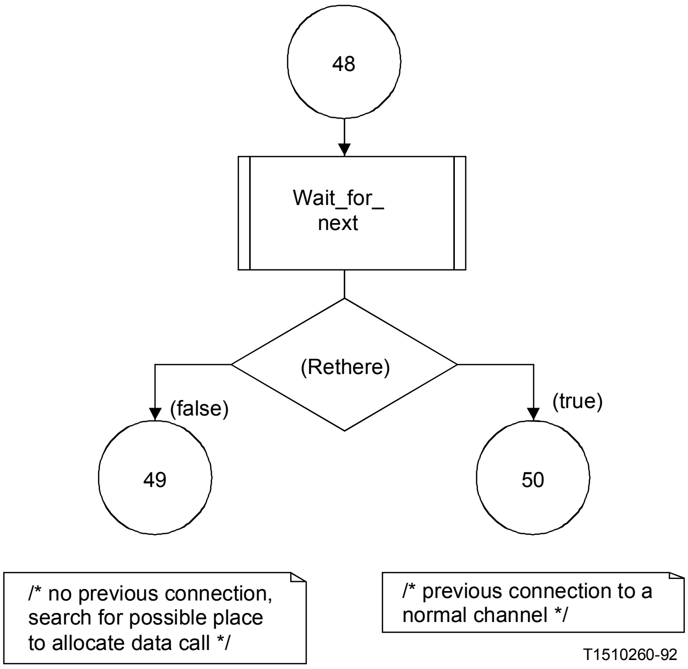
PROCESS RAG
31 (41)
39
/* handling of SC # k–1 */
(typ(k–1))
('voiceavail', 'dataavail')
('preassigned', 'bank')
('data')
('voice')
else
Release (cod(ic(k–1))) TO SBCPID
Release_enc TO ENCPID (cod(ic(k–1)))
sat(ic(k–1)):=0, cod(ic(k–1)):=0
45
Assign(k–1, nr,'transp', cod(nr)) TO SBCPID
Assign(k–1, nr,'transp', cod(nr)) VIA Route_14
Assign_enc(k–1, nr,'transp') TO ENCPID (cod(nr))
ic(k):=0, typ(k):='transp', typ(k–1):= 'transp'
ic(k–1):=nr, sat(nr):=k–1, req_in_queue (nr):=0
Pop3 (pr3count,more)
Bit_bank_ handling
(More)
(true)
(false)
Messages_ in_queue
No_messages_ in_queue
prev:=True
(typ(k))
('data')
else
reins:=True
32
/* search for a new connection */
Return2 :=True
typ(k):='transp', tk:=k, tnr:=nr
k:=k–1, nr:=ic(k), new:=0
40
/* reassign a data connection */
Return2 :=True
typ(k):='transp', tk:=k, tnr:=nr
k:=k–1, nr:=ic(k)
41
/* reassign a voice connection */
Assign(k–1, nr,'transp', cod(nr)) TO SBCPID
Assign(k–1, nr,'transp'), (cod(nr)) VIA Route_14
Assign_enc(k–1, nr,'transp') TO ENCPID (cod(nr))
typ(k–1):= 'transp', typ(k):='transp' ic(k–1):=nr
sat(nr):=k–1 req_in_queue (nr):=0
Pop3 (pr3count,more)
Bit_bank_ handling
(More)
(false)
(true)
No_messages_ in_queue
Messages_ in_queue
/* reload old variables */
43
k:=tk, nr:=tnr
Wait_for_ next
45
T1510210-92

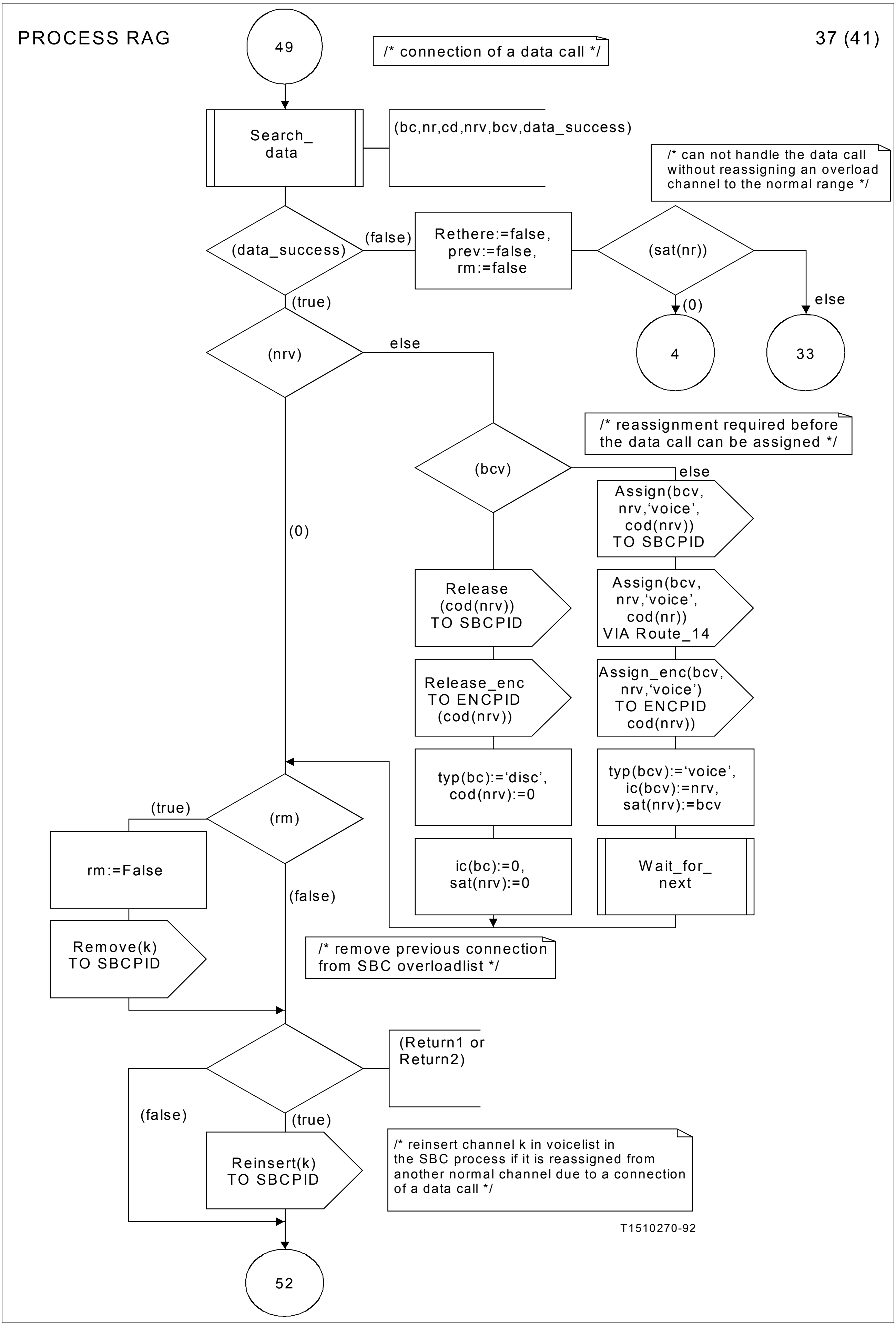

25
/* data request handling */
Check_for_ additional_ bitbank
(new, dav)
Read4 (nr)
Count (nt,nd,nb,nv)
/* not connected */
/* connected to a SC */
(sat(nr))
(0)
else
(nv)
k:=sat(nr)
(0)
else
d:=float (no*4–nt*8– (nd+nb+new+1)*4)
d:=float (no*4–nt*8– (nd+nb+new+1)*4) /float(nv)
(d>=0)
(false)
(d>=float(s))
(false)
(sat(nr)>b)
40
(true)
(true)
(true)
(false)
47
4
47
4
46
/* refresh */
/* refresh */
/* connected to a normal channel */
/* connect data call */
/* connect data call */
27
/* connected to an overload channel */
T1510220-92

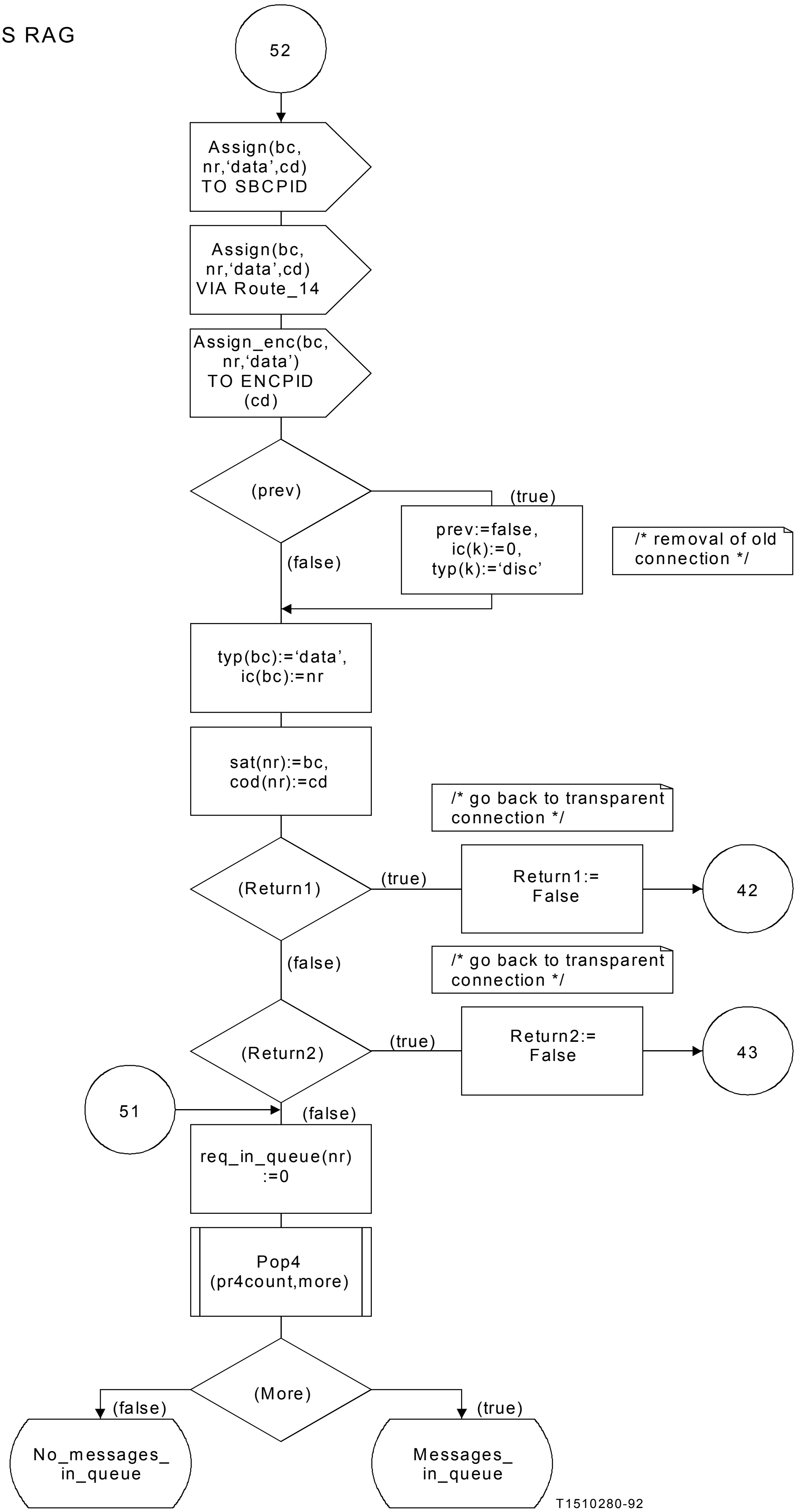

27
/* handling of a data request previously connected as an overload channel */
d:=float (no*4–nt*8–(nd+nb+new+1)*4) /float(nv)
(d>=float(s))
(false)
(true)
prev:=True, rm:=True
40
33
/* search for connection */
/* disconnect */
T1510230-92

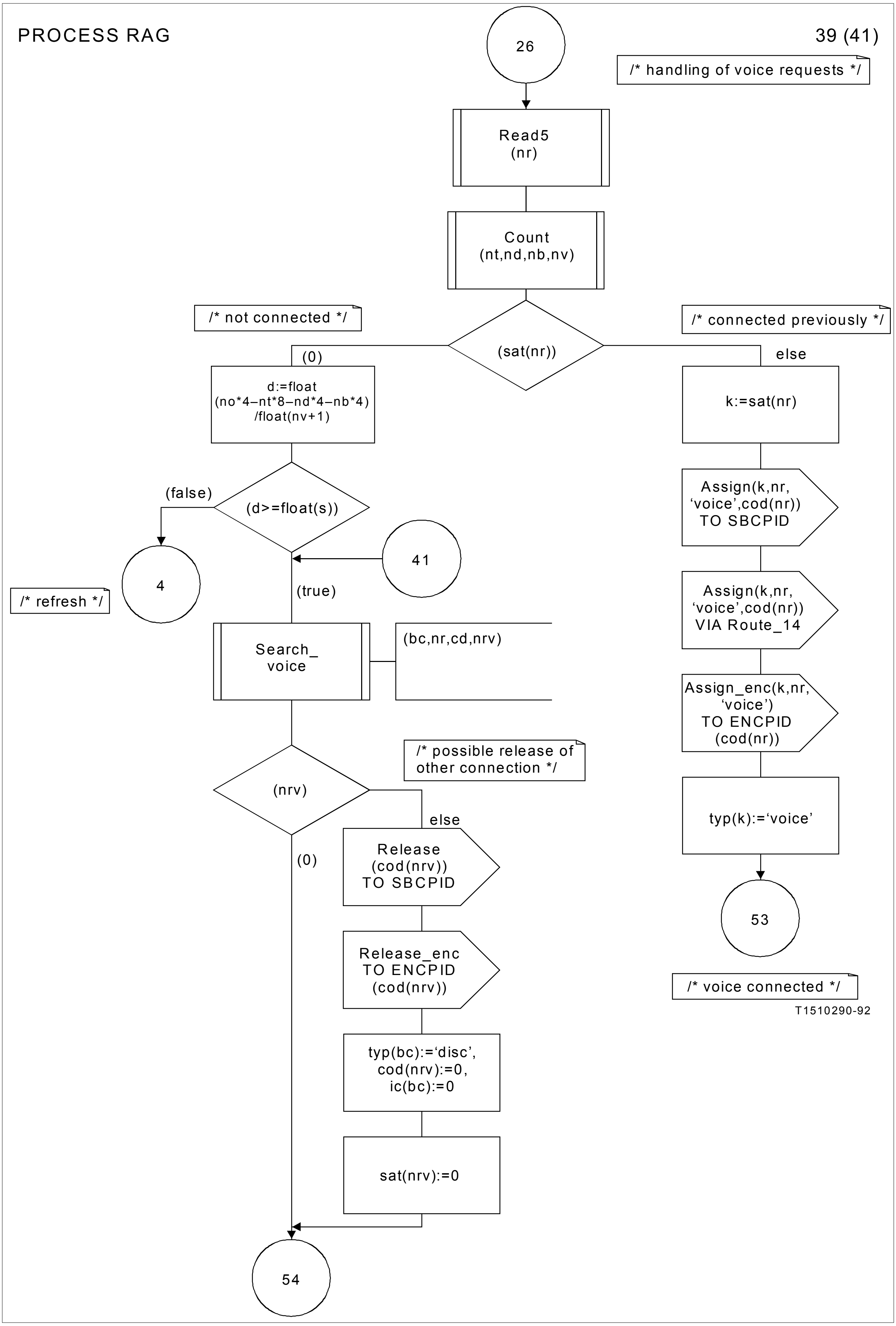

PROCESS RAG
34 (41)
46
/* connected as a normal channel */
(nv>1)
(false)
(true)
d:=float (no*4–nt*8–(nd+nb+new+1)*4
d:=float (no*4–nt*8–(nd+nb+new+1)*4 /float(nv)
(d>0)
(false)
33
/* disconnect */
(true)
(d>=float(s))
(false)
33
/* disconnect */
(dav)
(1)
Release (cod(nr)) TO SBCPID
Release_enc TO ENCPID (cod(nr))
typ(k):='disc', ic(k):=0, sat(nr):=0
cod(nr):=0
49
(0)
(new)
(1)
Rethere :=True
47
/* allocation of a new bitbank needed */
(else)
Check_overload_ reassignment_ when_data
(data_success)
/* bitbank allocated */
50
Rethere :=False
(data_success)
(false)
33
/* disconnect */
(true)
Assign(k,nr, 'data',cod(nr)) TO SBCPID
Assign(k,nr, 'data',cod(nr)) VIA Route_14
Assign_enc(k,nr, 'data') TO ENCPID (cod(nr))
typ(k):='data'
51
/* data call connected */
T1510240-92

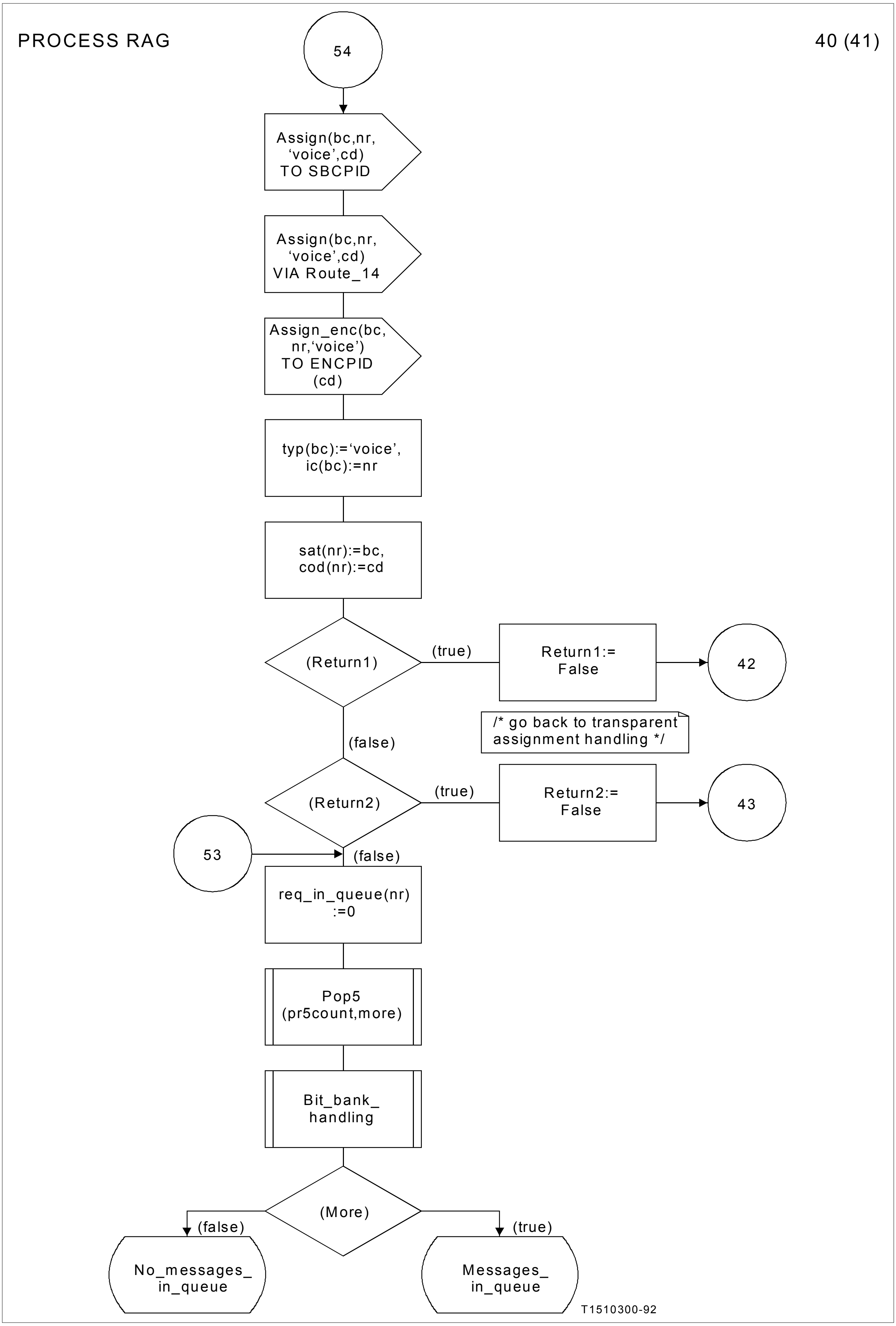

PROCESS RAG
35 (41)
47
(new)
/* connection of new bitbank and possible reassignment to overload of detected channel */
(0)
else
49
Search_ data
(bc,nr,cd,nrv,bcv, data_success)
/* Can not handle the data call without reassigning an overload channel to the normal range */
/* no bitbank required */
(data_success)
(false)
Rethere:=false, prev:=false, rm:=false
(sat(nr))
(true)
(0)
else
4
33
(nrv)
else
(bcv)
else
(0)
Assign(bcv, nrv,'voice', cod(nrv)) TO SBCPID
(0)
Release (cod(nrv)) TO SBCPID
Assign(bcv, nrv,'voice', cod(nv)) VIA Route_14
Release_enc TO ENCPID (cod(nrv))
Assign_enc(bcv, nrv,'voice') TO ENCPID cod(nrv))
Assign(bc, 250,'bank',0) TO SBCPID
typ(bc):='disc', cod(nrv):=0
typ(bcv):='voice', ic(bcv):=nrv, sat(nrv):=bcv
Assign(bc, 250,'voice',0) VIA Route_14
ic(bc):=0, sat(nrv):=0
Wait_for_ next
T1510250-92
typ(bc):='bank', ic(bc):=0
Make_room_ in_bitbank_ array (nw,bc,bitbank)
btot:=btot+1. bitbank(nw) :=bc
48

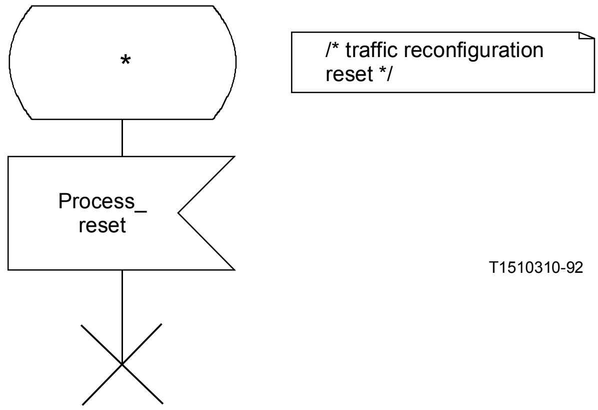

48
Wait_for_
next
(Rethere)
(false)
(true)
49
50
/* no previous connection,
search for possible place
to allocate data call */
/* previous connection to a
normal channel */
T1510260-92

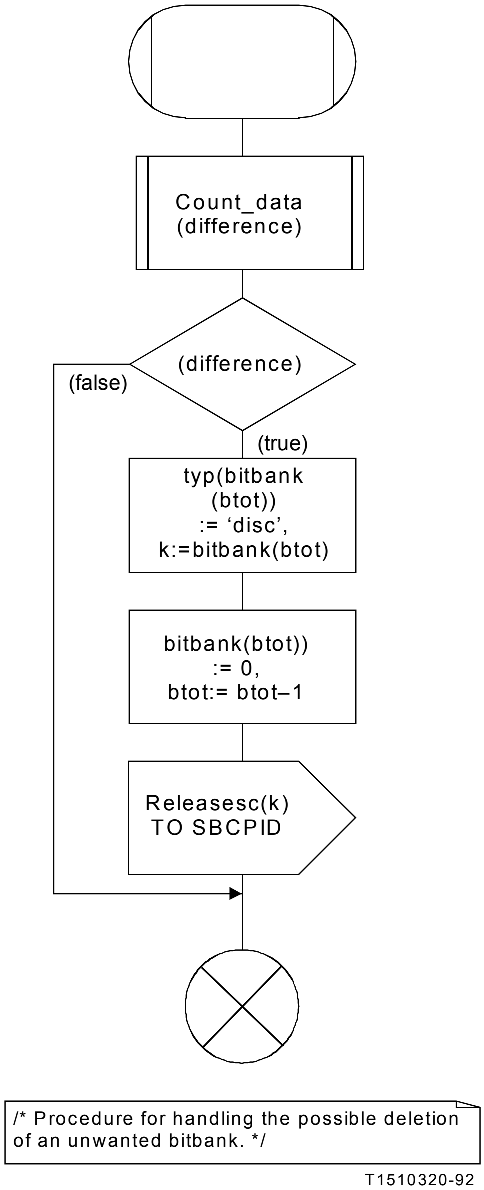

PROCESS RAG
37 (41)
49
/* connection of a data call */
Search_ data
(bc,nr,cd,nrv,bcv,data_success)
/* can not handle the data call without reassigning an overload channel to the normal range */
(data_success)
(false)
Rethere:=false, prev:=false, rm:=false
(sat(nr))
(0)
else
4
33
(true)
(nrv)
else
/* reassignment required before the data call can be assigned */
(bcv)
else
Assign(bcv, nrv,'voice', cod(nrv)) TO SBCPID
(0)
Release (cod(nrv)) TO SBCPID
Assign(bcv, nrv,'voice', cod(nr)) VIA Route_14
Release_enc TO ENCPID (cod(nrv))
Assign_enc(bcv, nrv,'voice') TO ENCPID cod(nrv))
typ(bc):='disc', cod(nrv):=0
typ(bcv):='voice', ic(bcv):=nrv, sat(nrv):=bcv
(true)
(rm)
rm:=False
ic(bc):=0, sat(nrv):=0
Wait_for_ next
(false)
/* remove previous connection from SBC overloadlist */
Remove(k) TO SBCPID
(Return1 or Return2)
(false)
(true)
Reinsert(k) TO SBCPID
/* reinsert channel k in voicelist in the SBC process if it is reassigned from another normal channel due to a connection of a data call */
T1510270-92
52

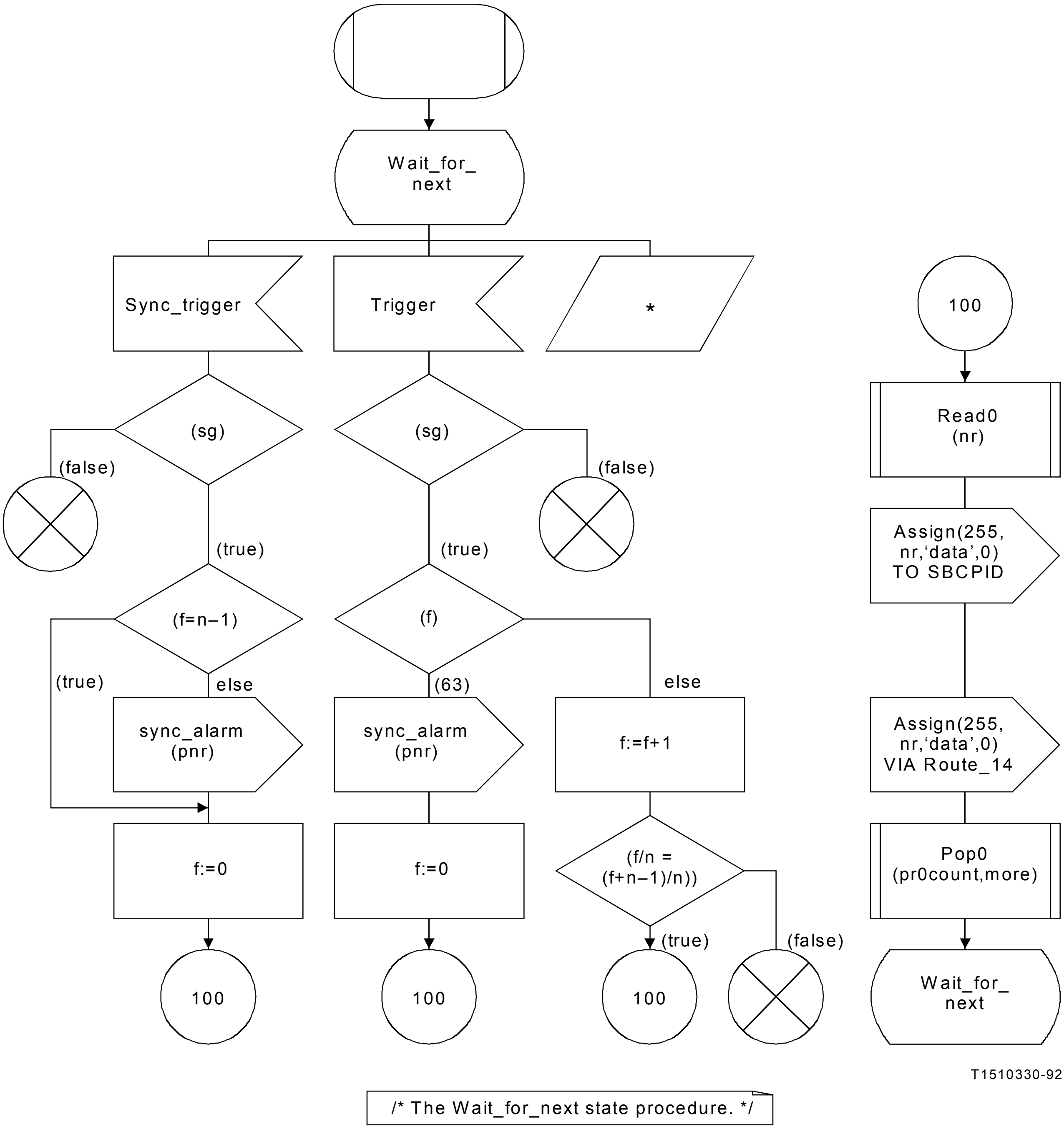

52
Assign(bc, nr,'data',cd) TO SBCPID
Assign(bc, nr,'data',cd) VIA Route_14
Assign_enc(bc, nr,'data') TO ENCPID (cd)
(prev)
(true)
prev:=false, ic(k):=0, typ(k):='disc'
/* removal of old connection */
(false)
typ(bc):='data', ic(bc):=nr
sat(nr):=bc, cod(nr):=cd
/* go back to transparent connection */
(Return1)
(true)
Return1:= False
42
(false)
/* go back to transparent connection */
(Return2)
(true)
Return2:= False
43
51
(false)
req_in_queue(nr) :=0
Pop4 (pr4count,more)
(More)
(false)
(true)
No_messages_ in_queue
Messages_ in_queue
T1510280-92

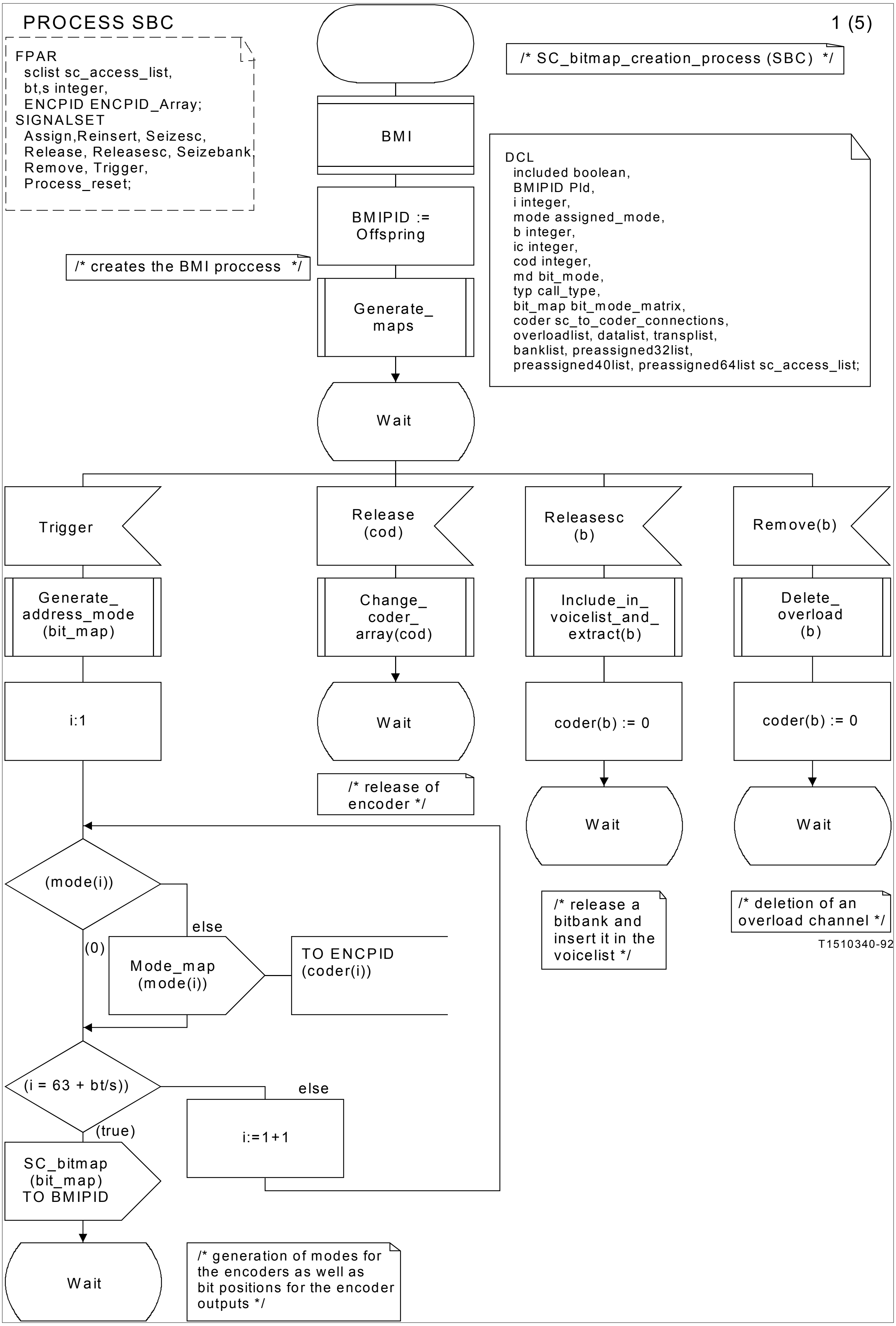

26
/* handling of voice requests */
Read5
(nr)
Count
(nt,nd,nb,nv)
/* not connected */
(sat(nr))
/* connected previously */
(0)
else
d:=float
(no*4–nt*8–nd*4–nb*4)
/float(nv+1)
k:=sat(nr)
(false)
(d>=float(s))
Assign(k,nr,
'voice',cod(nr))
TO SBCPID
41
4
/* refresh */
(true)
Assign(k,nr,
'voice',cod(nr))
VIA Route_14
Search_
voice
(bc,nr,cd,nrv)
Assign_enc(k,nr,
'voice')
TO ENCPID
(cod(nr))
/* possible release of
other connection */
(nrv)
typ(k):='voice'
else
(0)
Release
(cod(nrv))
TO SBCPID
53
Release_enc
TO ENCPID
(cod(nrv))
/* voice connected */
T1510290-92
typ(bc):='disc',
cod(nrv):=0,
ic(bc):=0
sat(nrv):=0
54

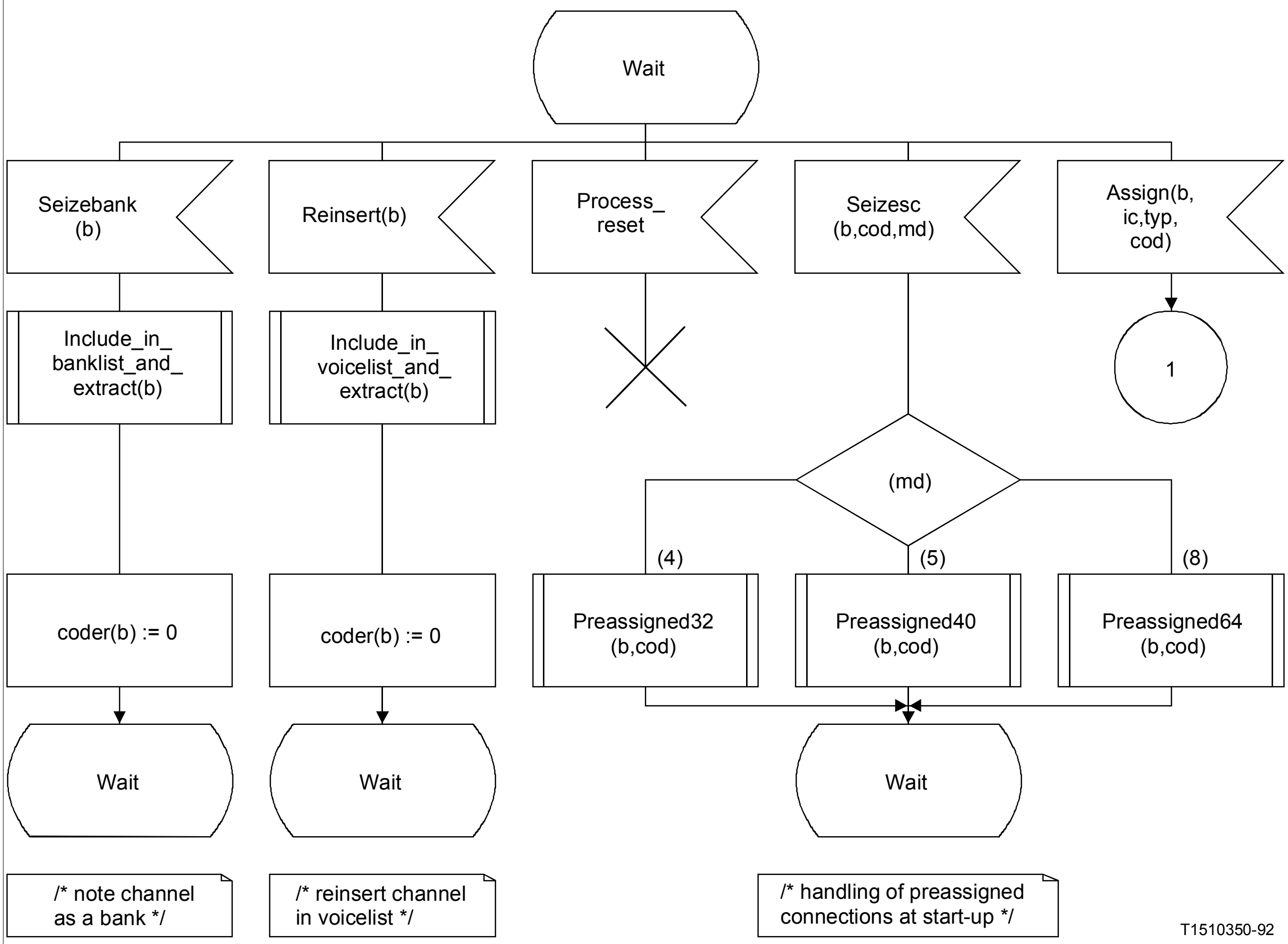

PROCESS RAG
40 (41)
54
Assign(bc,nr,
'voice',cd)
TO SBCPID
Assign(bc,nr,
'voice',cd)
VIA Route_14
Assign_enc(bc,
nr,'voice')
TO ENCPID
(cd)
typ(bc):='voice',
ic(bc):=nr
sat(nr):=bc,
cod(nr):=cd
(Return1)
(true)
Return1:=
False
42
/* go back to transparent
assignment handling */
(false)
(Return2)
(true)
Return2:=
False
43
53
(false)
req_in_queue(nr)
:=0
Pop5
(pr5count,more)
Bit_bank_
handling
(More)
(false)
(true)
No_messages_
in_queue
Messages_
in_queue
T1510300-92


*
/* traffic reconfiguration reset */
Process_ reset
T1510310-92

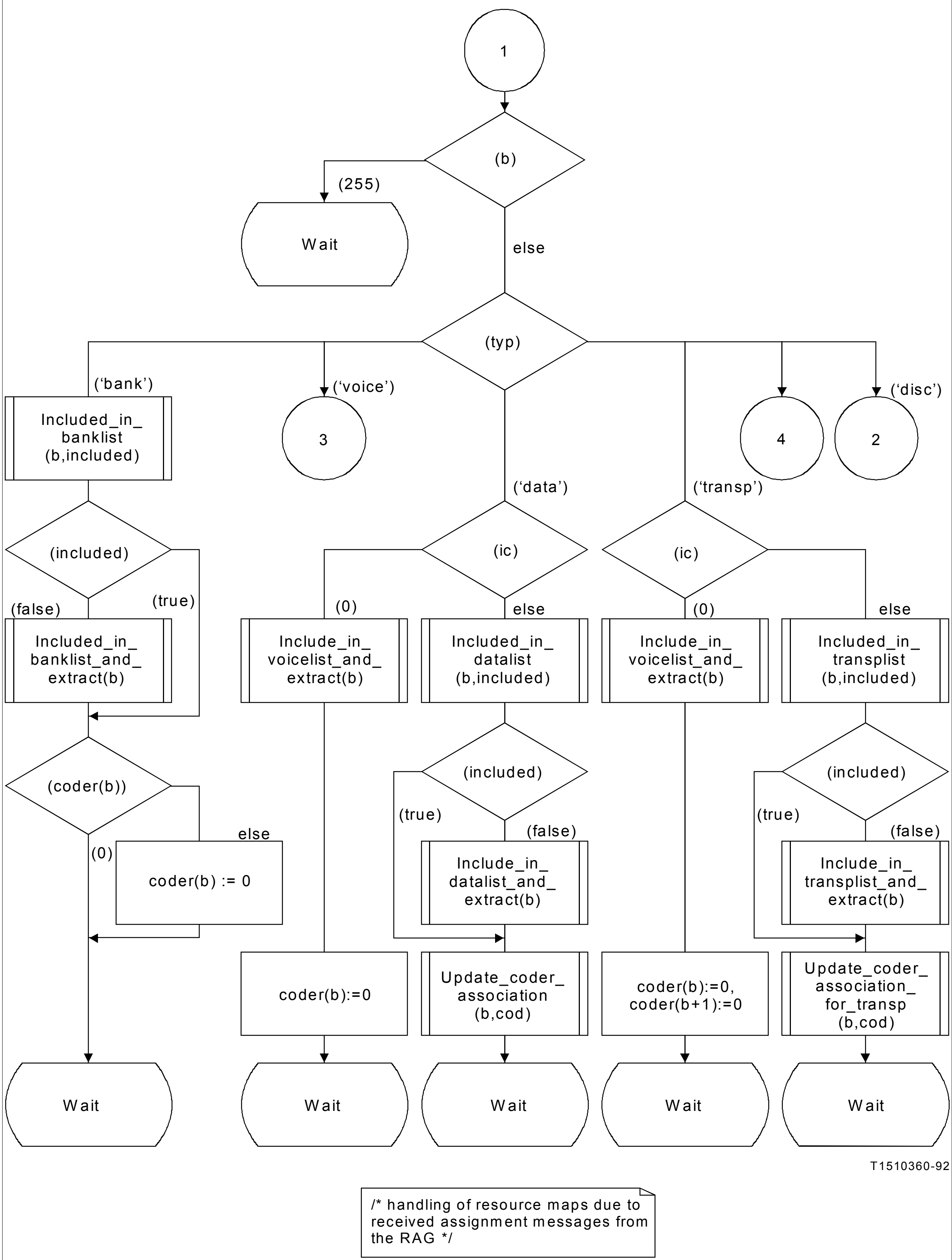

Count_data
(difference)
(difference)
(false)
(true)
typ(bitbank
(btot))
:= 'disc',
k:=bitbank(btot)
bitbank(btot))
:= 0,
btot:= btot–1
Releasesc(k)
TO SBCPID
/* Procedure for handling the possible deletion
of an unwanted bitbank. */
T1510320-92


Wait_for_
next
Sync_trigger
Trigger
*
100
(sg)
(false)
(sg)
(false)
Read0
(nr)
(true)
(true)
Assign(255,
nr,‘data’,0)
TO SBCPID
(f=n–1)
(f)
(true)
else
(63)
else
sync_alarm
(pnr)
sync_alarm
(pnr)
f:=f+1
Assign(255,
nr,‘data’,0)
VIA Route_14
f:=0
f:=0
(f/n =
(f+n–1)/n))
Pop0
(pr0count,more)
(true)
(false)
100
100
100
Wait_for_
next
T1510330-92
/* The Wait_for_next state procedure. */

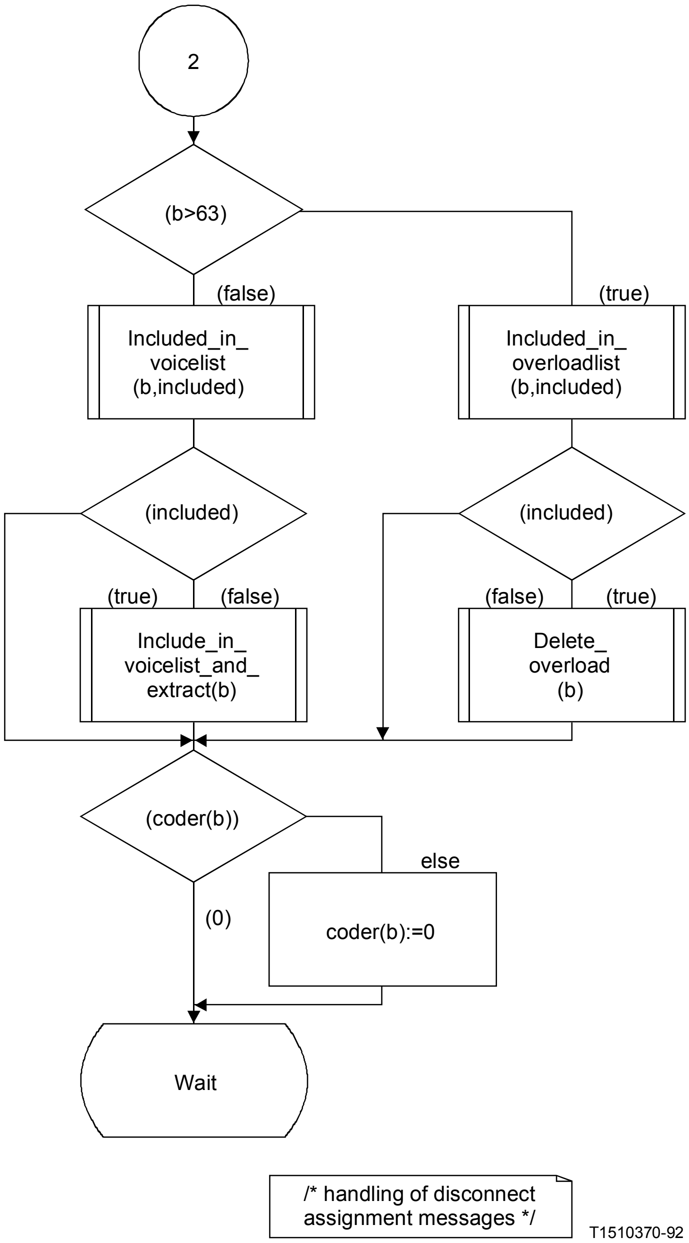

PROCESS SBC
1 (5)
FPAR
sclist sc_access_list,
bt,s integer,
ENCPID ENCPID_Array;
SIGNALSET
Assign,Reinsert, Seizesc,
Release, Releasesc, Seizebank,
Remove, Trigger,
Process_reset;
/* SC_bitmap_creation_process (SBC) */
BMI
BMIPID := Offspring
/* creates the BMI proccess */
Generate_ maps
DCL
included boolean,
BMIPID PId,
i integer,
mode assigned_mode,
b integer,
ic integer,
cod integer,
md bit_mode,
typ call_type,
bit_map bit_mode_matrix,
coder sc_to_coder_connections,
overloadlist, datalist, transplist,
banklist, preassigned32list,
preassigned40list, preassigned64list sc_access_list;
Wait
Trigger
Release (cod)
Releasesc (b)
Remove(b)
Generate_ address_mode (bit_map)
Change_ coder_ array(cod)
Include_in_ voicelist_and_ extract(b)
Delete_ overload (b)
i:1
Wait
coder(b) := 0
coder(b) := 0
/* release of encoder */
Wait
Wait
(mode(i))
else
(0)
Mode_map (mode(i))
TO ENCPID (coder(i))
/* release a bitbank and insert it in the voicelist */
/* deletion of an overload channel */
T1510340-92
(i = 63 + bt/s))
else
(true)
i:=1+1
SC_bitmap (bit_map) TO BMIPID
Wait
/* generation of modes for the encoders as well as bit positions for the encoder outputs */


Wait
Seizebank (b)
Reinsert(b)
Process_ reset
Seizesc (b,cod,md)
Assign(b, ic,typ, cod)
Include_in_ banklist_and_ extract(b)
Include_in_ voicelist_and_ extract(b)
1
(md)
(4)
(5)
(8)
coder(b) := 0
coder(b) := 0
Preassigned32 (b,cod)
Preassigned40 (b,cod)
Preassigned64 (b,cod)
Wait
Wait
Wait
/* note channel as a bank */
/* reinsert channel in voicelist */
/* handling of preassigned connections at start-up */
T1510350-92

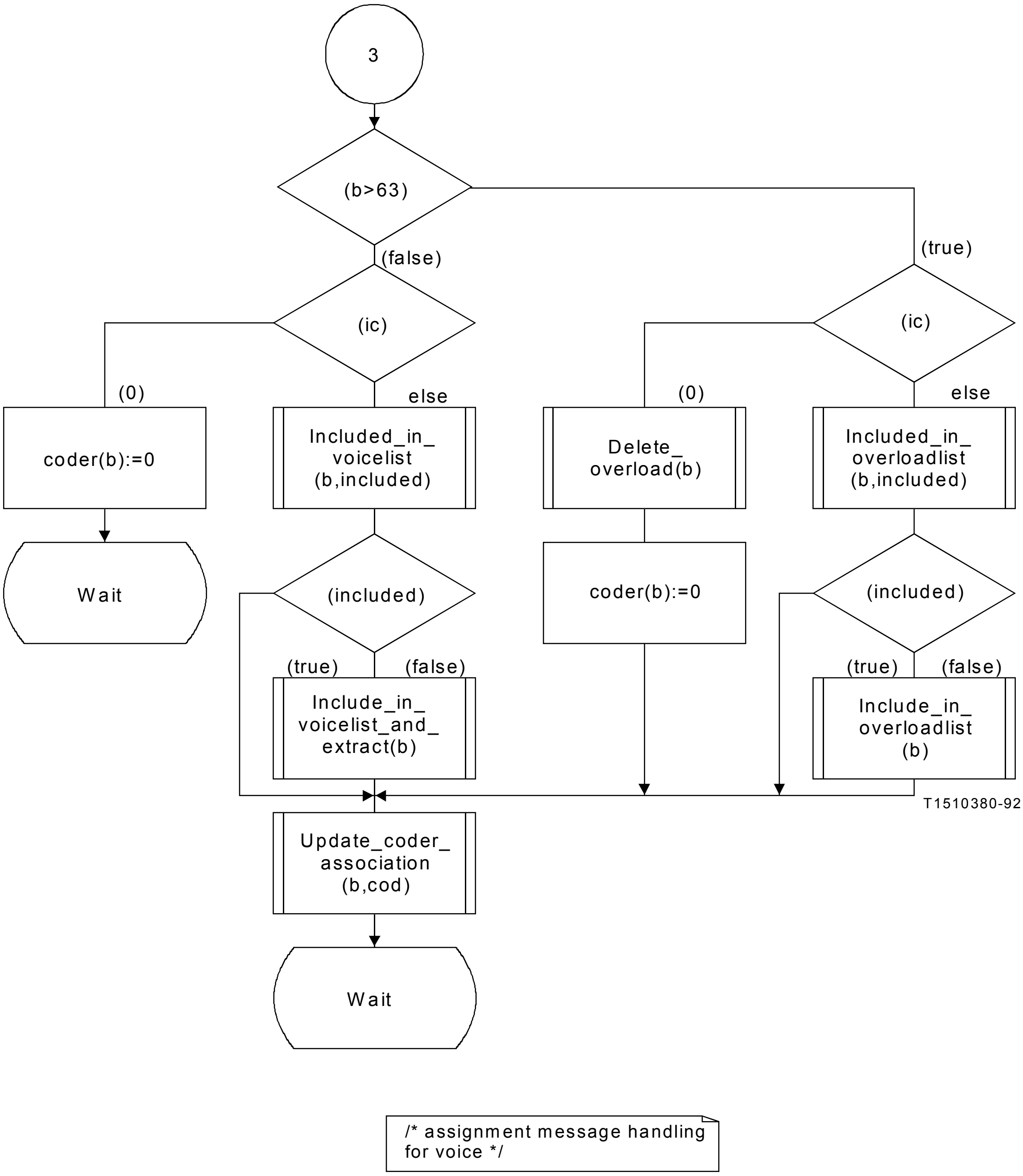
PROCESS SBC



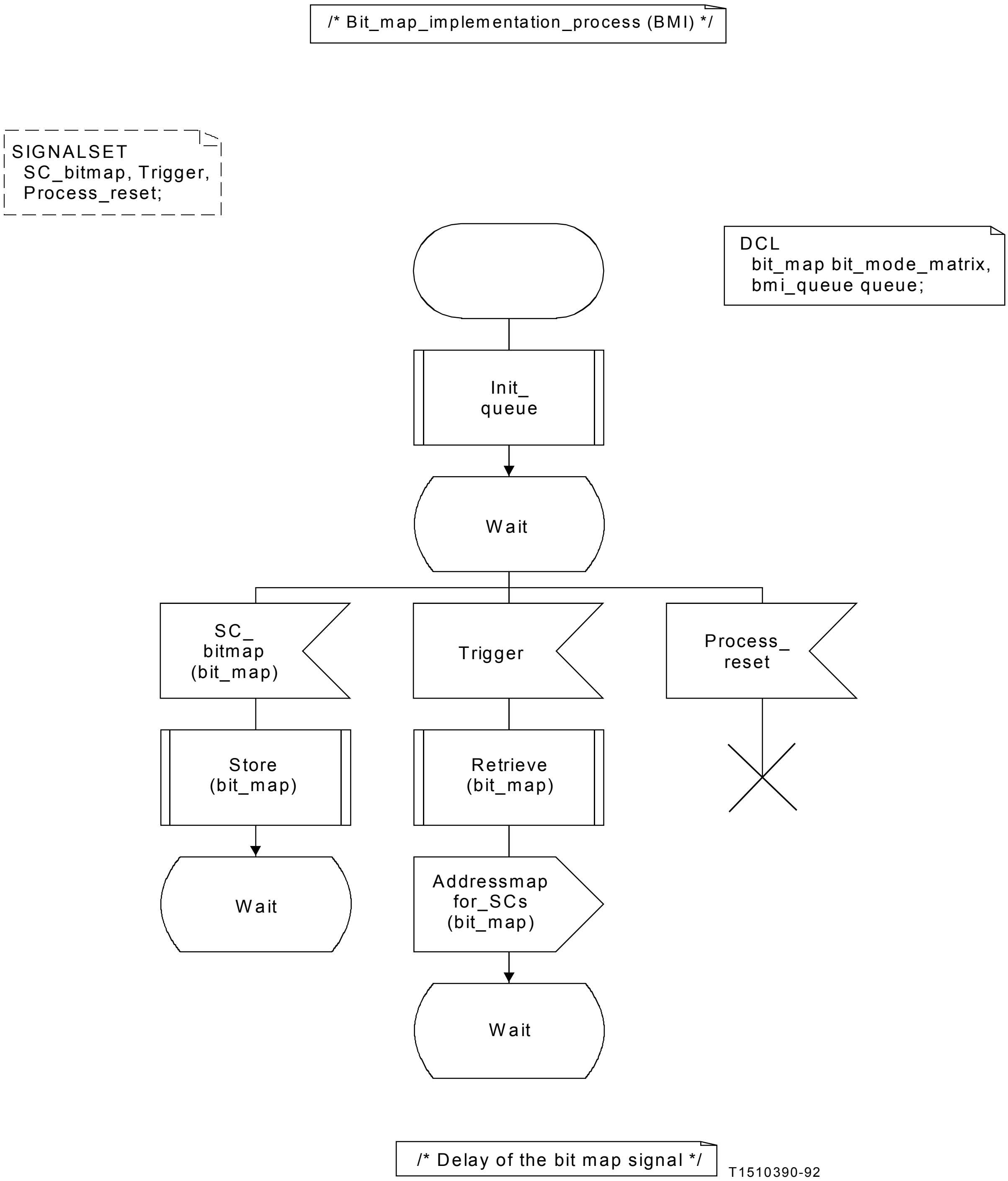
1
(b)
(255)
Wait
else
(typ)
('bank')
('voice')
('disc')
Included_in_ banklist (b,included)
3
4
2
('data')
('transp')
(included)
(ic)
(ic)
(false)
(true)
(0)
else
(0)
else
Included_in_ banklist_and_ extract(b)
Include_in_ voicelist_and_ extract(b)
Included_in_ datalist (b,included)
Include_in_ voicelist_and_ extract(b)
Included_in_ transplist (b,included)
(coder(b))
(included)
(included)
(true)
(true)
else
(false)
(false)
(0)
coder(b) := 0
Include_in_ datalist_and_ extract(b)
Include_in_ transplist_and_ extract(b)
coder(b):=0
Update_coder_ association (b,cod)
coder(b):=0, coder(b+1):=0
Update_coder_ association_ for_transp (b,cod)
Wait
Wait
Wait
Wait
Wait
T1510360-92
/* handling of resource maps due to received assignment messages from the RAG */


2
(b>63)
(false)
(true)
Included_in_
voicelist
(b,included)
Included_in_
overloadlist
(b,included)
(included)
(included)
(true)
(false)
(false)
(true)
Include_in_
voicelist_and_
extract(b)
Delete_
overload
(b)
(coder(b))
else
(0)
coder(b):=0
Wait
/* handling of disconnect
assignment messages */

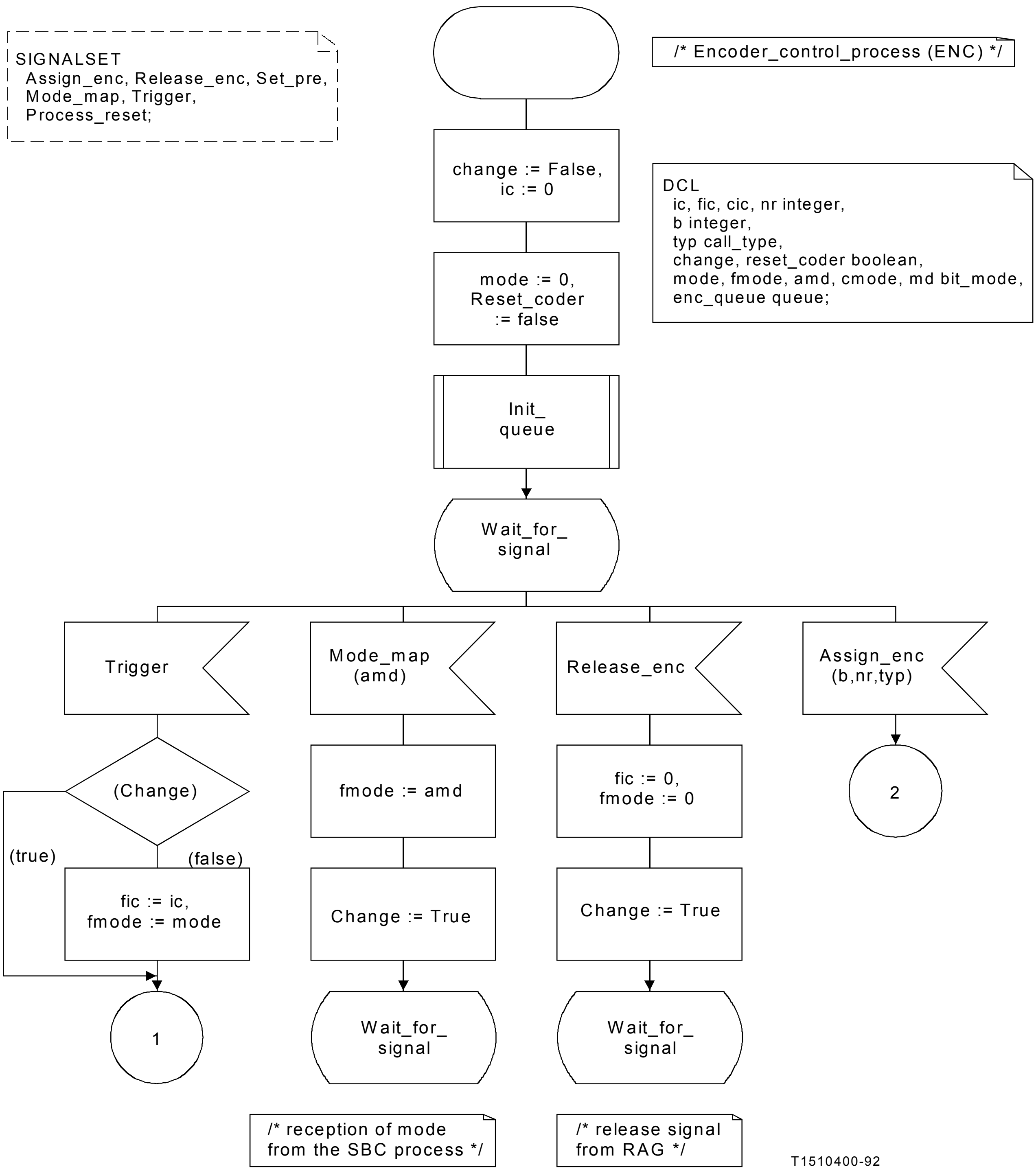
T1510370-92


3
(b>63)
(false)
(true)
(ic)
(ic)
(0)
else
(0)
else
coder(b):=0
Included_in_ voicelist (b,included)
Delete_ overload(b)
Included_in_ overloadlist (b,included)
Wait
(included)
coder(b):=0
(included)
(true)
(false)
(true)
(false)
Include_in_ voicelist_and_ extract(b)
Include_in_ overloadlist (b)
T1510380-92
Update_coder_ association (b,cod)
Wait
/* assignment message handling for voice */

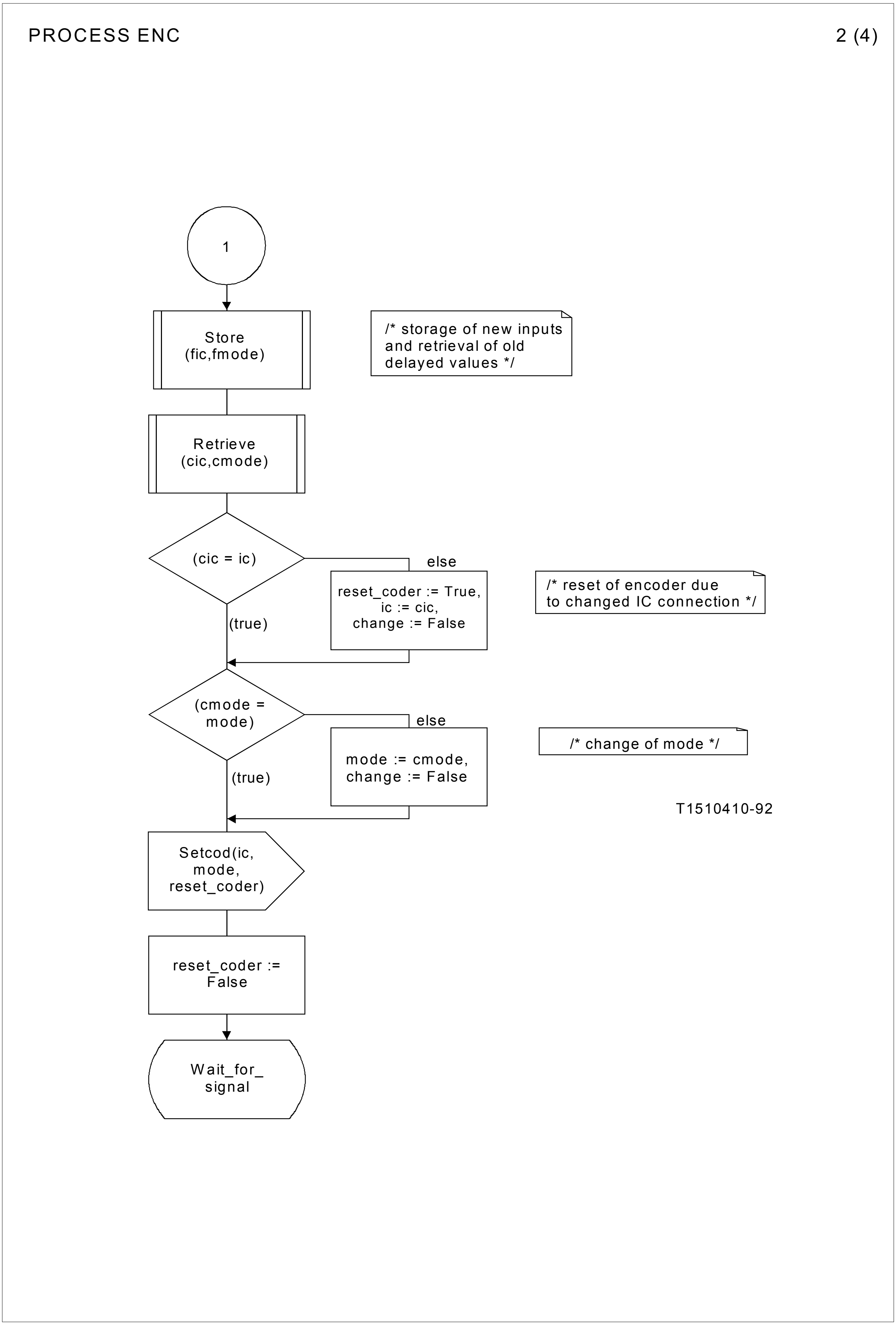
PROCESS BMI

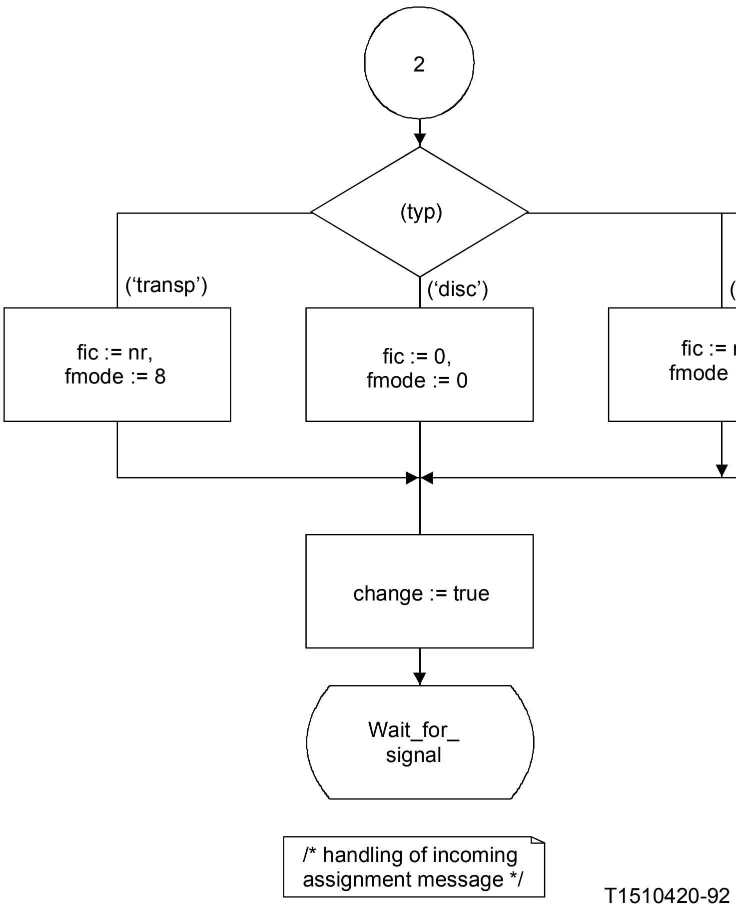

/* Bit_map_implementation_process (BMI) */
SIGNALSET
SC_bitmap, Trigger,
Process_reset;
DCL
bit_map bit_mode_matrix,
bmi_queue queue;
Init_
queue
Wait
SC_
bitmap
(bit_map)
Trigger
Process_
reset
Store
(bit_map)
Retrieve
(bit_map)
Wait
Addressmap
for_SCs
(bit_map)
Wait
/* Delay of the bit map signal */
T1510390-92

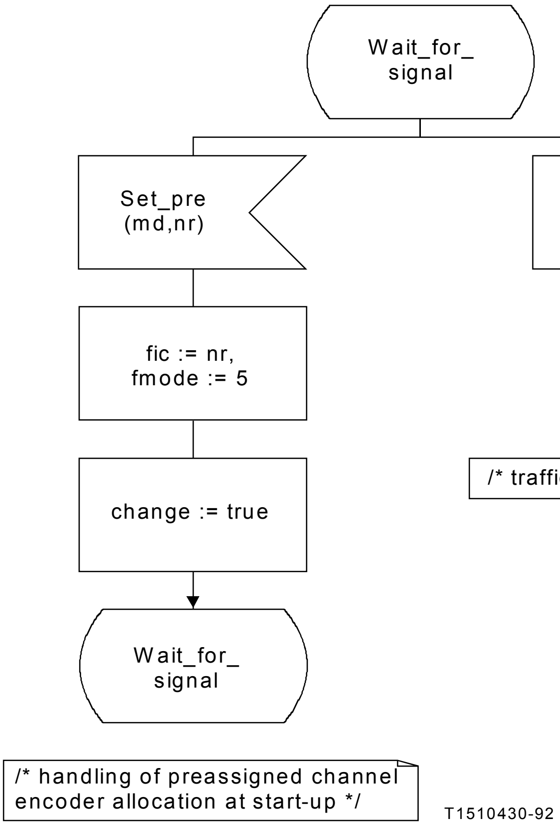
PROCESS ENC

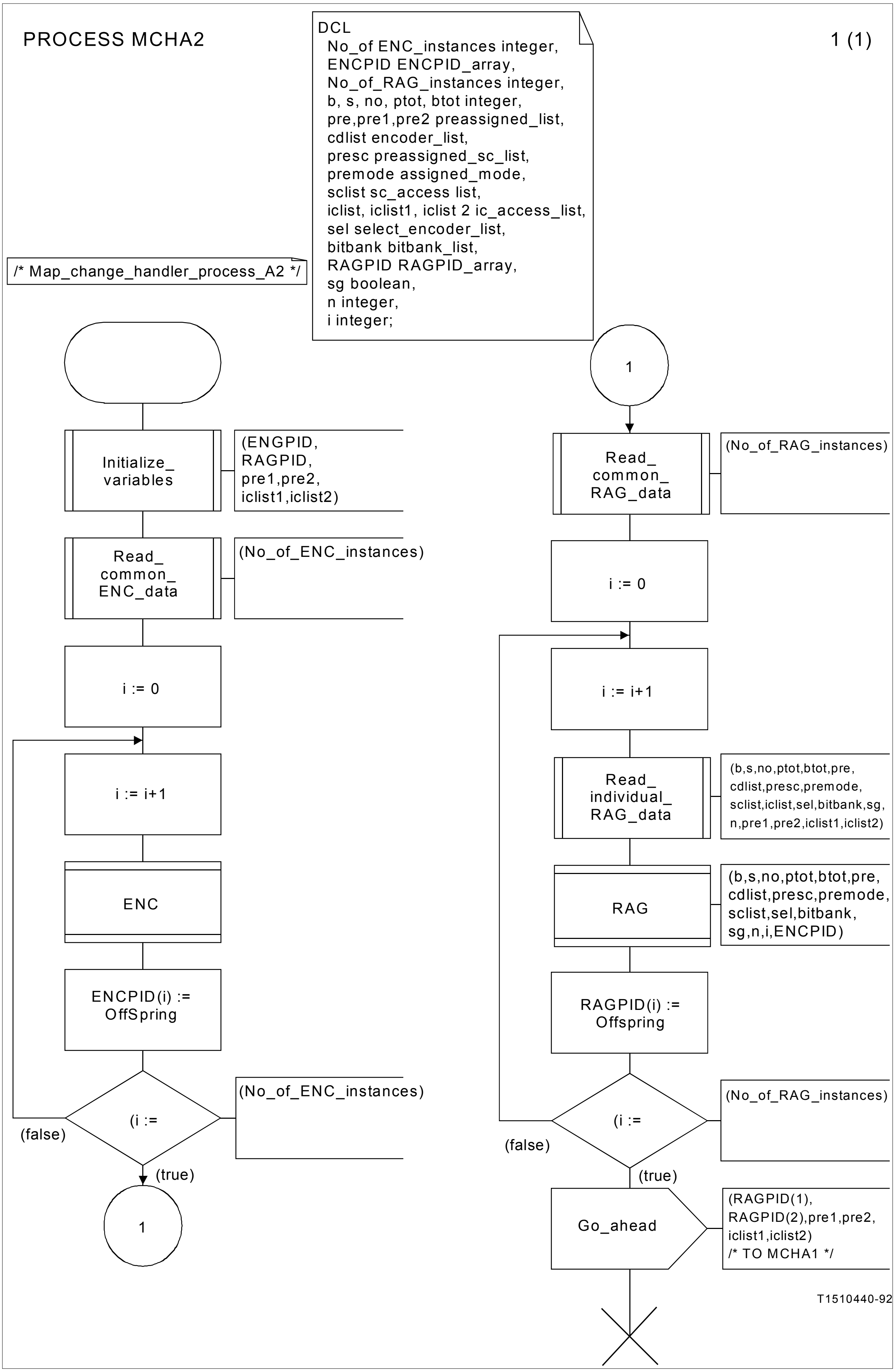


SIGNALSET
Assign_enc, Release_enc, Set_pre,
Mode_map, Trigger,
Process_reset;
/* Encoder_control_process (ENC) */
change := False,
ic := 0
DCL
ic, fic, cic, nr integer,
b integer,
typ call_type,
change, reset_coder boolean,
mode, fmode, amd, cmode, md bit_mode,
enc_queue queue;
mode := 0,
Reset_coder
:= false
Init_
queue
Wait_for_
signal
Trigger
Mode_map
(amd)
Release_enc
Assign_enc
(b,nr,typ)
(Change)
(true)
(false)
fic := ic,
fmode := mode
1
fmode := amd
Change := True
Wait_for_
signal
/* reception of mode
from the SBC process */
fic := 0,
fmode := 0
Change := True
Wait_for_
signal
/* release signal
from RAG */
2
T1510400-92


PROCESS ENC
2 (4)
1
Store
(fic,fmode)
/* storage of new inputs
and retrieval of old
delayed values */
Retrieve
(cic,cmode)
(cic = ic)
else
reset_coder := True,
ic := cic,
change := False
/* reset of encoder due
to changed IC connection */
(true)
(cmode =
mode)
else
mode := cmode,
change := False
/* change of mode */
(true)
T1510410-92
Setcod(ic,
mode,
reset_coder)
reset_coder :=
False
Wait_for_
signal


2
(typ)
('transp')
('disc')
('data')
else
fic := nr,
fmode := 8
fic := 0,
fmode := 0
fic := nr,
fmode := 5
fic := nr
change := true
Wait_for_
signal
/* handling of incoming
assignment message */
T1510420-92


Wait_for_
signal
Set_pre
(md,nr)
Process_
reset
fic := nr,
fmode := 5
change := true
/* traffic reconfiguration reset */
Wait_for_
signal
/* handling of preassigned channel
encoder allocation at start-up */
T1510430-92


PROCESS MCHA2
DCL
No_of ENC_instances integer,
ENCPID ENCPID_array,
No_of_RAG_instances integer,
b, s, no, ptot, btot integer,
pre,pre1,pre2 preassigned_list,
cdlist encoder_list,
presc preassigned_sc_list,
premode assigned_mode,
sclist sc_access list,
iclist, iclist1, iclist 2 ic_access_list,
sel select_encoder_list,
bitbank bitbank_list,
RAGPID RAGPID_array,
sg boolean,
n integer,
i integer;
1 (1)
/* Map_change_handler_process_A2 */
Initialize_ variables
(ENGPID, RAGPID, pre1,pre2, iclist1,iclist2)
Read_ common_ ENC_data
(No_of_ENC_instances)
i := 0
i := i+1
ENC
ENCPID(i) := OffSpring
(i :=
(No_of_ENC_instances)
(false)
(true)
1
1
Read_ common_ RAG_data
(No_of_RAG_instances)
i := 0
i := i+1
Read_ individual_ RAG_data
(b,s,no,ptot,btot,pre, cdlist,presc,premode, sclist,iclist,sel,bitbank,sg, n,pre1,pre2,iclist1,iclist2)
RAG
(b,s,no,ptot,btot,pre, cdlist,presc,premode, sclist,sel,bitbank, sg,n,i,ENCPID)
RAGPID(i) := Offspring
(i :=
(No_of_RAG_instances)
(false)
(true)
Go_ahead
(RAGPID(1), RAGPID(2),pre1,pre2, iclist1,iclist2) /* TO MCHA1 */
T1510440-92

A.3.2 *Logic diagrams for DCME receive side*

The logic diagrams in this section of Annex A.3 are supplemental to the description of the DCME receive side structure given in Annex A.2. The receive side handling procedures are contained within one block:

RCP: Receive channel processing block.

A.3.2.1 *The RCP block*

The RCP block contains three different processes and the following signals:

L4: Rxdata (Integer);

L51: Assign (Integer, Integer, Call-type);

L52: Rxtranspreq (Integer), Rxtransprel (Integer);

L53: SC-bitmap (Bit-mode-matrix);

L54: Seize (Integer, Integer), Release,
Seizev (Integer), Mode-map (Integer);

L55: Trigger;

L56: Setcod (Integer, Integer, Boolean);

L57: Addressmap-for-SCs (Bit-mode-matrix);

L58: Process-reset;

L59: Process-Reset;

L60: Trigger;

L61: Process-reset.

The states used by the processes are the following:

a) *(RUD) Rx status update and overload channel decoding process (0,4)*

Wait

b) *(BMI) Bit map implementation process (0,4)*

Wait

c) *(DEC) Decoder control process (0,)*

Wait-for-signal.

The signals have the following meaning:

L4 – Rxdata (Integer) – This signal is sent to the transmit side assignment processes upon reception of an assignment message destined for the terminal, which marks a transition from a different state to a data call.

L51 – Assign (Integer, Integer, Call-type) – This signal contains the information included within the assignment message delivered to the process. The variables contain the following information:

SC number, IC number and Call-type.

The last variable can contain three different possibilities:

"Voice", "Data", "Transp".

Within the resource maps of the RUD process, it is possible to have the following different Call-type values:

"Disc", "Voice", "Data", "Transp", "Bank".

L52 – Rxtranspreq (Integer) – A signal given to the 64 kbit/s transparent circuit handler (TCH) upon reception of an assignment message, destined for the terminal, marking a transition from a different state to a transparent call. The Integer value refers to the local IC.

L52 – Rxtransprel (Integer) – The reverse of the above. It is sent when a transition occurs from a transparent call to a different state.

L53 – SC bitmap (Bit-mode-matrix) – A complex signal which defines the mapping between the bits of the incoming bearer and the decoder inputs.

L54 – Seize (Integer, Integer) – This signal contains the information required to connect the output of a decoder to the correct IC number in order to complete the circuit. It contains the local IC number associated with the received channel and the mode in which the decoder should be set (4/5/8 bit mode).

L54 – Seizev (Integer) – This signal is sent in order to associate the output at a decoder used for a voice connection with the correct local IC. The signal contains the local IC number to be used.

L54 – Release – This signal is used to release a designated decoder back into the decoder pool.

L54 – Mode-map (Integer) – This signals contains the mode that is to be used for a decoder that is connected to a voice channel. The parameter contains the actual mode to be used.

L55, L60 – Trigger – This signal is used to indicate that a Setcod signal, as well as an Addressmap-for-SCs signal, should be delivered to the hardware.

L56 – Setcod (Integer, Integer, Boolean) – This signal is issued to generate a specific hardware connection for a certain decoder. The signal contains the local IC number, the mode to be used and a Reset command.

L57 – Addressmap-for-SCs (Bit-mode-matrix) – This complex signal contains the bit associations required to connect the bits of an SC to a decoder.

L58, L59, L61 – Process-reset – This signal is issued by the map-change-handler in association with a map change. The reception of this signal causes the termination of the process that receives it.

A.3.2.2 *The RUD process*

The RUD process is assumed to be generated by the map-change-handler (MCH). The MCH will generate as many instances of the RUD process as is required. There will be one instance of the RUD process for each pool containing traffic destined for this unit (up to four). The RUD process will process the assignment message contained within the pool that it has been assigned to and generate the required actions based on the contents of this message. When the RUD process is created by the map-change-handler, a number of variables are passed along to the RUD process. These are:

– *dcdlist* – The list contains a list of the decoder numbers that the process may use. The list excludes those decoders used to handle pre-assigned channels.

– *bt* – The total number of 4-bit channels that the remote correspondents pool contains.

– *iclist* – This list includes all IC numbers that may be contained in the received assignment message of the received pool to which the RUD instance is assigned. Pre-assigned IC numbers are excluded.

– *sclist* – This list contains the SC numbers that may be contained in the received assignment message of the received pool to which the RUD instance is assigned. Pre-assigned SCs are excluded.

– *pre* – This array contains the IC numbers of any pre-assigned channels that the remote DCME may have in the pool to which the RUD instance is assigned.

– *presc* – This array contains the SC numbers associated with the pre-assigned channels that the remote DCME may use in the pool to which the RUD instance is assigned. Only the even numbered SCs for transparent channels are given.

– *premode* – This array contains the mode that is to be associated with each pre-assigned SC given in "presc".

– *sel* – This array contains the decoders that are to be used in association with pre-assigned channels described above.

– *ptot* – This integer value contains the total number of pre-assigned channels that are to be dealt with at start-up.

– *ad* – This array uses the remote IC number to index the local IC numbers that make up the circuit. If the remote IC is not addressed to the unit, the number contained will be zero.

– *bit bank* – This array, with a maximum of 12 entries, contains in ascending SC number order, the SC numbers that are to be used for bit bank handling. At start-up, the array will contain those SCs that are to be used for bit banks in order to handle those SCs that are to be pre-assigned 40 kbit/s channels.

– *btot* – This Integer value contains the total number of bit banks that are in use at any given time. At start-up, this value equals the number of bit banks that are required to handle the pre-assigned 40 kbit/s channels.

– *DECPID(i)* – This process identifier array gives the correct addresses to any required decoder process with a given number. It is used when addressing signals towards the DEC processes.

– *BMIPID* – This process identifier variable is used to address signals to the correct BMI process.

– *s* – This integer variable defines the lowest number of bits/samples permitted. Its value is either three, for 3-bit encoding, or two, for 2-bit encoding.

Arrays are used as a resource map for the receive process. These are:

– *sat(nr)=bc* – This array uses the remote IC number to index the SC number that the IC is connected to. The array is initialized as all zeros.

– *ic(bc)=nr* – This array uses the SC numbers to index the remote IC number to which it is connected. The array is initialized as all zeros.

– *typ(bc)="Call-type"* – This array uses the received SC numbers to index the type of connection being received on that SC. This value can be either:

  "Transp", "Data", "Voice", "Disc", "Bank".

The array is initialized as all "Disc".

– *dec(lnr)=dcd* – This array uses the local (transmit) IC number to index the decoder to which a received channel, destined for this unit, is connected. It is initialized as all zeros.

The process also uses two lists in order to be able to generate the overload bit positions. These are:

Voicelist, Overloadlist

They are handled in the same way as on the transmit side. The following variables and procedure calls are used within the RUD process.

- *i* – Counter.

- *prep* – This integer is used to delay deletion of bit banks just after a bit bank has been assigned. Normally, an attempt will be made to delete bit banks which are not required after each assignment message has been processed. This is done by the procedure Bit-bank-handling. There may, however, be a delay of one or two DCME frames after the assignment of a bit bank has occurred until the data assignment is made which required the generation of a new bit bank. This is due to the possible need to reassign a voice call before the data call can be assigned. When a bit bank is assigned, prep is set to 1. This will cause the procedure to bypass deletion of possible bit banks when it is invoked. After a maximum of two DCME frames, the deletion will once more be started, i.e., when a bit bank is created in frame i the deletion will start in frame i + 2.

- *Count data (difference)* – This procedure checks to see if there is a possibility to delete a bit bank. This is checked by comparing the amount of data calls and pre-assigned 40 kbit/s calls with the amount of bit banks in use. If there are too many bit banks, the variable "Difference" is set to TRUE, otherwise FALSE. If the procedure finds that there are too few bit banks in comparison to the number of data calls and pre-assigned 40 kbit/s calls being handled, the deletion of bit banks is stopped. This is done by setting the variable "Difference" to FALSE.

- *Difference* – A Boolean variable described above.

- *scr* – This variable contains the SC number contained within the received assignment message.

- *icr* – This variable contains the remote IC number that "scr" is to be connected to according to the received assignment message.

- *flag* – This variable contains the type of connection that is specified by the assignment message (voice, data, transp).

- *Check content (scr, icr, flag, correct)* – This procedure checks if the assignment message content is valid. If this is the case, the value TRUE is delivered in the variable "Correct", the value FALSE if this is not the case. If the message is incorrect, it shall be disregarded. The checks that are assumed to be made are the following:

  - "icr" is in the range of numbers that the remote destination may use. This implies that it is part of the "iclist".

  - "scr" is in the range of the DSI pool (including overload channels) and is not used for a pre-assigned connection, i.e., it is included within the "sclist".

  - The connection proposed does not violate any strict rules, such as connection of a transparent call to an odd numbered SC or the connection of something other than "voice" to an overload channel, i.e., a channel with a number higher than bt.

- *Correct* – A Boolean variable described above.

- *Again* – This Boolean variable is used to disconnect both SCs when a transparent disconnect message is received and the SC is declared as something else than transparent.

– *Delete overload(sc)* – This procedure removes the SC number "sc" from the overloadlist.

– *k* – The SC number that "icr" was connected to previously.

– *nr* – The IC number that was connected to "scr" previously.

– *nr1* – The IC number that was connected to "scr+1" previously.

– *Insert in voicelist (sc)* – This procedure inserts the SC number "sc" into the voicelist in its appropriate place.

– *Remove from bit bankarray (sc, btot)* – This procedure removes the SC number "sc" from the bit bank array and pushes the indexed values above "SC" down one index position. It also updates the value of btot.

– *Insert2 in voicelist (s1, s2)* – This procedure inserts the SC numbers "s1" and "s2" into their appropriate place in the voicelist.

– *Insert in overloadlist (sc)* – This procedure inserts the SC number "sc" into its appropriate place in the overloadlist.

– *Make room in bit bankarray (nw, sc, bit bank)* – This procedure delivers the index of the bit bankarray where the SC number "sc" is supposed to fit in accordance with the rule for handling this array in variable nw. Starting at the greatest index used, "k" indexing a non-zero value, all indexed values down to "nw + 1" are moved up one index, leaving a space in the array at index "nw". The entry which should have the value "SC" associated with it is thus given the variable "nw".

– *nw* – An integer variable described above.

– *Remove from voicelist (sc)* – This procedure removes the SC number "sc" from the voicelist.

– *Select decoders (dcd)* – This procedure selects an unused decoder out of the "pool" and delivers the result in the variable "dcd". An unused decoder is a decoder which is part of the dcdlist but one which is not indexed by the "dec" array at a given moment. It should be noted that this pool could consist of one decoder per local IC number should this be the way in which a manufacturer elects to implement this.

– *dcd* – An Integer variable described above.

– *Remove2 from voicelist (s1, s2)* – This procedure removes SC numbers "s1" and "s2" from the voicelist.

– *Generate addresses (bitmap, mode array)* – This procedure uses the value of icr as a pointer and generates the modes to be used by the decoders handling voice connections in accordance with the current situation regarding overload channels. It also generates the addresses required in order for bits on the incoming bearer to be mapped to the correct bit positions at the input of the decoders. It puts the contents of this information into the signals Mode-map and SC bitmap. When there are not enough bits in the bit banks, the bits in the bit banks are distributed from the lowest SC numbered data channel to the highest SC numbered data channel. The data channel(s) which cannot be accommodated by the existing bit bank channels receives a dummy fifth bit set to "0". When there is not enough bits to create all existing overload channels, the available bits are distributed from the lowest SC numbered overload channel to the highest numbered overload channel. The overload channel(s) which cannot be accommodated receives dummy bits set to "0".

– *bitmap* – A complex variable which contains the bit map generated by the procedure Generate-addresses every DCME frame.

– *mode array* – An Integer array which contains the number of bits each decoder will receive each DCME frame.

A.3.2.3 *Transitions allowed within the RUD process*

It should be noted that the logic diagrams contained for the receive side handling allow for transitions that should not occur unless assignment messages are missed. These transitions have been included in order to achieve the quickest recovery of the bearer frame after losses of assignment messages have occurred. A list is given of these impossibilities below.

1) Explicit disconnection of a channel declared as "bank".
2) Implicit disconnection of an overload channel.
3) Implicit disconnection of channels declared as "transp".
4) Connection of "icr" to "scr" where "scr" is declared as "bank".
5) Connection of "icr" to "scr" where "scr" is not connected to "nr" but is declared as "transp".
6) Connection of "icr" to "scr" where "scr+1" is declared as "bank" and "flag" is "transp".
7) Implicit changes from "transp" to something else.

The complete description of the above has resulted in a significant number of diagrams needed to describe the receive protocols.

A.3.2.4 *The DEC process*

The DEC process is created by the map-change-handler at system start-up. It contains the following variables and procedure calls:

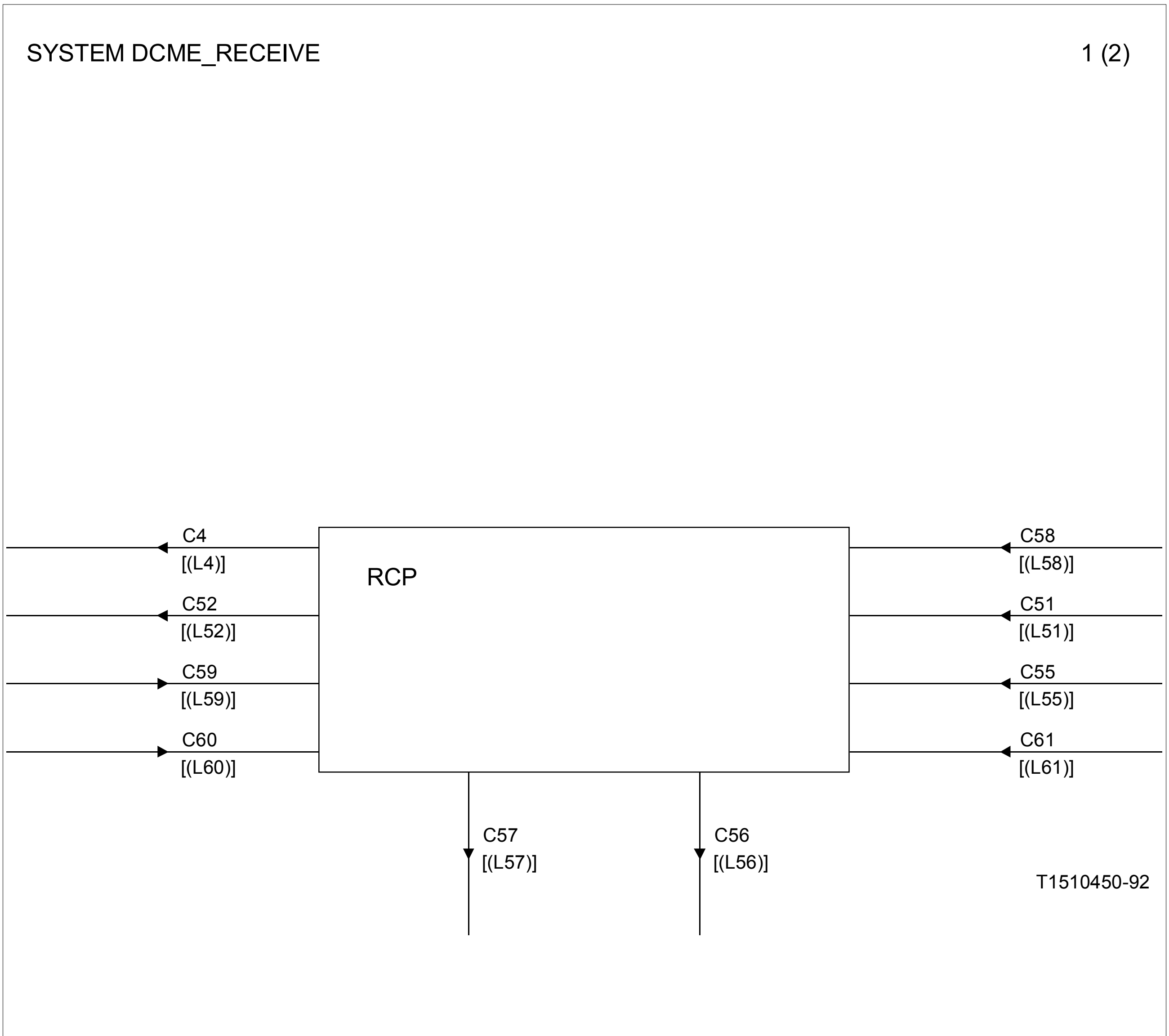
- *ic* – The local IC number to which the decoder is currently connected.
- *sc* – The SC number to which the decoder is currently connected.
- *mode* – The current mode of the decoder.
- *dec reset* – This variable stores the possibility of resetting the decoder. It is TRUE if a reset is to be made, otherwise FALSE.
- *change* – This variable is TRUE if a change in future values has occurred since the last Trigger signal, otherwise FALSE.
- *nr* – The transmit IC number that the decoder output should be given to.
- *fic* – The future IC number.
- *fmode* – The future mode.
- *cic* – The current local IC number.
- *cmode* – The current mode.
- *amd* – An Integer variable which stores the mode received within the mode-map signal.
- *md* – An Integer variable which stores the mode received within the Seize signal.
- *Store (fic, fmode)* – This procedure stores the parameters at the bottom of a queue.
- *Retrieve (cic, cmode)* – This procedure retrieves the appropriately delayed values stored in the queue, delivers the contents in the variables "cic", and "cmode". At initialization, the queue is to contain 0 in all of its positions.

  It should be noted that a Setcod signal containing the values (0, 0, FALSE) should not cause the hardware to generate any type of connection.

A.3.2.5 *The BMI process*

This process will only delay the incoming signal SC-bitmap by an appropriate amount of DCME frames before sending the delayed contents in the signal Addressmap-for-SCs. The process contains the following procedure calls:

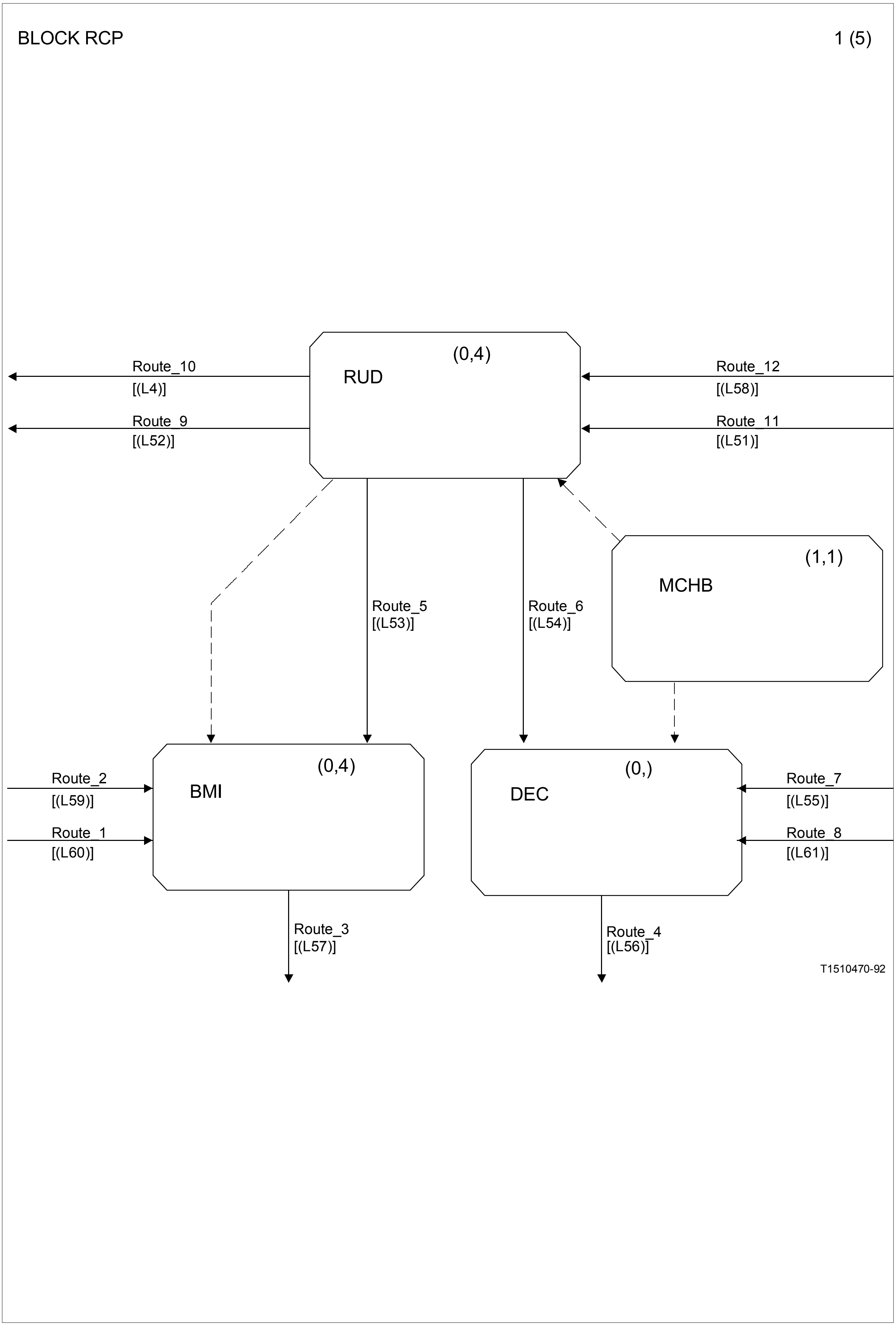
- *Store (bitmap)* – This procedure stores the contents of the SC-bitmap signal at the bottom of the queue.
- *Retrieve (bitmap)* – This procedure removes the appropriately delayed information stored in the queue and loads this information into the Address-map-for-SCs signal. At initialization, the queue contains information such that no connection will be generated when removing the contents and generating a signal to the hardware.



SYSTEM DCME_RECEIVE
1 (2)
RCP
C4
[(L4)]
C52
[(L52)]
C59
[(L59)]
C60
[(L60)]
C58
[(L58)]
C51
[(L51)]
C55
[(L55)]
C61
[(L61)]
C57
[(L57)]
C56
[(L56)]
T1510450-92

```
newtype Call_Type 1
  literals
    'voice',
    'data',
    'transp';
endnewtype call_type 1;
newtype Bit_mode_matrix
  literals
    1,
    0;
endnewtype Bit_mode_matrix;
newtype Call_Type 2
  literals
    'disc',
    'voice',
    'data',
    'transp',
    'bank';
endnewtype call_type 2;

/* Signal definitions */
SIGNAL
  Rxdata(Integer),
  Assign(integer,Integer,Call_Type2),
  Rxtransp(Integer), Rxtransprel(Integer),
  Trigger,
  Setcod(Integer,Integer, Boolean),
  Addressmap_for_SCs(Bit_mode_matrix),
  Process_reset;

/* Signallist definitions */
SIGNALLIST L4 = Rxdata;
SIGNALLIST L51 = Assign;
SIGNALLIST L52 = Rxtransp, Rxtransprel;
SIGNALLIST L55 = Trigger;
SIGNALLIST L56 = Setcod;
SIGNALLIST L57 = Addressmap_for_SCs;
SIGNALLIST L58 = Process_reset;
SIGNALLIST L59 = Process_reset;
SIGNALLIST L60 = Trigger;
SIGNALLIST L61 = Process_reset;
```

T1510460-92



BLOCK RCP
1 (5)
(0,4)
RUD
Route_10
[(L4)]
Route_9
[(L52)]
Route_12
[(L58)]
Route_11
[(L51)]
(1,1)
MCHB
Route_5
[(L53)]
Route_6
[(L54)]
(0,4)
BMI
(0,)
DEC
Route_2
[(L59)]
Route_1
[(L60)]
Route_7
[(L55)]
Route_8
[(L61)]
Route_3
[(L57)]
Route_4
[(L56)]
T1510470-92

BLOCK RCP DATADEFINITION1 (5)

```
connect C60 and Route_1;
connect C59 and Route_2;
connect C57 and Route_3;
connect C56 and Route_4;
connect C55 and Route_7;
connect C61 and Route_8;
connect C52 and Route_9;
connect C4 and Route_10;
connect C51 and Route_11;
connect C58 and Route_12;

SYNONYM number_of_SCs Integer=EXTERNAL;
SYNONYM number_of_decoders Integer=EXTERNAL;
SYNONYM number_of_ICs Integer=EXTERNAL;
synonym preassigned Call_Type2=EXTERNAL;

SYNTYPE decoder_range=Natural
  CONSTANTS 1:number_of_decoders
ENDSYNTYPE decoder_range;

SYNTYPE ic_range=Natural
  CONSTANTS 1:number_of_ICs
ENDSYNTYPE ic_range;

SYNTYPE sc_range=Natural
  CONSTANTS 1:number_of_SCs
ENDSYNTYPE sc_range;

SYNTYPE bit_mode=Natural
  CONSTANTS 0,3,4,5,8
ENDSYNTYPE bit_mode;
```

T1510480-92

BLOCK RCP DATADEFINITION2 (5)

```
NEWTYPE decoder_list
  Array (decoder_range, integer)
ENDNEWTYPE decoder_list;

NEWTYPE ic_access_list
  Array (ic_range, integer)
ENDNEWTYPE ic_access_list;

NEWTYPE sc_access_list
  Array (sc_range, integer)
ENDNEWTYPE sc_access_list;

NEWTYPE assigned_list
  Array (sc_range, bit_mode)
ENDNEWTYPE assigned_mode;

NEWTYPE selected_decoder
  Array (sc_range, integer)
ENDNEWTYPE selected_decoder;

SYNTYPE bitbank_range=Natural
  CONSTANTS 0:11
ENDSYNTYPE bitbank_range;

NEWTYPE bitbank_list
  Array (bitbank_range, integer)
ENDNEWTYPE bitbank_list;

NEWTYPE connection_list
  Array (ic_range, integer)
ENDNEWTYPE connection_list;
```

T1510490-92

BLOCK RCP DATADEFINITION3 (5)

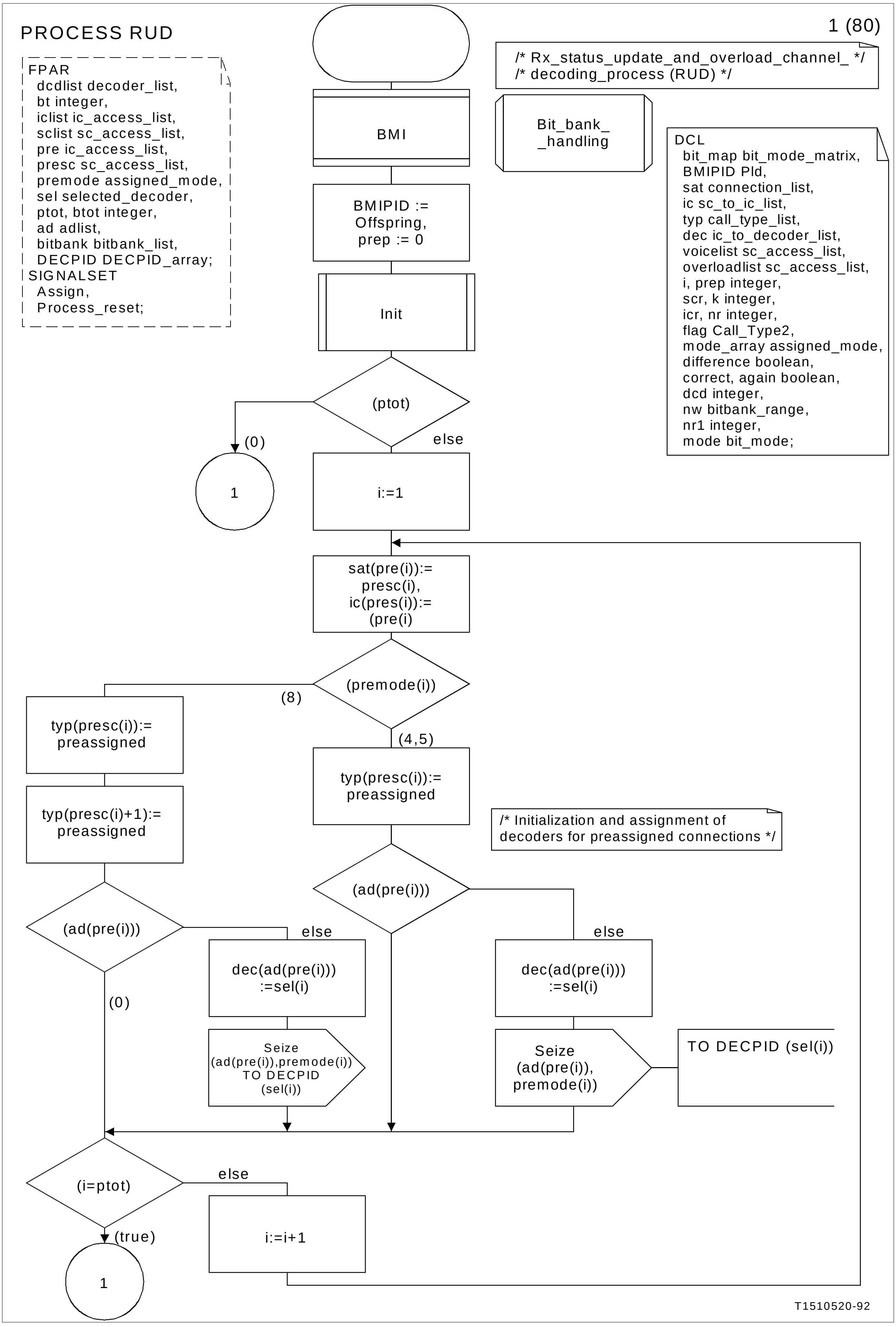
```
NEWTYPE sc_to_ic_list
  Array (sc_range, integer)
ENDNEWTYPE sc_to_ic_list;

NEWTYPE call_type_list
  Array (sc_range, call_type2)
ENDNEWTYPE call_type_list;

NEWTYPE ic_to_decoder_list
  Array (ic_range, integer)
ENDNEWTYPE ic_to_decoder_list;

NEWTYPE DECPID_array
  Array (decoder_range, PId)
ENDNEWTYPE DECPID_array;

NEWTYPE adlist
  Array (ic_range, integer)
ENDNEWTYPE adlist;

NEWTYPE queue /* = EXTERNAL */
ENDNEWTYPE queue;
```

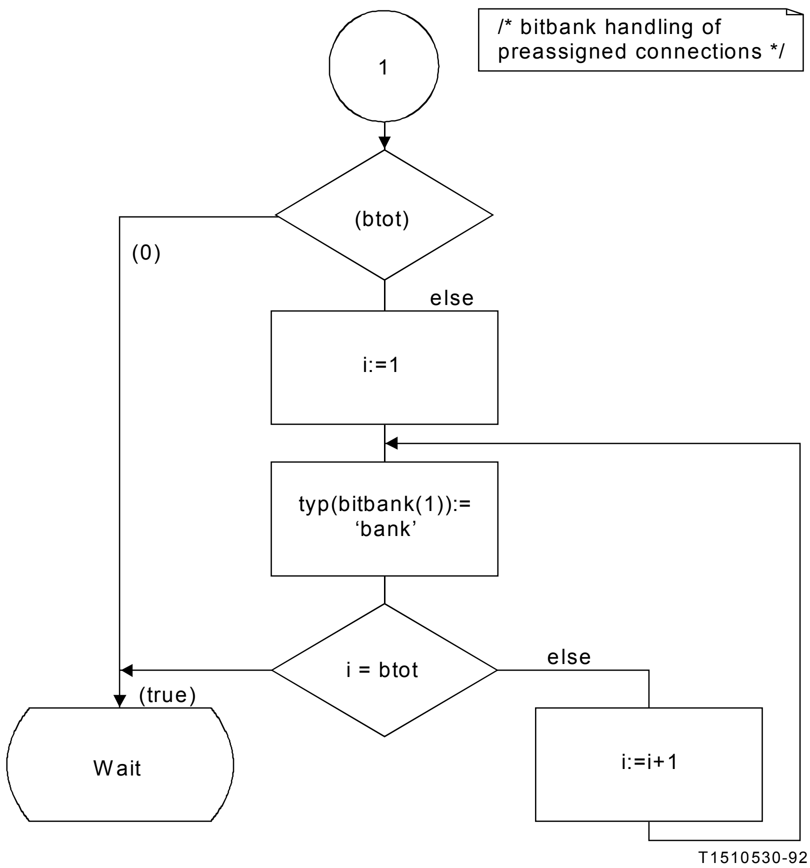
T1510500-92

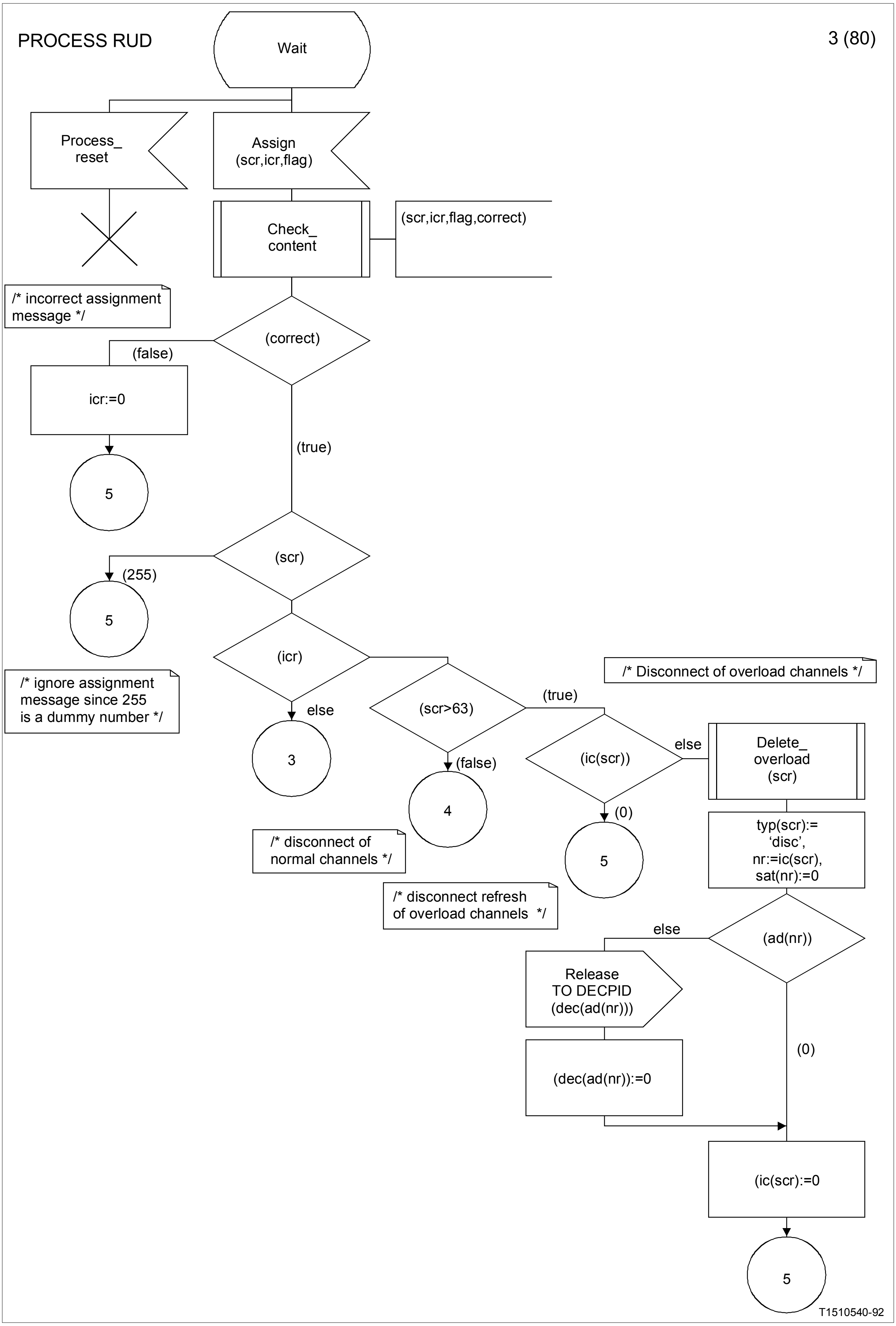
BLOCK RCP SIGNALDEFINITION (5)

```
/* Signal definitions */
SIGNAL
  SC_ bitmap (Bit_mode_matrix),
  Seize(Integer,Integer),
  Release, Seizev(Integer),
  Mode_map(Integer);

/* Signallist definitions */
SIGNALLIST L53 = SC_bitmap;
SIGNALLIST L54 = Seize, Release, Seizev, Mode_map;
```

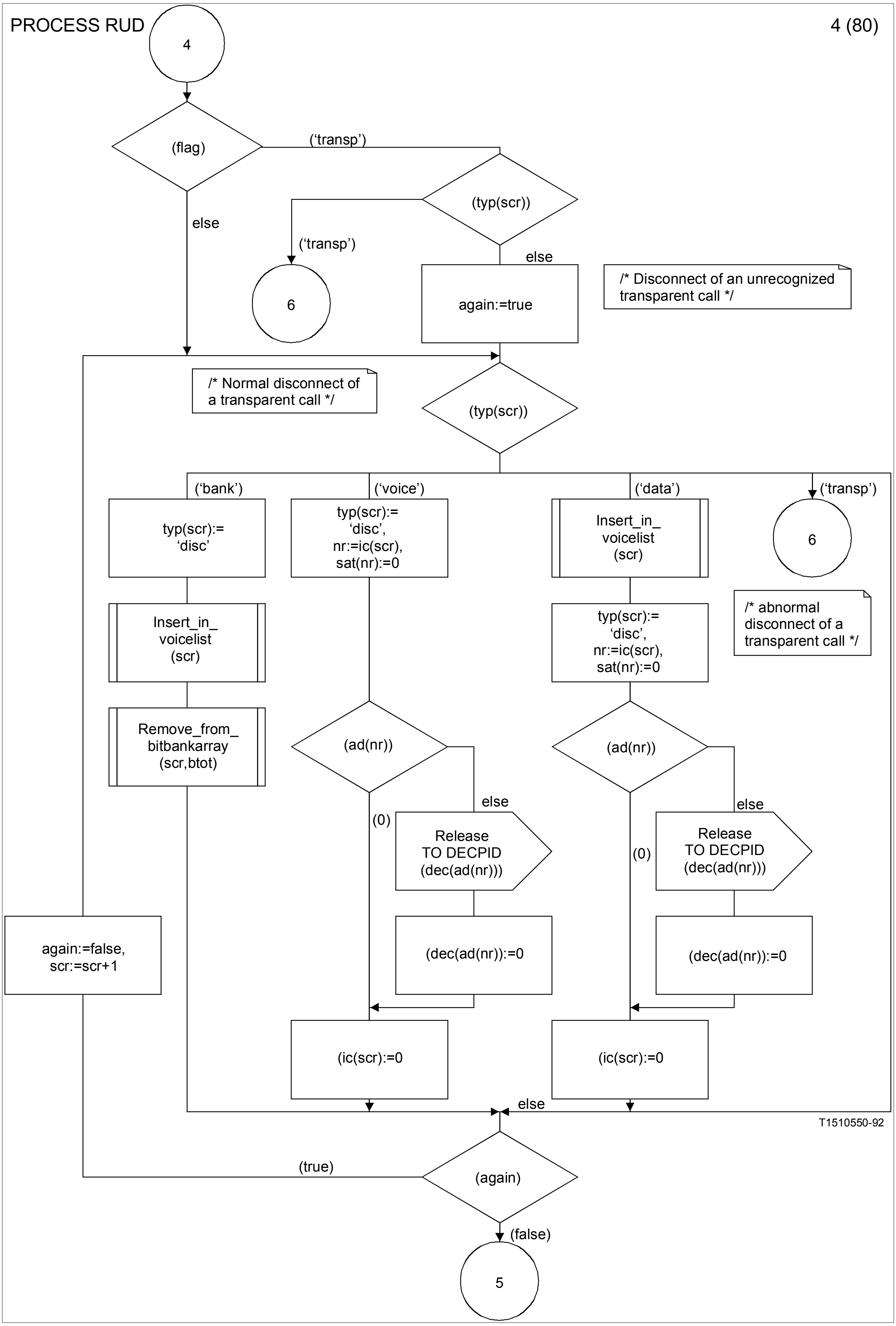
T1510510-92


PROCESS RUD
1 (80)
FPAR
dcdlist decoder_list,
bt integer,
iclist ic_access_list,
sclist sc_access_list,
pre ic_access_list,
presc sc_access_list,
premode assigned_mode,
sel selected_decoder,
ptot, btot integer,
ad adlist,
bitbank bitbank_list,
DECPID DECPID_array;
SIGNALSET
Assign,
Process_reset;
/* Rx_status_update_and_overload_channel_ */
/* decoding_process (RUD) */
BMI
Bit_bank_
_handling
DCL
bit_map bit_mode_matrix,
BMIPID PId,
sat connection_list,
ic sc_to_ic_list,
typ call_type_list,
dec ic_to_decoder_list,
voicelist sc_access_list,
overloadlist sc_access_list,
i, prep integer,
scr, k integer,
icr, nr integer,
flag Call_Type2,
mode_array assigned_mode,
difference boolean,
correct, again boolean,
dcd integer,
nw bitbank_range,
nr1 integer,
mode bit_mode;
BMIPID :=
Offspring,
prep := 0
Init
(ptot)
(0)
else
1
i:=1
sat(pre(i)):=
presc(i),
ic(pres(i)):=
(pre(i)
(premode(i))
(8)
(4,5)
typ(presc(i)):=
preassigned
typ(presc(i)+1):=
preassigned
typ(presc(i)):=
preassigned
/* Initialization and assignment of
decoders for preassigned connections */
(ad(pre(i)))
(ad(pre(i)))
else
else
dec(ad(pre(i)))
:=sel(i)
dec(ad(pre(i)))
:=sel(i)
(0)
Seize
(ad(pre(i)),premode(i))
TO DECPID
(sel(i))
Seize
(ad(pre(i)),
premode(i))
TO DECPID (sel(i))
(i=ptot)
else
(true)
i:=i+1
1
T1510520-92

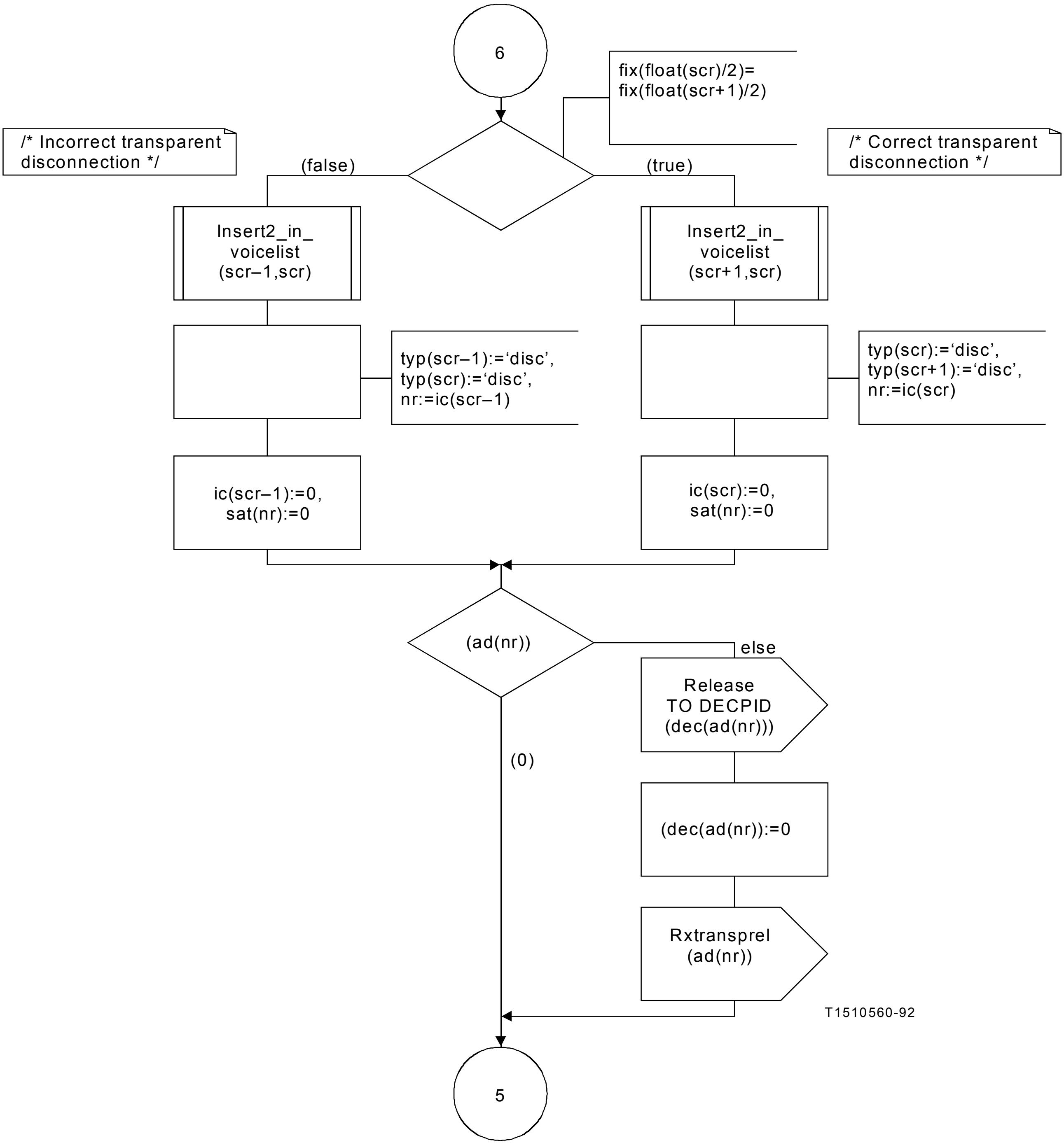

/* bitbank handling of
preassigned connections */
1
(btot)
(0)
else
i:=1
typ(bitbank(1)):=
'bank'
i = btot
else
(true)
Wait
i:=i+1
T1510530-92

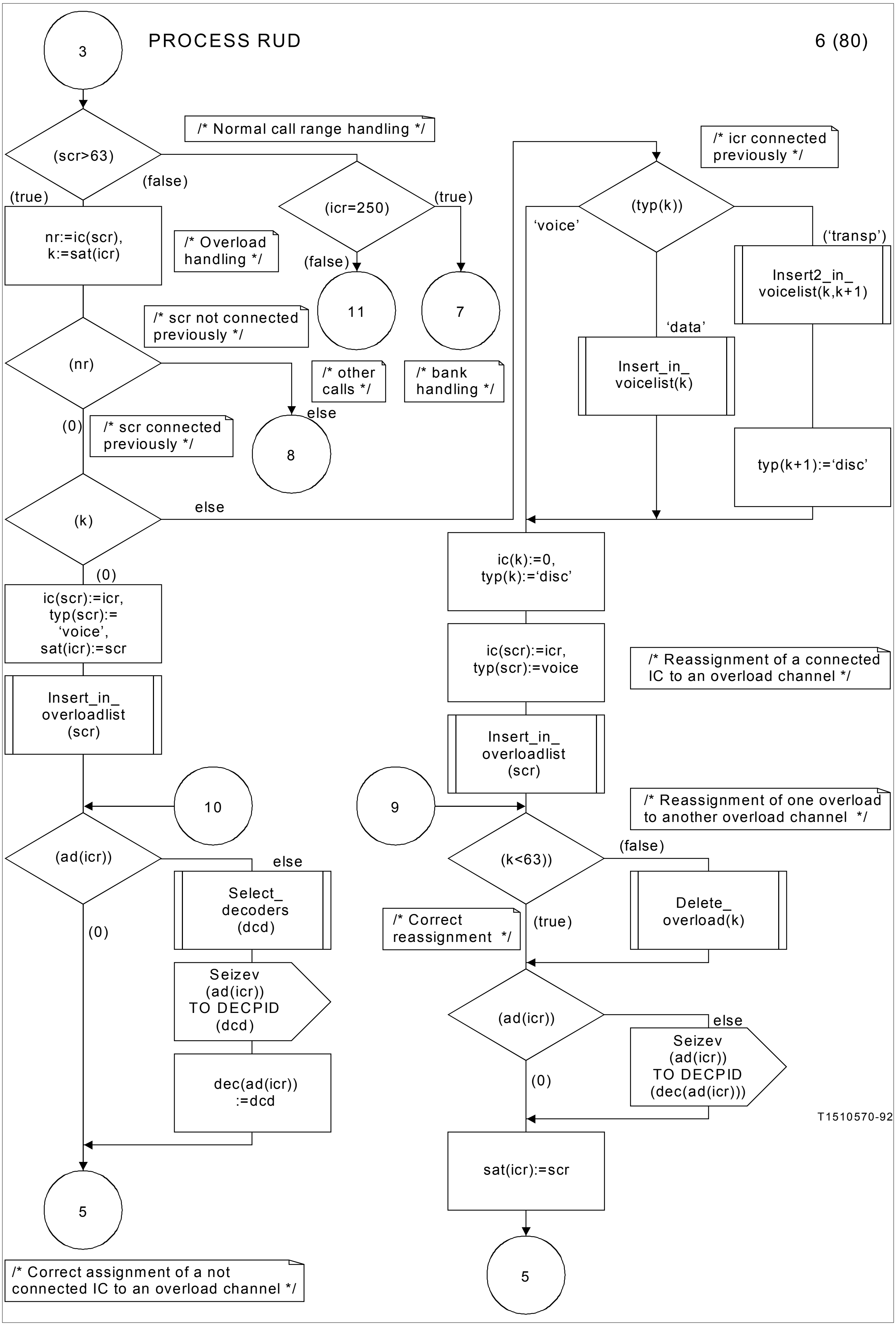


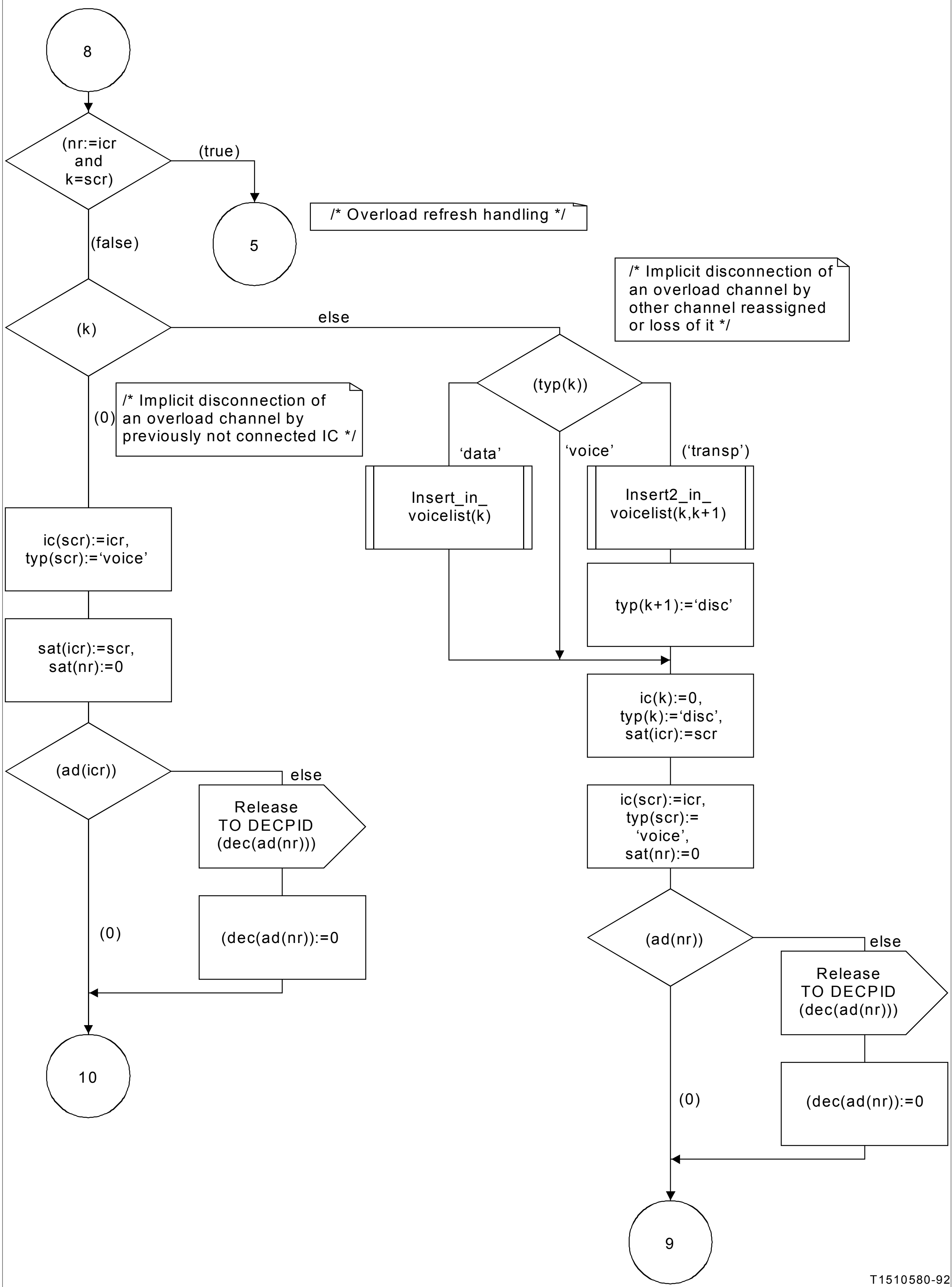
PROCESS RUD
3 (80)
Wait
Process_
reset
Assign
(scr,icr,flag)
Check_
content
(scr,icr,flag,correct)
/* incorrect assignment
message */
(correct)
(false)
icr:=0
5
(true)
(scr)
(255)
5
/* ignore assignment
message since 255
is a dummy number */
(icr)
else
3
(scr>63)
(false)
4
/* disconnect of
normal channels */
/* disconnect refresh
of overload channels */
(true)
/* Disconnect of overload channels */
(ic(scr))
else
(0)
5
Delete_
overload
(scr)
typ(scr):=
'disc',
nr:=ic(scr),
sat(nr):=0
(ad(nr))
else
Release
TO DECPID
(dec(ad(nr)))
(dec(ad(nr)):=0
(0)
(ic(scr):=0
5
T1510540-92

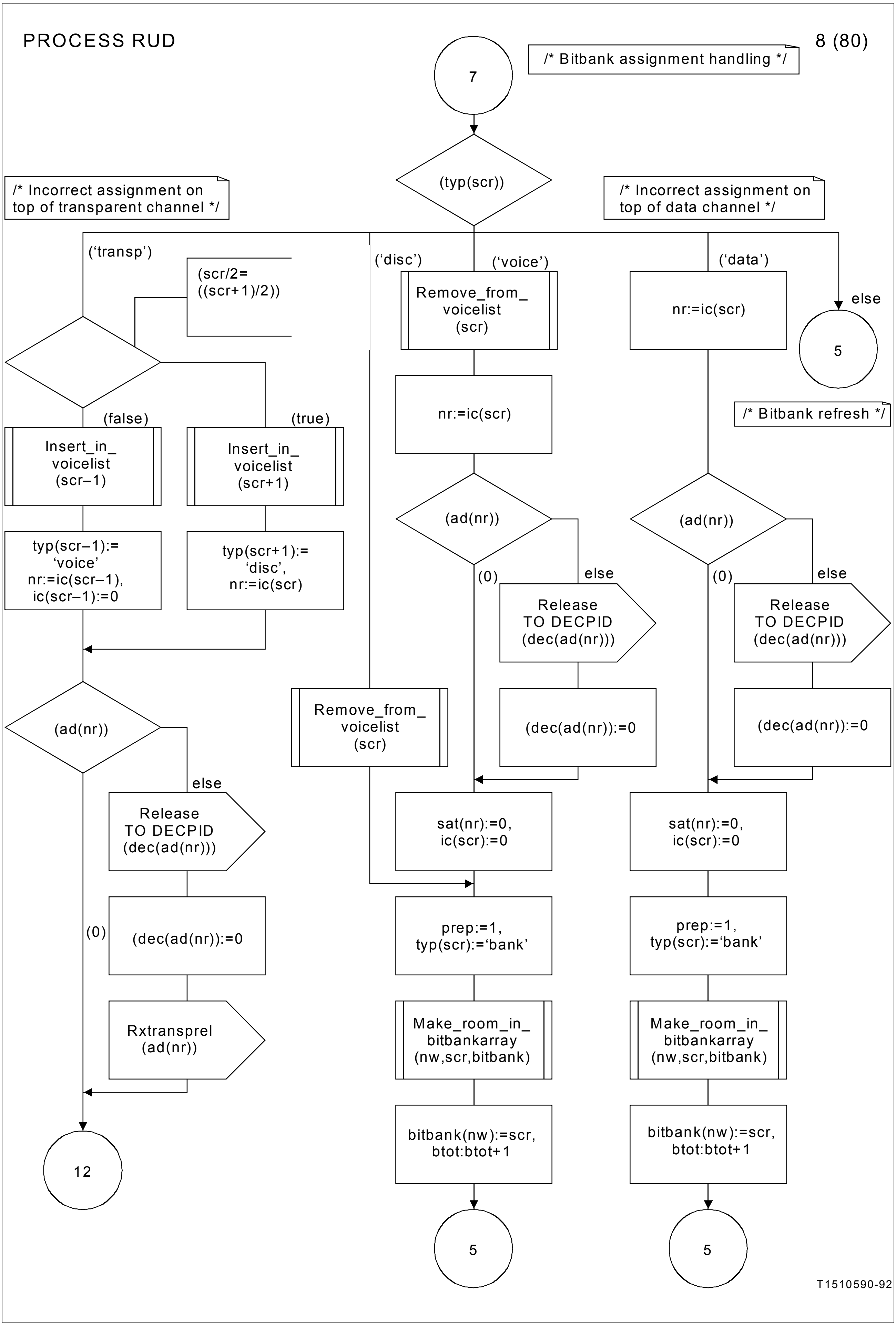


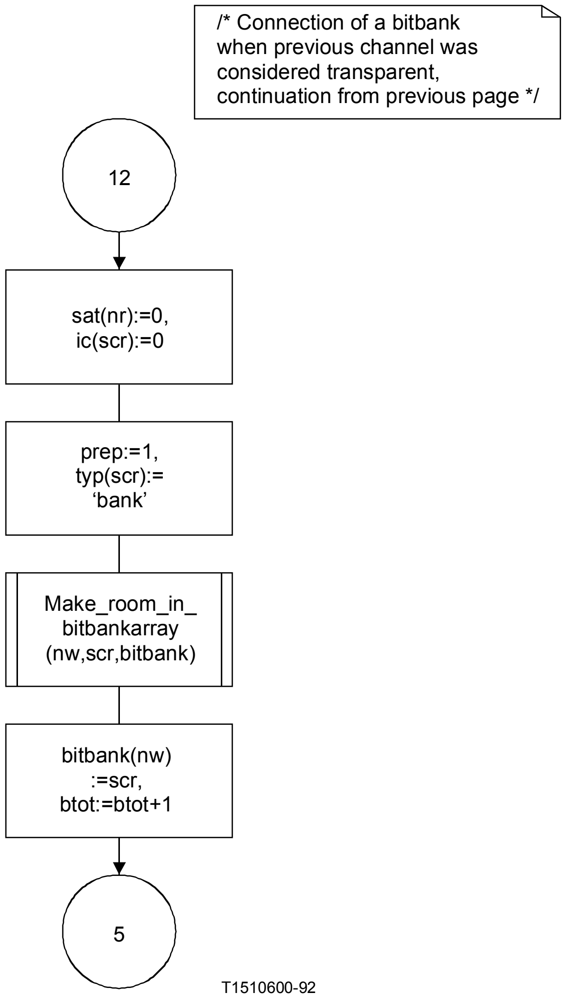
PROCESS RUD
4 (80)
4
(flag)
('transp')
(typ(scr))
else
('transp')
6
else
again:=true
/* Disconnect of an unrecognized transparent call */
/* Normal disconnect of a transparent call */
(typ(scr))
('bank')
('voice')
('data')
('transp')
typ(scr):= 'disc'
typ(scr):= 'disc', nr:=ic(scr), sat(nr):=0
Insert_in_ voicelist (scr)
6
/* abnormal disconnect of a transparent call */
Insert_in_ voicelist (scr)
typ(scr):= 'disc', nr:=ic(scr), sat(nr):=0
Remove_from_ bitbankarray (scr,btot)
(ad(nr))
(ad(nr))
else
else
(0)
Release TO DECPID (dec(ad(nr)))
(0)
Release TO DECPID (dec(ad(nr)))
again:=false, scr:=scr+1
(dec(ad(nr)):=0
(dec(ad(nr)):=0
(ic(scr):=0
(ic(scr):=0
else
T1510550-92
(true)
(again)
(false)
5


6
fix(float(scr)/2)=
fix(float(scr+1)/2)
/* Incorrect transparent
disconnection */
/* Correct transparent
disconnection */
(false)
(true)
Insert2_in_
voicelist
(scr–1,scr)
Insert2_in_
voicelist
(scr+1,scr)
typ(scr–1):='disc',
typ(scr):='disc',
nr:=ic(scr–1)
typ(scr):='disc',
typ(scr+1):='disc',
nr:=ic(scr)
ic(scr–1):=0,
sat(nr):=0
ic(scr):=0,
sat(nr):=0
(ad(nr))
else
Release
TO DECPID
(dec(ad(nr)))
(0)
(dec(ad(nr)):=0
Rxtransprel
(ad(nr))
T1510560-92
5

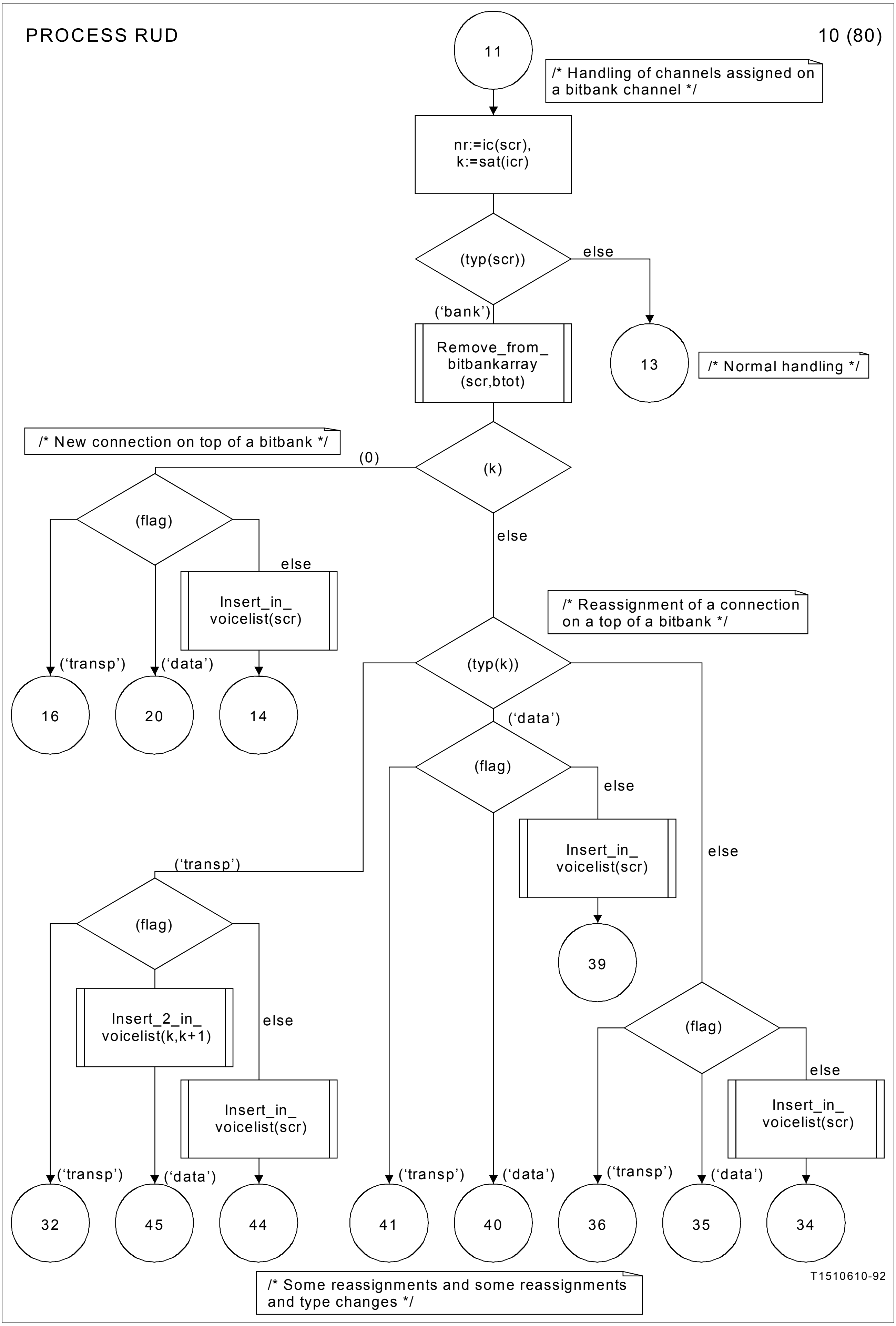

3
PROCESS RUD
6 (80)
/* Normal call range handling */
(scr>63)
(true)
(false)
nr:=ic(scr),
k:=sat(icr)
/* Overload handling */
(icr=250)
(true)
(false)
11
7
/* other calls */
/* bank handling */
/* scr not connected previously */
(nr)
else
8
(0)
/* scr connected previously */
(k)
else
(0)
ic(scr):=icr,
typ(scr):=
'voice',
sat(icr):=scr
Insert_in_
overloadlist
(scr)
10
(ad(icr))
else
(0)
Select_
decoders
(dcd)
Seizev
(ad(icr))
TO DECPID
(dcd)
dec(ad(icr))
:=dcd
5
/* Correct assignment of a not connected IC to an overload channel */
/* icr connected previously */
(typ(k))
'voice'
('transp')
'data'
Insert2_in_
voicelist(k,k+1)
Insert_in_
voicelist(k)
typ(k+1):='disc'
ic(k):=0,
typ(k):='disc'
ic(scr):=icr,
typ(scr):=voice
/* Reassignment of a connected IC to an overload channel */
Insert_in_
overloadlist
(scr)
9
/* Reassignment of one overload to another overload channel */
(k<63))
(false)
(true)
/* Correct reassignment */
Delete_
overload(k)
(ad(icr))
else
(0)
Seizev
(ad(icr))
TO DECPID
(dec(ad(icr)))
sat(icr):=scr
5
T1510570-92


8
(nr:=icr
and
k=scr)
(true)
5
/* Overload refresh handling */
(false)
/* Implicit disconnection of
an overload channel by
other channel reassigned
or loss of it */
(k)
else
(typ(k))
(0)
/* Implicit disconnection of
an overload channel by
previously not connected IC */
'data'
'voice'
('transp')
Insert_in_
voicelist(k)
Insert2_in_
voicelist(k,k+1)
ic(scr):=icr,
typ(scr):='voice'
typ(k+1):='disc'
sat(icr):=scr,
sat(nr):=0
ic(k):=0,
typ(k):='disc',
sat(icr):=scr
(ad(icr))
else
Release
TO DECPID
(dec(ad(nr)))
ic(scr):=icr,
typ(scr):=
'voice',
sat(nr):=0
(0)
(dec(ad(nr)):=0
(ad(nr))
else
Release
TO DECPID
(dec(ad(nr)))
10
(0)
(dec(ad(nr)):=0
9
T1510580-92

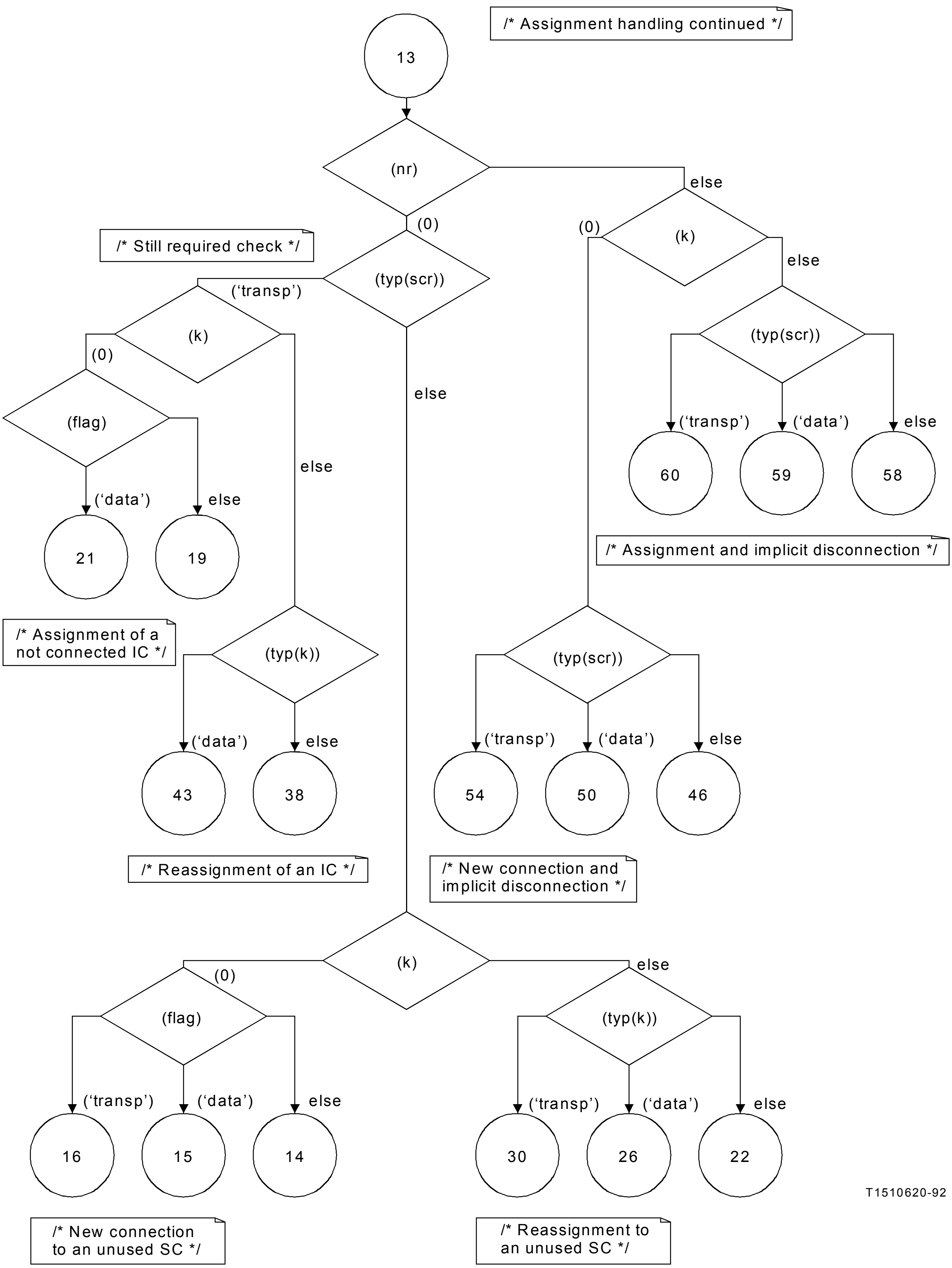

PROCESS RUD
8 (80)
7
/* Bitbank assignment handling */
(typ(scr))
/* Incorrect assignment on top of transparent channel */
/* Incorrect assignment on top of data channel */
('transp')
('disc')
('voice')
('data')
else
(scr/2= ((scr+1)/2))
Remove_from_ voicelist (scr)
nr:=ic(scr)
5
/* Bitbank refresh */
(false)
(true)
Insert_in_ voicelist (scr–1)
Insert_in_ voicelist (scr+1)
nr:=ic(scr)
(ad(nr))
(ad(nr))
(0)
else
(0)
else
typ(scr–1):= 'voice' nr:=ic(scr–1), ic(scr–1):=0
typ(scr+1):= 'disc', nr:=ic(scr)
Release TO DECPID (dec(ad(nr)))
Release TO DECPID (dec(ad(nr)))
(ad(nr))
Remove_from_ voicelist (scr)
(dec(ad(nr)):=0
(dec(ad(nr)):=0
else
Release TO DECPID (dec(ad(nr)))
sat(nr):=0, ic(scr):=0
sat(nr):=0, ic(scr):=0
(0)
(dec(ad(nr)):=0
prep:=1, typ(scr):='bank'
prep:=1, typ(scr):='bank'
Rxtransprel (ad(nr))
Make_room_in_ bitbankarray (nw,scr,bitbank)
Make_room_in_ bitbankarray (nw,scr,bitbank)
bitbank(nw):=scr, btot:btot+1
bitbank(nw):=scr, btot:btot+1
12
5
5
T1510590-92

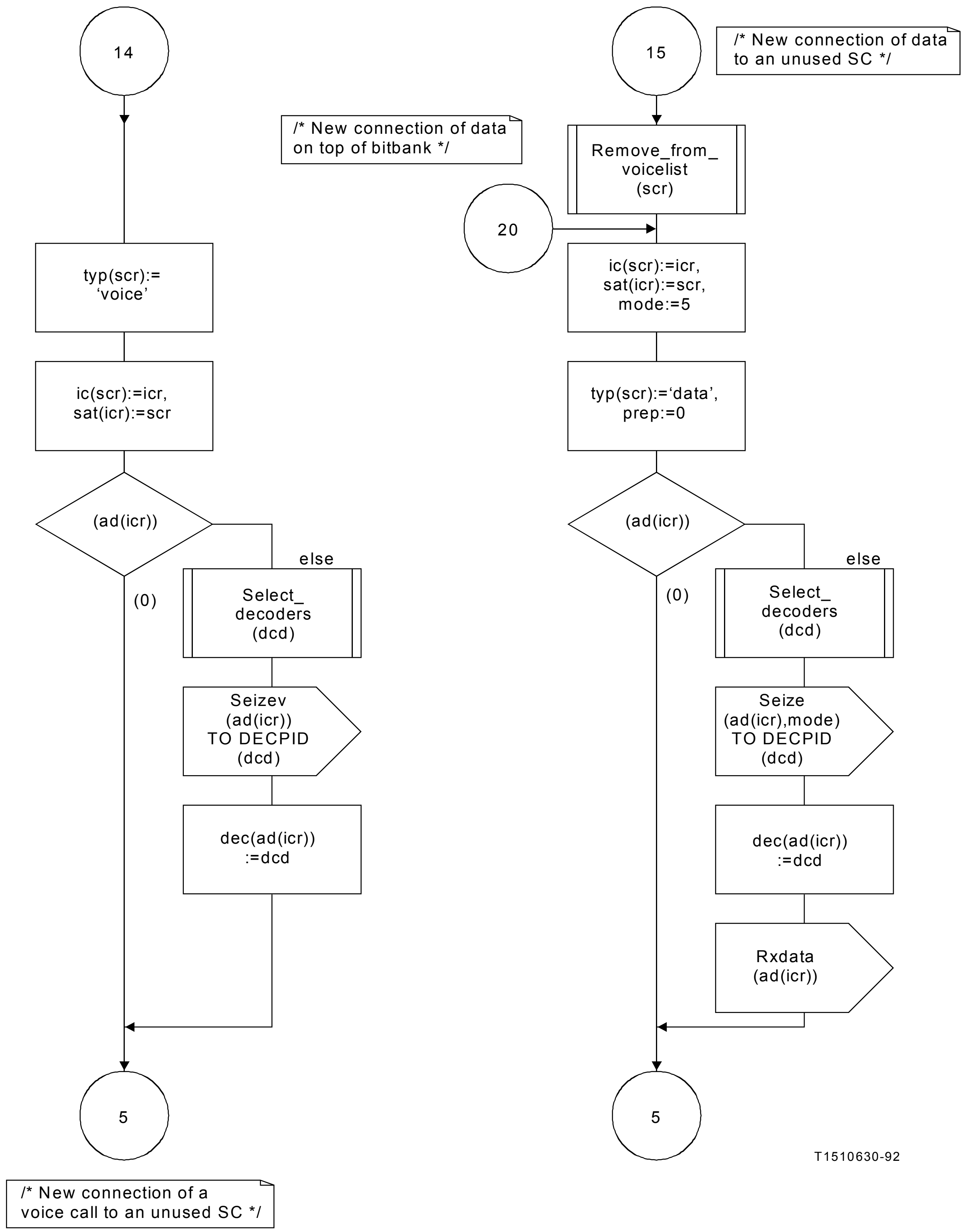

/* Connection of a bitbank
when previous channel was
considered transparent,
continuation from previous page */
12
sat(nr):=0,
ic(scr):=0
prep:=1,
typ(scr):=
'bank'
Make_room_in_
bitbankarray
(nw,scr,bitbank)
bitbank(nw)
:=scr,
btot:=btot+1
5

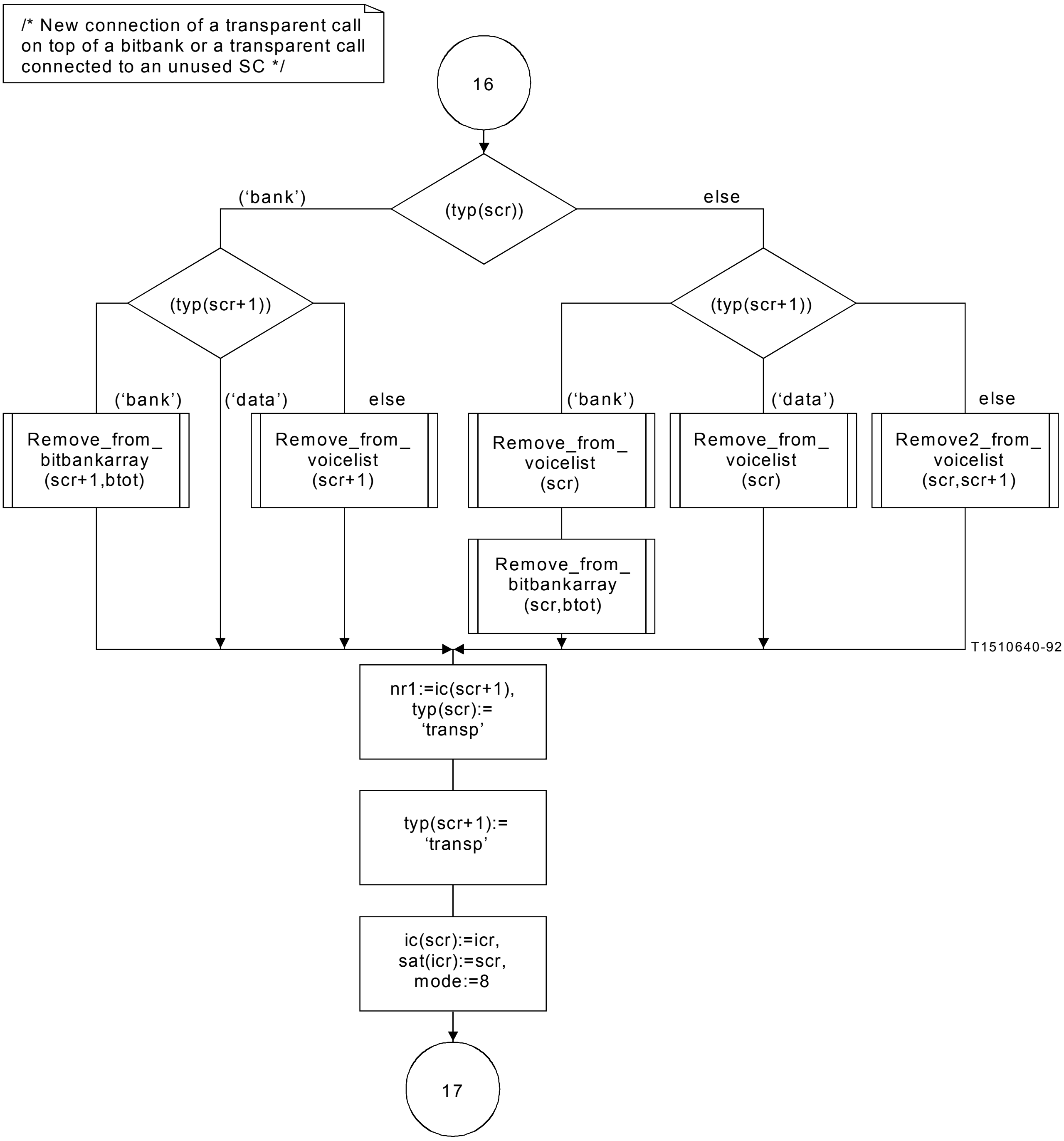
T1510600-92

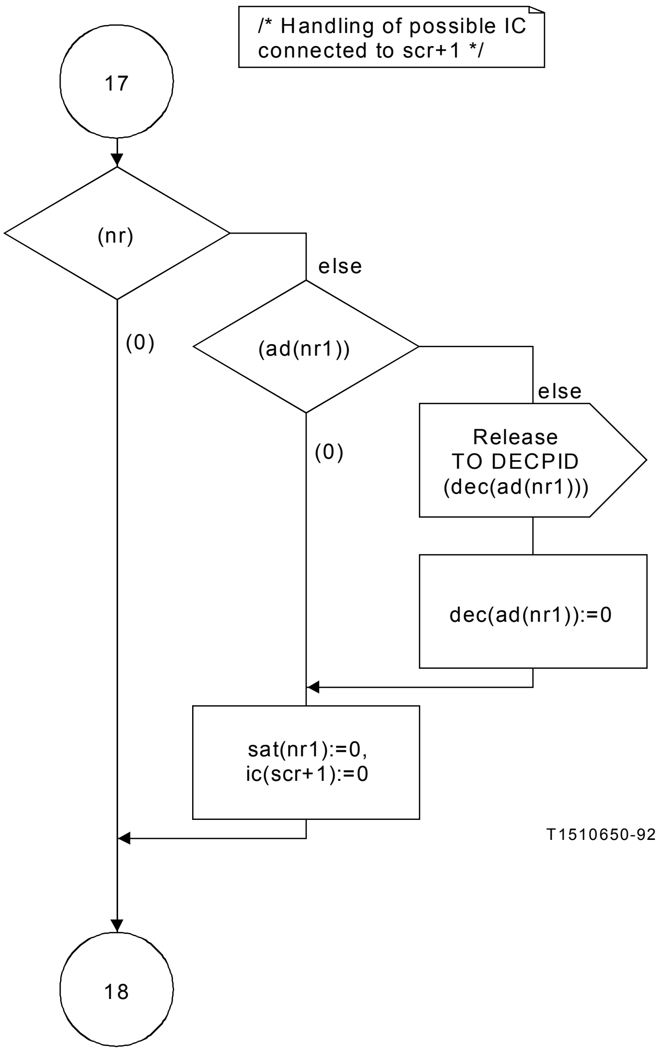


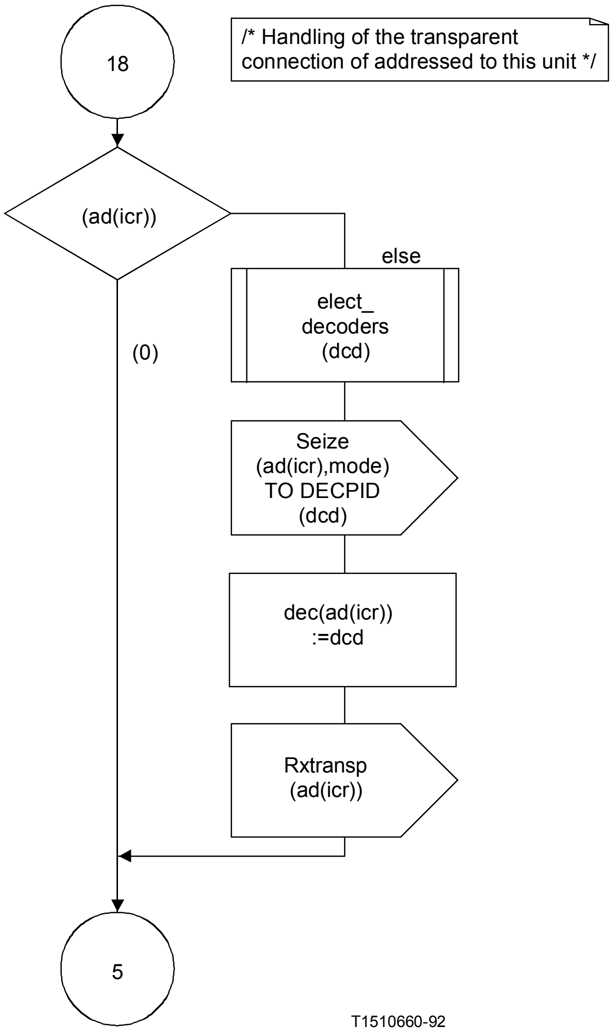
PROCESS RUD
10 (80)
11
/* Handling of channels assigned on a bitbank channel */
nr:=ic(scr),
k:=sat(icr)
(typ(scr))
else
('bank')
Remove_from_
bitbankarray
(scr,btot)
13
/* Normal handling */
/* New connection on top of a bitbank */
(0)
(k)
(flag)
else
Insert_in_
voicelist(scr)
('transp')
('data')
16
20
14
/* Reassignment of a connection on a top of a bitbank */
(typ(k))
('data')
(flag)
else
Insert_in_
voicelist(scr)
39
('transp')
(flag)
Insert_2_in_
voicelist(k,k+1)
else
Insert_in_
voicelist(scr)
(flag)
else
Insert_in_
voicelist(scr)
('transp')
('data')
('transp')
('data')
('transp')
('data')
32
45
44
41
40
36
35
34
T1510610-92
/* Some reassignments and some reassignments and type changes */

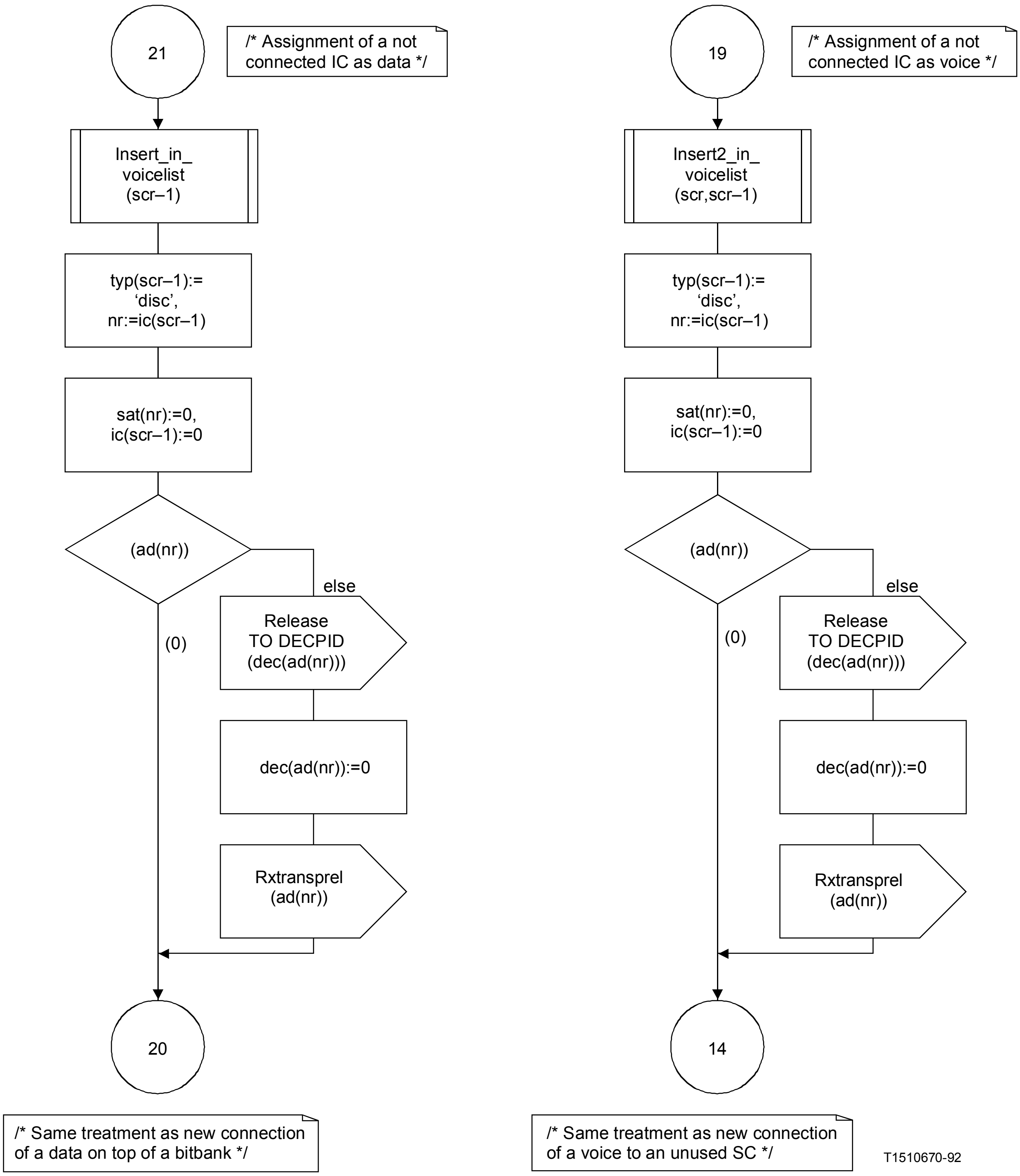

/* Assignment handling continued */
13
(nr)
(0)
else
/* Still required check */
(typ(scr))
('transp')
(k)
(0)
(flag)
('data')
else
21
19
else
/* Assignment of a not connected IC */
(typ(k))
('data')
else
43
38
/* Reassignment of an IC */
else
(0)
(k)
else
(typ(scr))
('transp')
('data')
else
60
59
58
/* Assignment and implicit disconnection */
(typ(scr))
('transp')
('data')
else
54
50
46
/* New connection and implicit disconnection */
(k)
(0)
else
(flag)
('transp')
('data')
else
16
15
14
/* New connection to an unused SC */
(typ(k))
('transp')
('data')
else
30
26
22
/* Reassignment to an unused SC */
T1510620-92

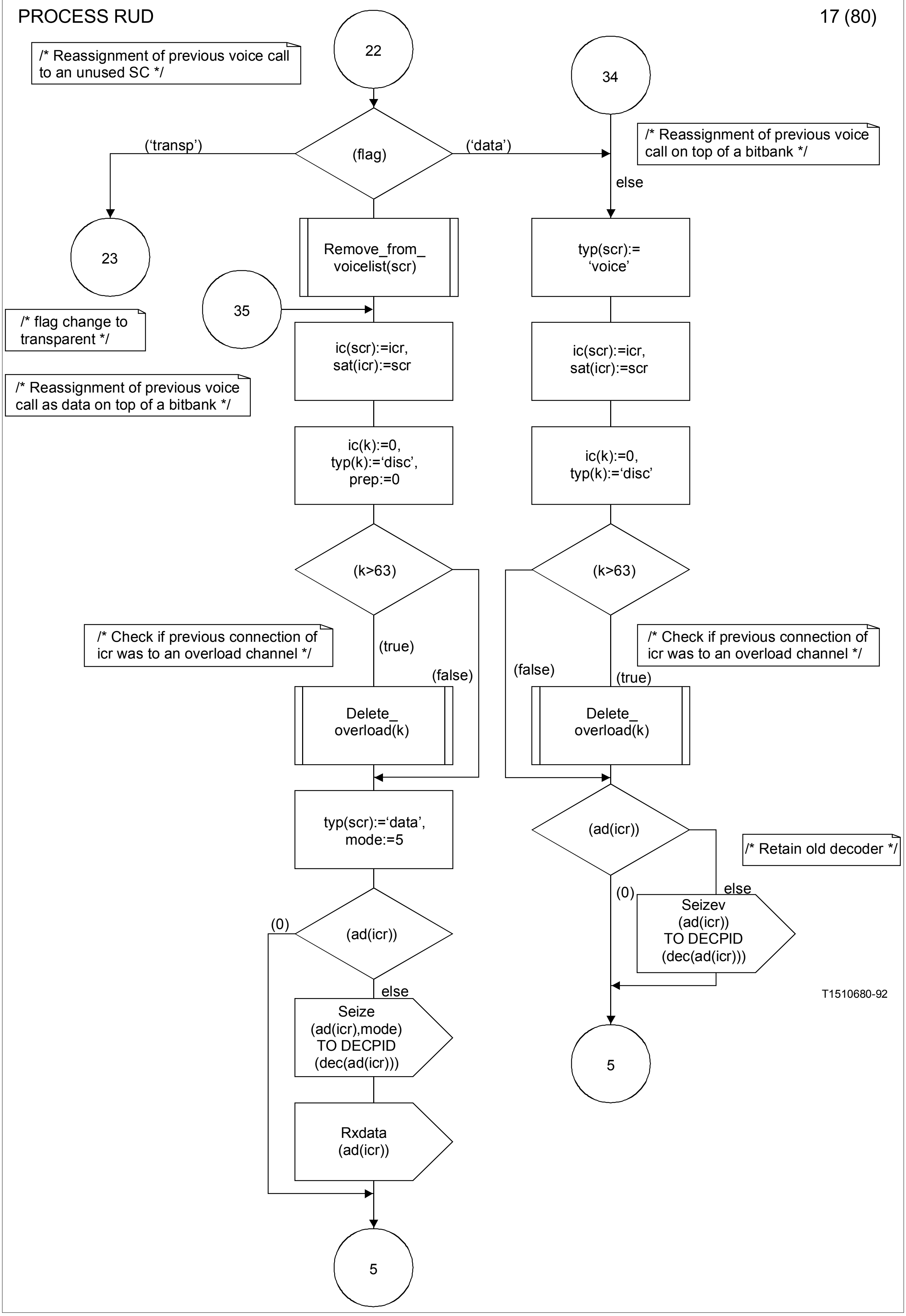

14
typ(scr):=
'voice'
ic(scr):=icr,
sat(icr):=scr
(ad(icr))
(0)
else
Select_
decoders
(dcd)
Seizev
(ad(icr))
TO DECPID
(dcd)
dec(ad(icr))
:=dcd
5
/* New connection of a
voice call to an unused SC */
/* New connection of data
on top of bitbank */
15
/* New connection of data
to an unused SC */
Remove_from_
voicelist
(scr)
20
ic(scr):=icr,
sat(icr):=scr,
mode:=5
typ(scr):='data',
prep:=0
(ad(icr))
(0)
else
Select_
decoders
(dcd)
Seize
(ad(icr),mode)
TO DECPID
(dcd)
dec(ad(icr))
:=dcd
Rxdata
(ad(icr))
5
T1510630-92


/* New connection of a transparent call on top of a bitbank or a transparent call connected to an unused SC */
16
(typ(scr))
('bank')
else
(typ(scr+1))
(typ(scr+1))
('bank')
('data')
else
('bank')
('data')
else
Remove_from_ bitbankarray (scr+1,btot)
Remove_from_ voicelist (scr+1)
Remove_from_ voicelist (scr)
Remove_from_ voicelist (scr)
Remove2_from_ voicelist (scr,scr+1)
Remove_from_ bitbankarray (scr,btot)
T1510640-92
nr1:=ic(scr+1), typ(scr):= 'transp'
typ(scr+1):= 'transp'
ic(scr):=icr, sat(icr):=scr, mode:=8
17

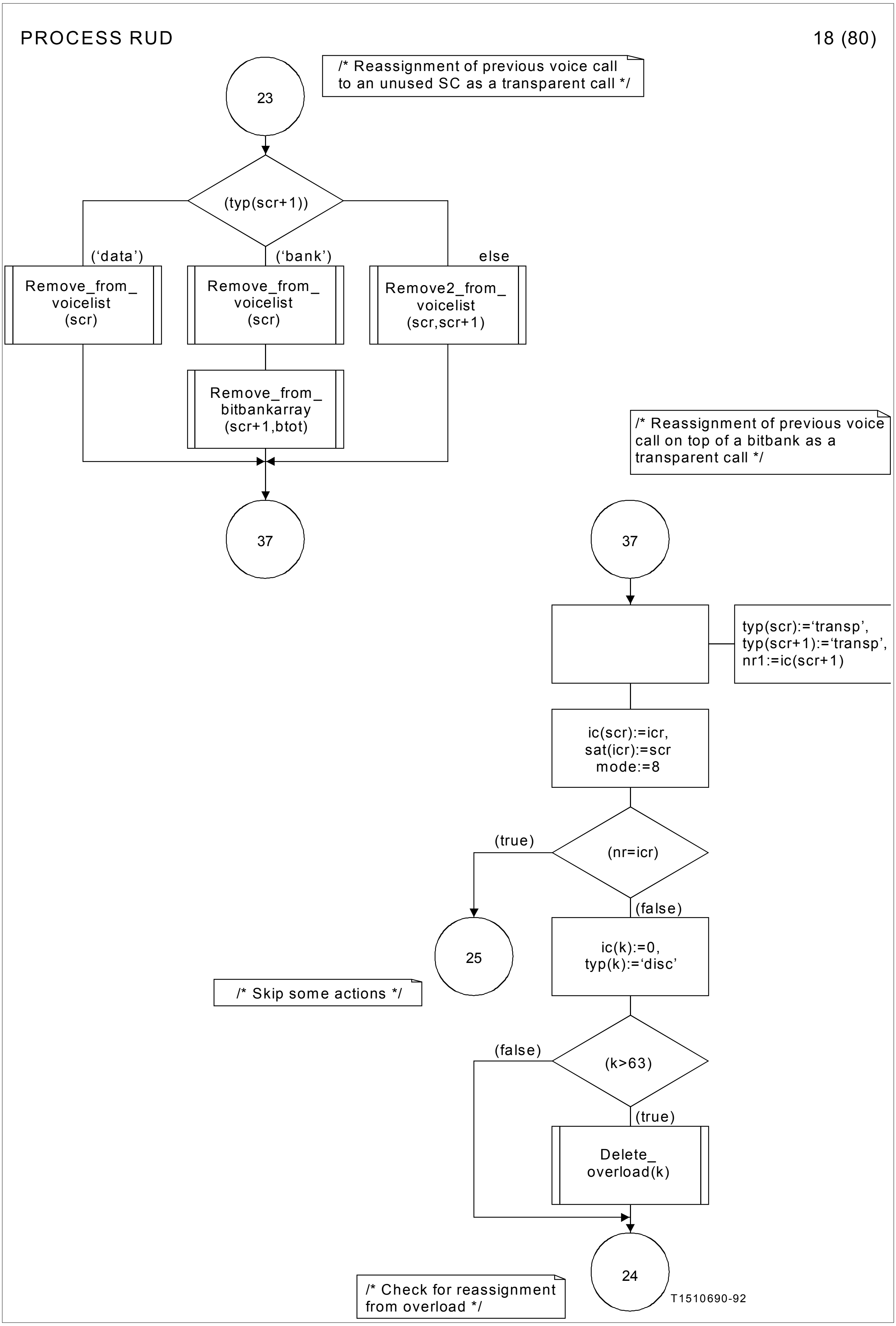

/* Handling of possible IC connected to scr+1 */
17
(nr)
else
(0)
(ad(nr1))
else
(0)
Release
TO DECPID
(dec(ad(nr1)))
dec(ad(nr1)):=0
sat(nr1):=0,
ic(scr+1):=0
T1510650-92
18

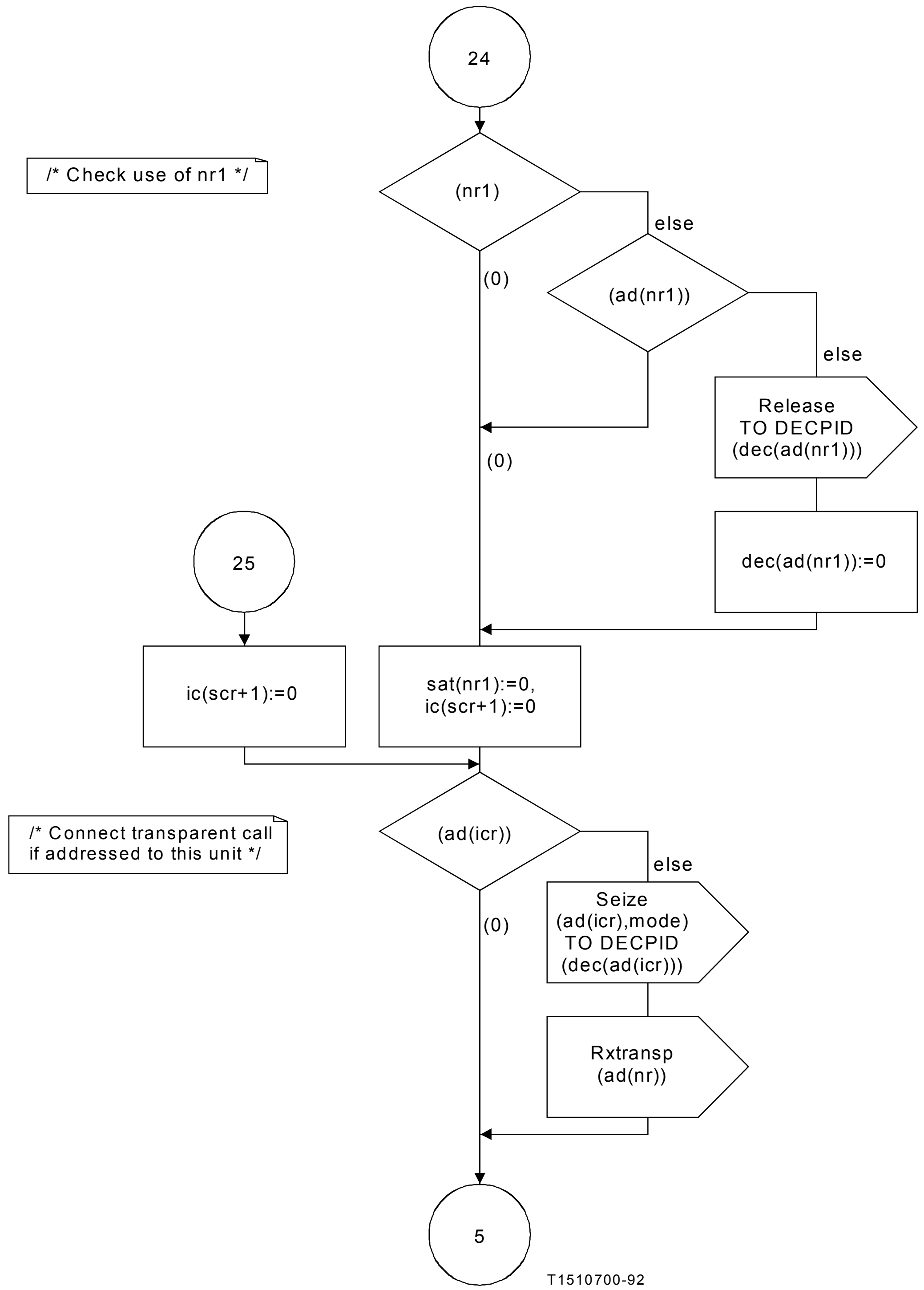

18
/* Handling of the transparent connection of addressed to this unit */
(ad(icr))
else
(0)
elect_ decoders (dcd)
Seize (ad(icr),mode) TO DECPID (dcd)
dec(ad(icr)) :=dcd
Rxtransp (ad(icr))
5
T1510660-92

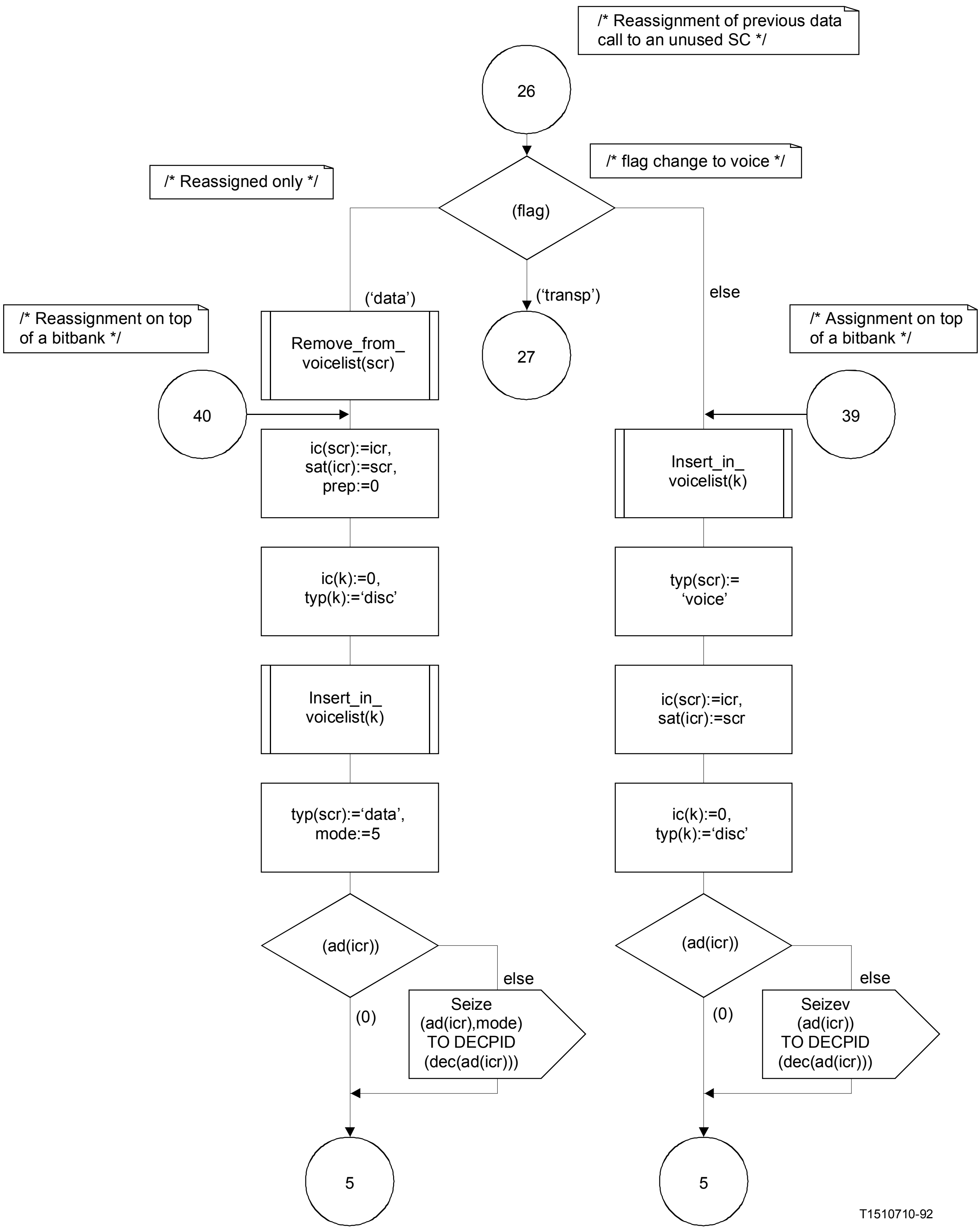

21
/* Assignment of a not connected IC as data */
Insert_in_ voicelist (scr–1)
typ(scr–1):= 'disc', nr:=ic(scr–1)
sat(nr):=0, ic(scr–1):=0
(ad(nr))
else
(0)
Release TO DECPID (dec(ad(nr)))
dec(ad(nr)):=0
Rxtransprel (ad(nr))
20
/* Same treatment as new connection of a data on top of a bitbank */
19
/* Assignment of a not connected IC as voice */
Insert2_in_ voicelist (scr,scr–1)
typ(scr–1):= 'disc', nr:=ic(scr–1)
sat(nr):=0, ic(scr–1):=0
(ad(nr))
else
(0)
Release TO DECPID (dec(ad(nr)))
dec(ad(nr)):=0
Rxtransprel (ad(nr))
14
/* Same treatment as new connection of a voice to an unused SC */
T1510670-92

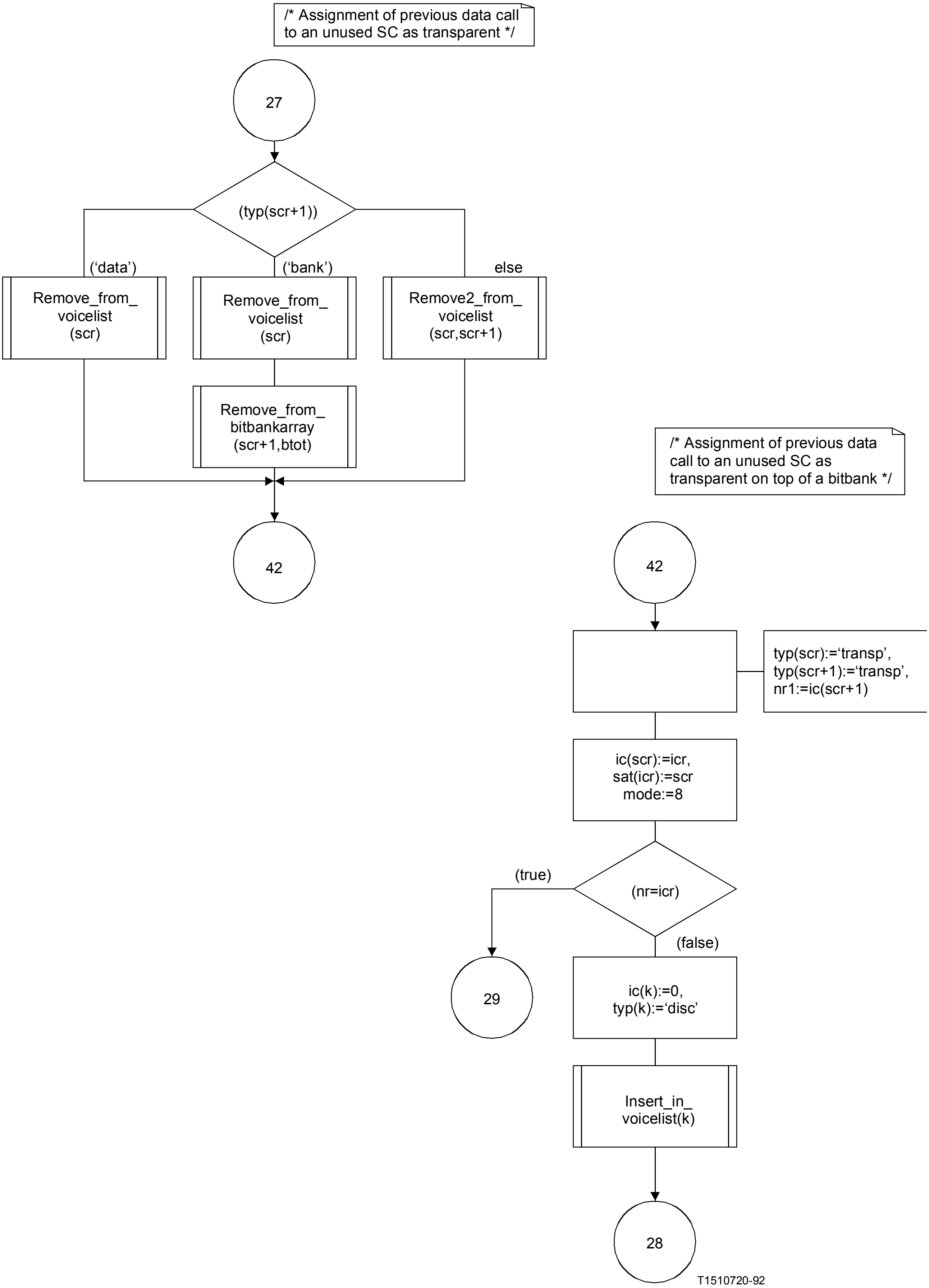


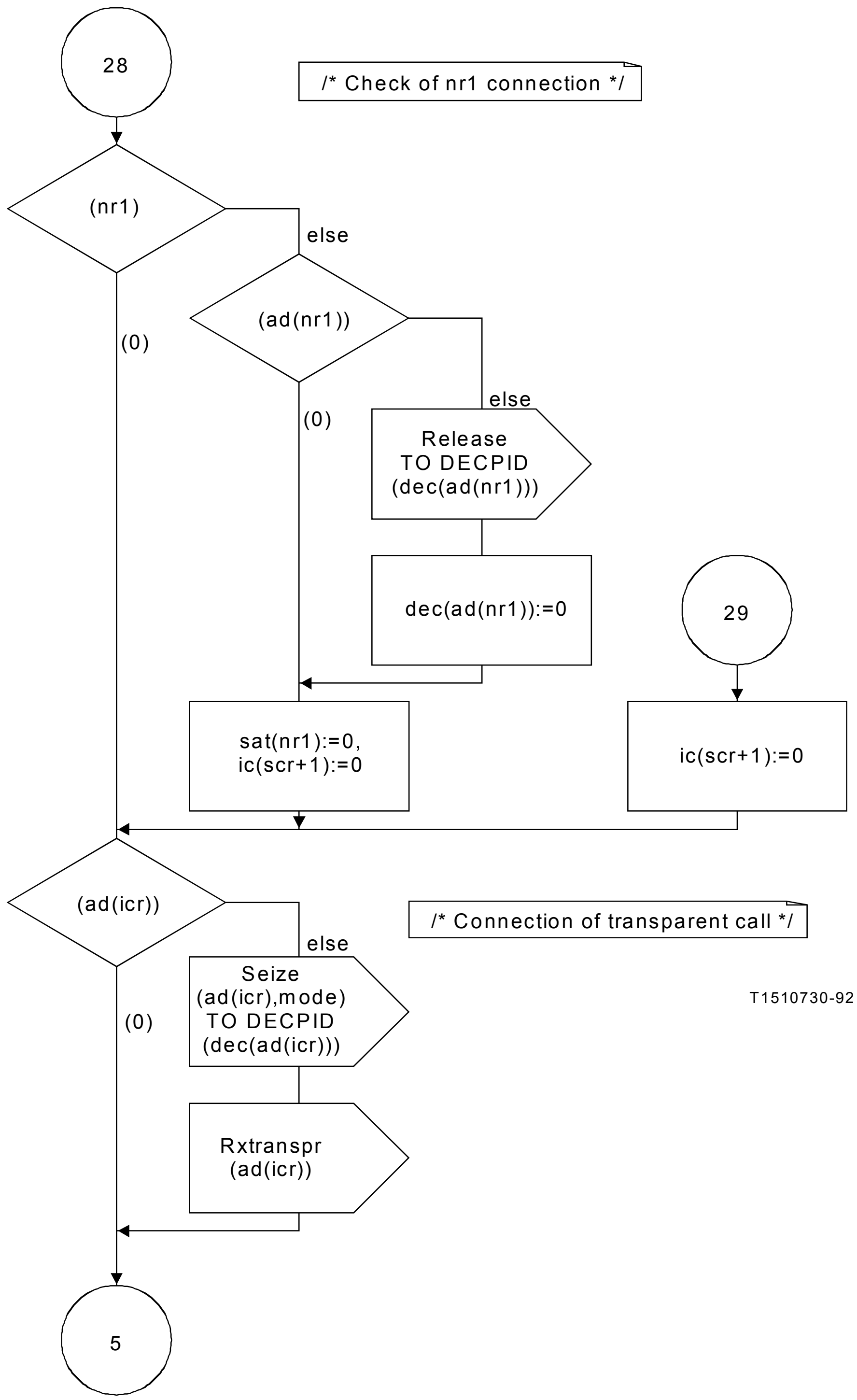
/* Reassignment of previous voice call to an unused SC */
22
34
(flag)
('transp')
('data')
/* Reassignment of previous voice call on top of a bitbank */
else
23
/* flag change to transparent */
/* Reassignment of previous voice call as data on top of a bitbank */
Remove_from_ voicelist(scr)
35
ic(scr):=icr, sat(icr):=scr
ic(k):=0, typ(k):='disc', prep:=0
(k>63)
(true)
(false)
/* Check if previous connection of icr was to an overload channel */
Delete_ overload(k)
typ(scr):='data', mode:=5
(ad(icr))
(0)
else
Seize (ad(icr),mode) TO DECPID (dec(ad(icr)))
Rxdata (ad(icr))
5
typ(scr):= 'voice'
ic(scr):=icr, sat(icr):=scr
ic(k):=0, typ(k):='disc'
(k>63)
(false)
(true)
/* Check if previous connection of icr was to an overload channel */
Delete_ overload(k)
(ad(icr))
/* Retain old decoder */
(0)
else
Seizev (ad(icr)) TO DECPID (dec(ad(icr)))
5
T1510680-92

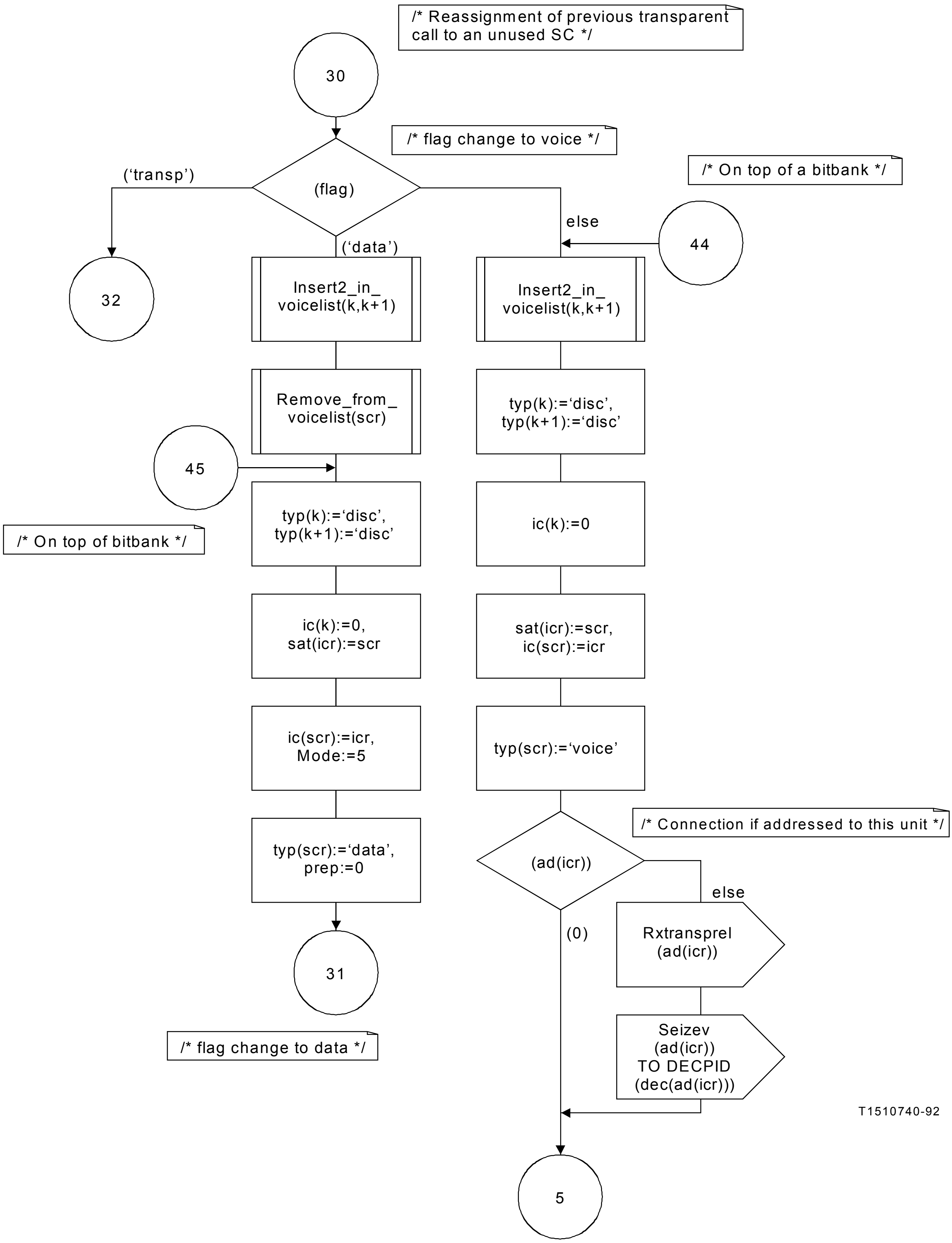

PROCESS RUD
18 (80)
/* Reassignment of previous voice call to an unused SC as a transparent call */
23
(typ(scr+1))
('data')
('bank')
else
Remove_from_ voicelist (scr)
Remove_from_ voicelist (scr)
Remove2_from_ voicelist (scr,scr+1)
Remove_from_ bitbankarray (scr+1,btot)
37
/* Reassignment of previous voice call on top of a bitbank as a transparent call */
37
typ(scr):='transp', typ(scr+1):='transp', nr1:=ic(scr+1)
ic(scr):=icr, sat(icr):=scr mode:=8
(true)
(nr=icr)
(false)
25
ic(k):=0, typ(k):='disc'
/* Skip some actions */
(false)
(k>63)
(true)
Delete_ overload(k)
24
/* Check for reassignment from overload */
T1510690-92


24
/* Check use of nr1 */
(nr1)
else
(0)
(ad(nr1))
else
Release
TO DECPID
(dec(ad(nr1)))
(0)
25
dec(ad(nr1)):=0
ic(scr+1):=0
sat(nr1):=0,
ic(scr+1):=0
/* Connect transparent call
if addressed to this unit */
(ad(icr))
else
Seize
(ad(icr),mode)
TO DECPID
(dec(ad(icr)))
(0)
Rxtransp
(ad(nr))
5
T1510700-92

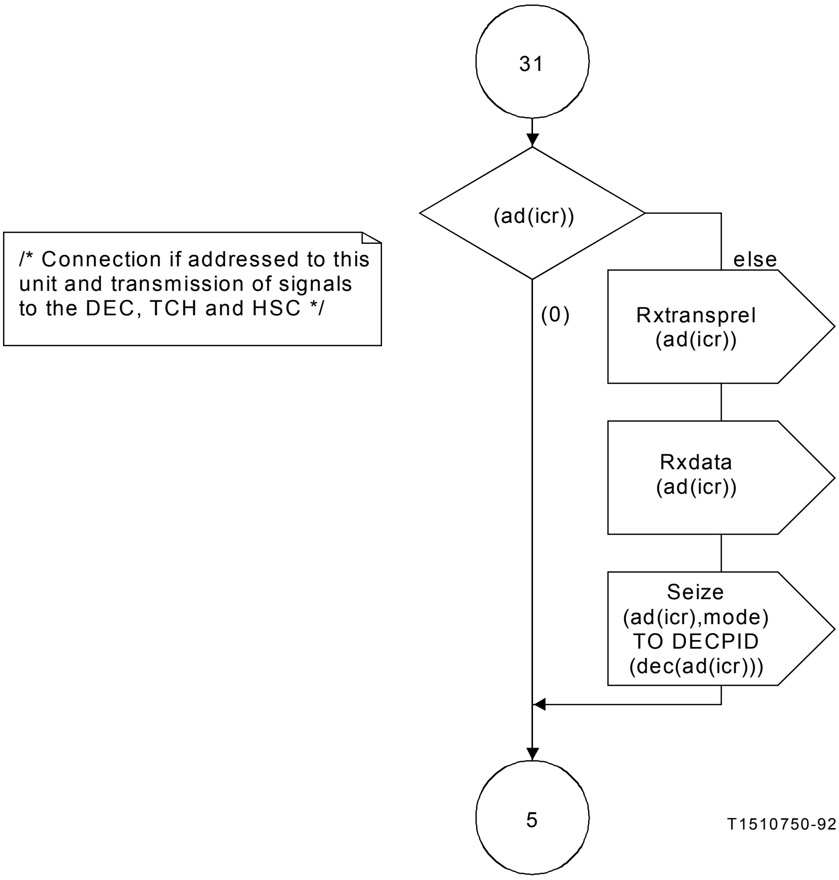

/* Reassignment of previous data call to an unused SC */
26
/* flag change to voice */
/* Reassigned only */
(flag)
('data')
('transp')
else
/* Reassignment on top of a bitbank */
Remove_from_ voicelist(scr)
27
/* Assignment on top of a bitbank */
40
39
ic(scr):=icr, sat(icr):=scr, prep:=0
Insert_in_ voicelist(k)
ic(k):=0, typ(k):='disc'
typ(scr):= 'voice'
Insert_in_ voicelist(k)
ic(scr):=icr, sat(icr):=scr
typ(scr):='data', mode:=5
ic(k):=0, typ(k):='disc'
(ad(icr))
else
Seize (ad(icr),mode) TO DECPID (dec(ad(icr)))
(0)
(ad(icr))
else
Seizev (ad(icr)) TO DECPID (dec(ad(icr)))
(0)
5
5
T1510710-92

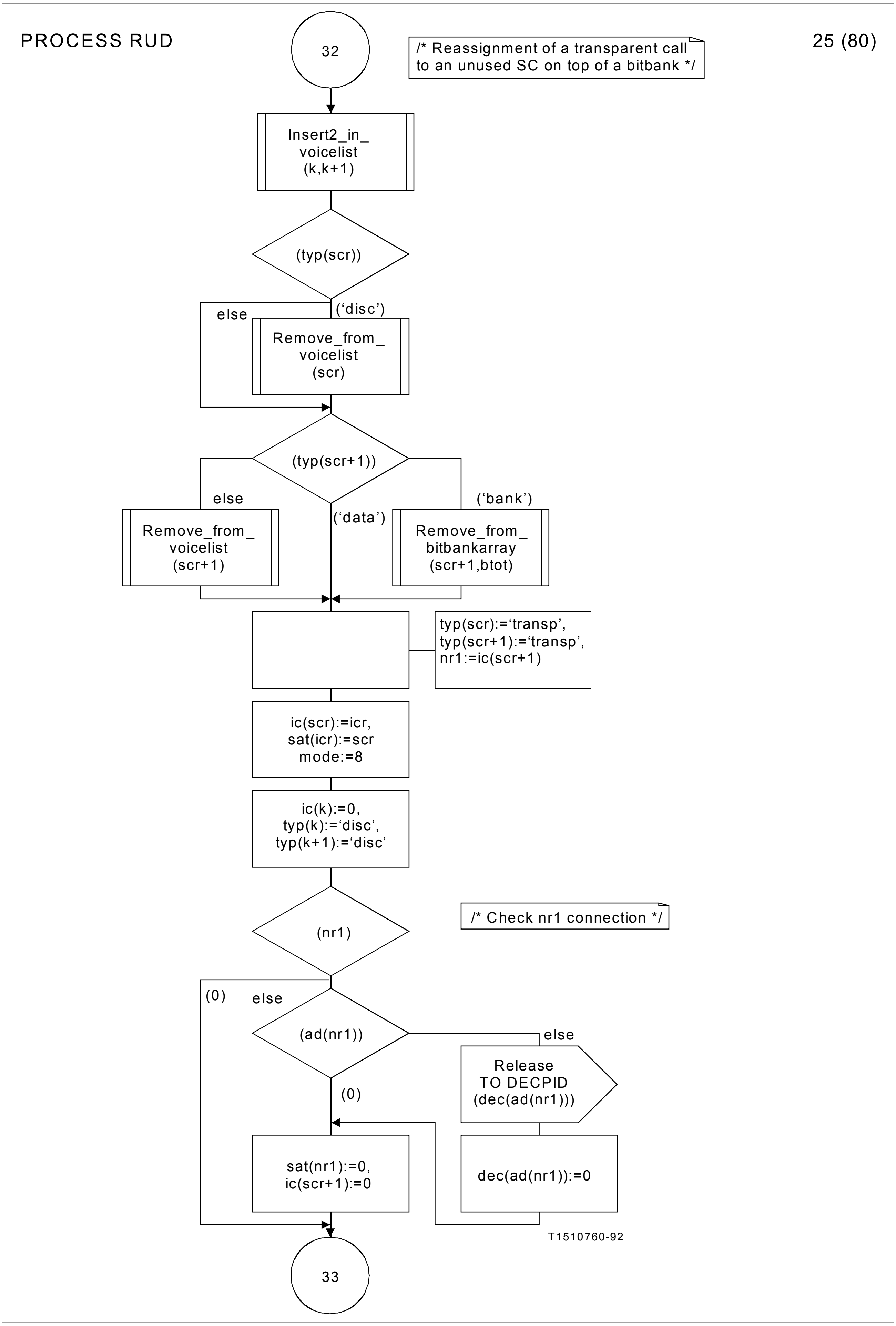

/* Assignment of previous data call to an unused SC as transparent */
27
(typ(scr+1))
('data')
('bank')
else
Remove_from_ voicelist (scr)
Remove_from_ voicelist (scr)
Remove2_from_ voicelist (scr,scr+1)
Remove_from_ bitbankarray (scr+1,btot)
42
/* Assignment of previous data call to an unused SC as transparent on top of a bitbank */
42
typ(scr):='transp', typ(scr+1):='transp', nr1:=ic(scr+1)
ic(scr):=icr, sat(icr):=scr mode:=8
(nr=icr)
(true)
(false)
29
ic(k):=0, typ(k):='disc'
Insert_in_ voicelist(k)
28
T1510720-92

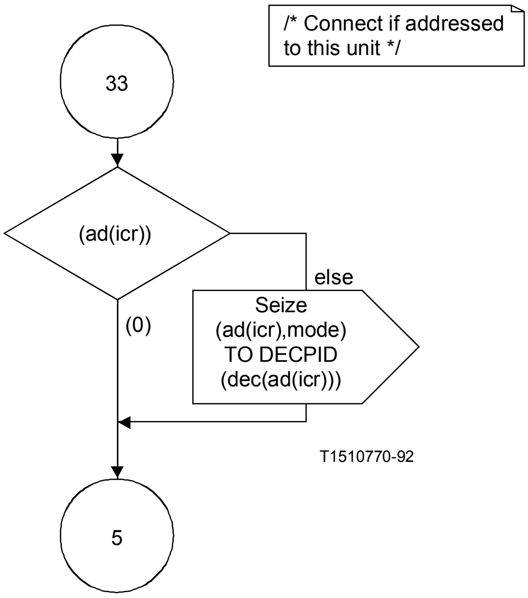

28
/* Check of nr1 connection */
(nr1)
else
(0)
(ad(nr1))
else
(0)
Release
TO DECPID
(dec(ad(nr1)))
dec(ad(nr1)):=0
29
sat(nr1):=0,
ic(scr+1):=0
ic(scr+1):=0
(ad(icr))
/* Connection of transparent call */
else
Seize
(ad(icr),mode)
TO DECPID
(dec(ad(icr)))
(0)
Rxtranspr
(ad(icr))
5
T1510730-92

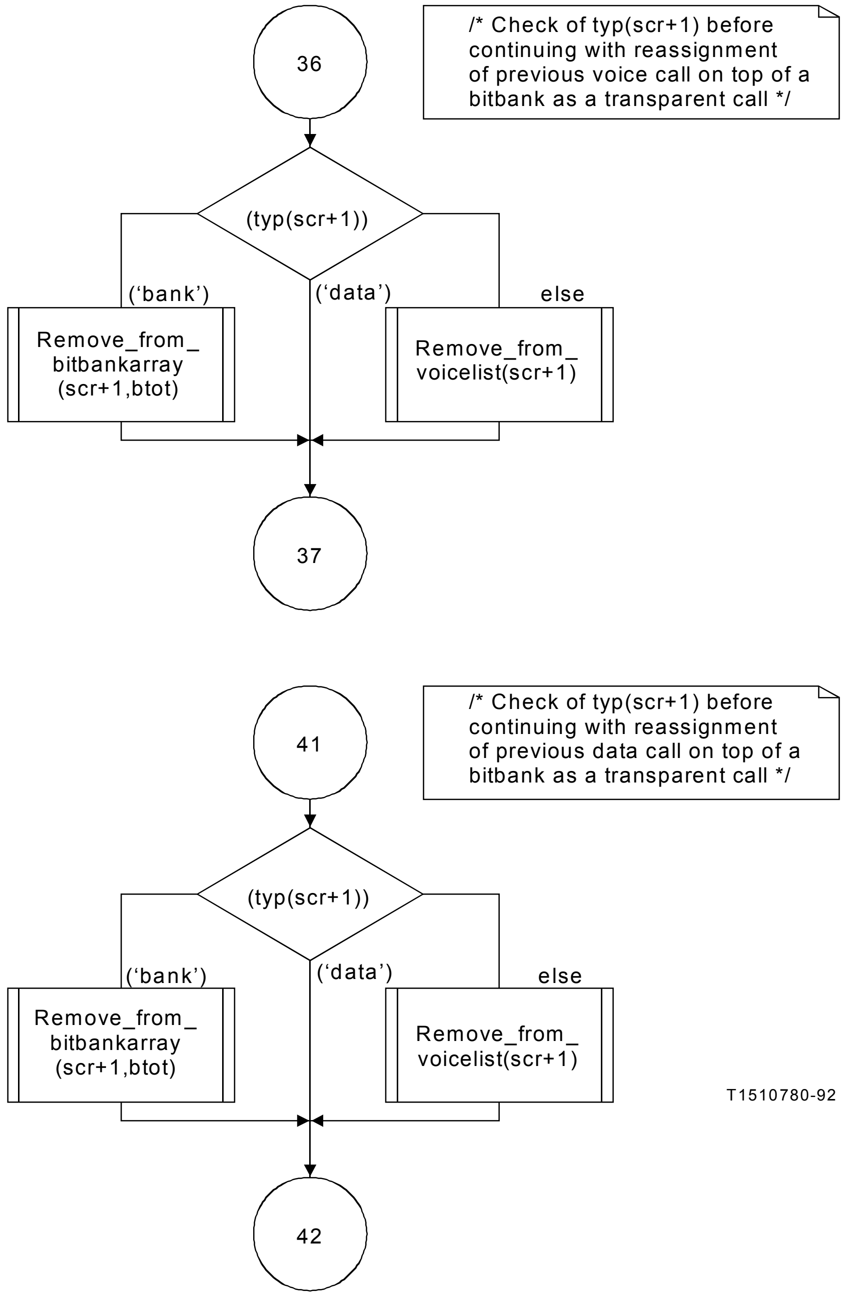

/* Reassignment of previous transparent call to an unused SC */
30
/* flag change to voice */
(flag)
('transp')
('data')
else
/* On top of a bitbank */
44
32
Insert2_in_ voicelist(k,k+1)
Insert2_in_ voicelist(k,k+1)
Remove_from_ voicelist(scr)
typ(k):='disc', typ(k+1):='disc'
45
typ(k):='disc', typ(k+1):='disc'
/* On top of bitbank */
ic(k):=0
ic(k):=0, sat(icr):=scr
sat(icr):=scr, ic(scr):=icr
ic(scr):=icr, Mode:=5
typ(scr):='voice'
/* Connection if addressed to this unit */
typ(scr):='data', prep:=0
(ad(icr))
else
(0)
Rxtransprel (ad(icr))
31
Seizev (ad(icr)) TO DECPID (dec(ad(icr)))
/* flag change to data */
5

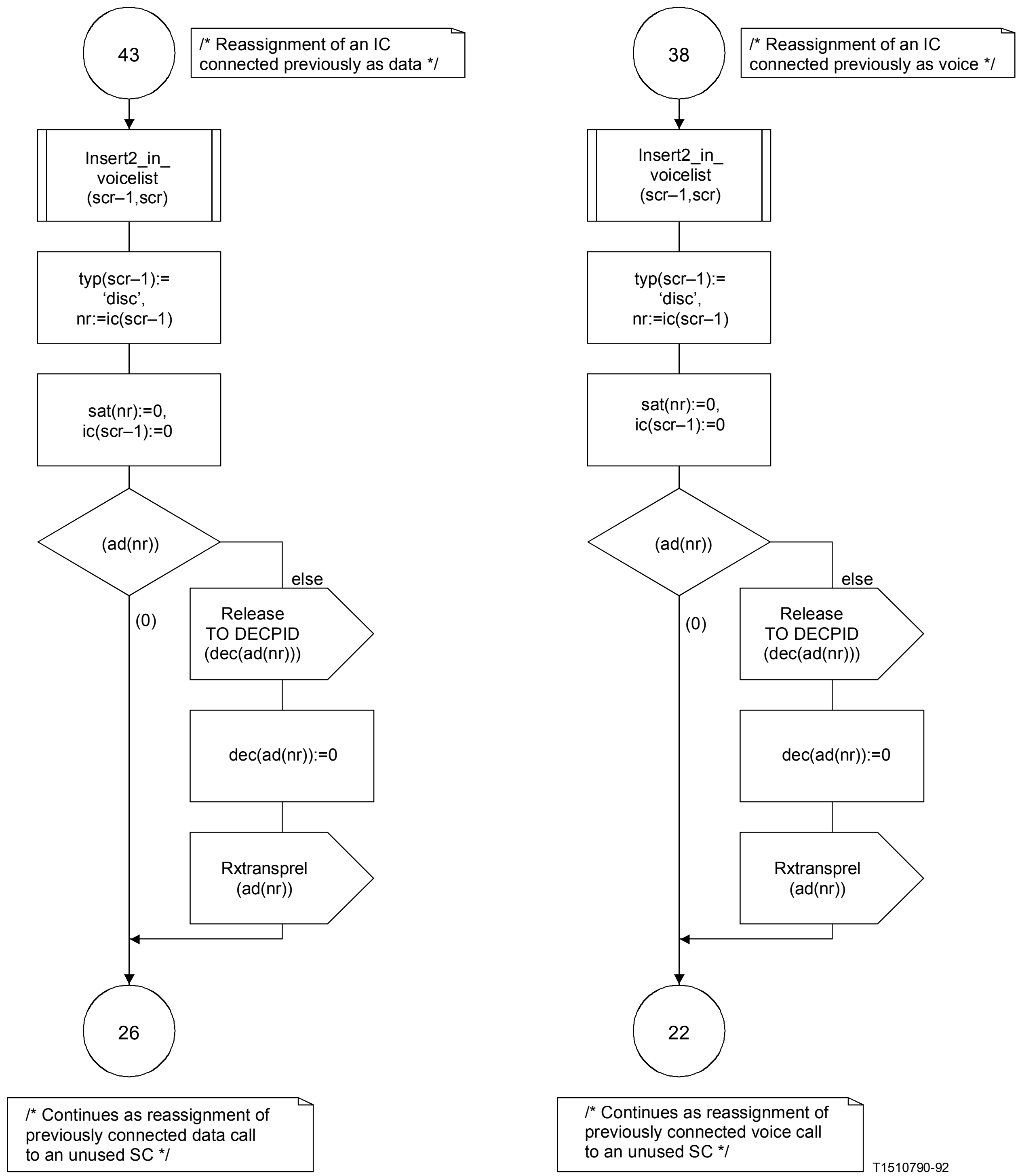
T1510740-92

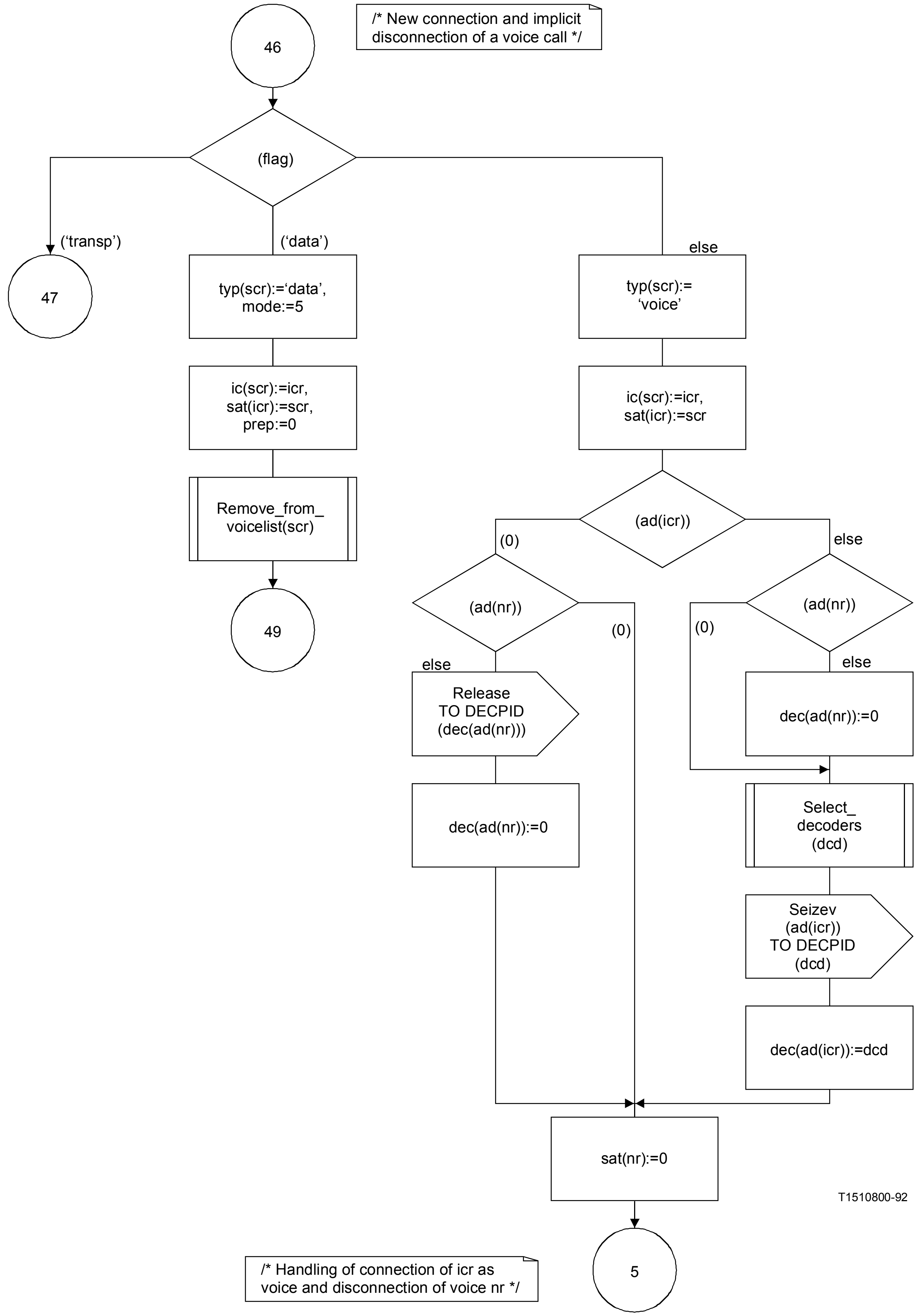

31
(ad(icr))
/* Connection if addressed to this unit and transmission of signals to the DEC, TCH and HSC */
(0)
else
Rxtransprel (ad(icr))
Rxdata (ad(icr))
Seize (ad(icr),mode) TO DECPID (dec(ad(icr)))
5
T1510750-92

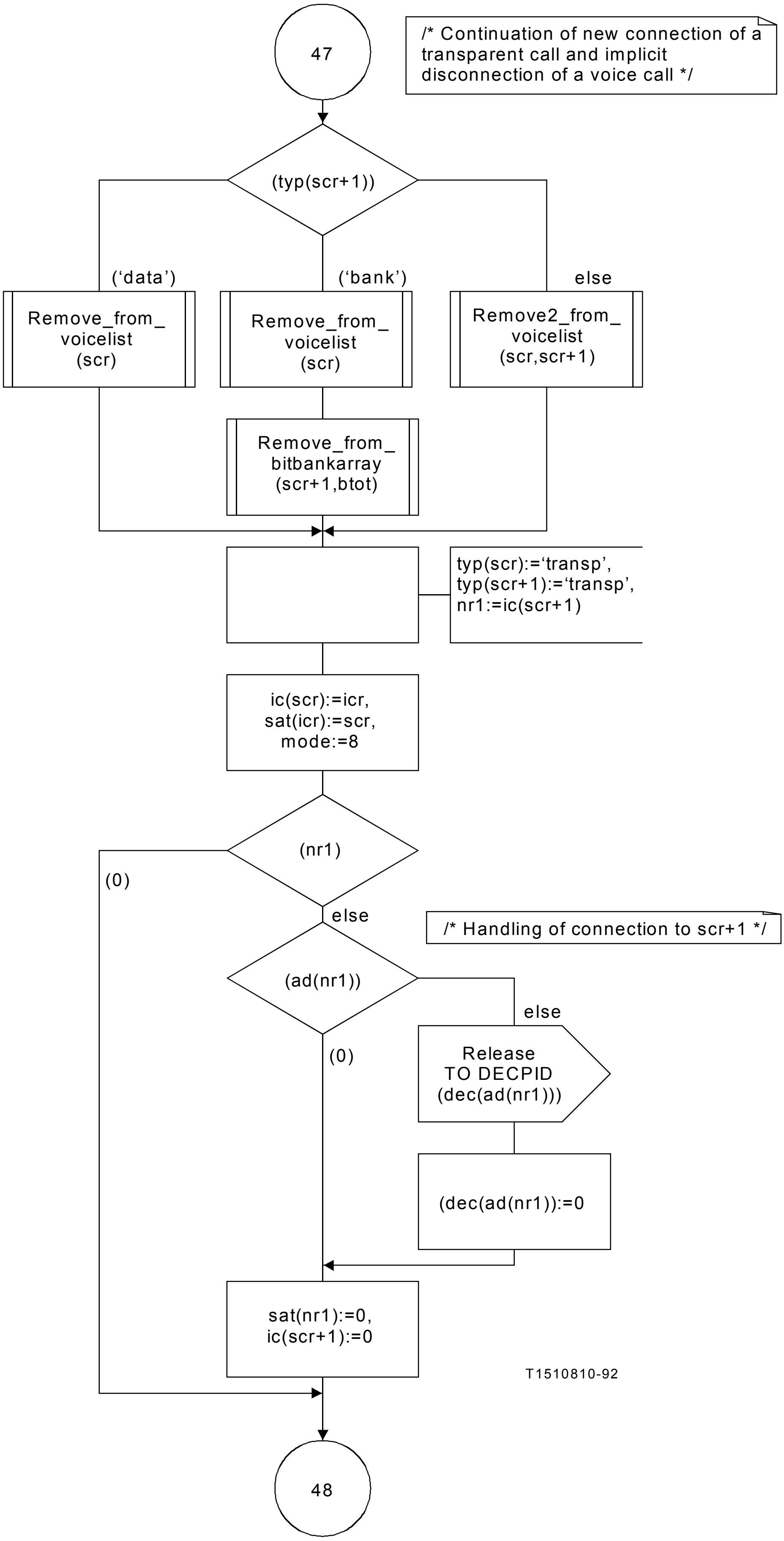

PROCESS RUD
25 (80)
32
/* Reassignment of a transparent call to an unused SC on top of a bitbank */
Insert2_in_ voicelist (k,k+1)
(typ(scr))
else
('disc')
Remove_from_ voicelist (scr)
(typ(scr+1))
else
('data')
('bank')
Remove_from_ voicelist (scr+1)
Remove_from_ bitbankarray (scr+1,btot)
typ(scr):='transp', typ(scr+1):='transp', nr1:=ic(scr+1)
ic(scr):=icr, sat(icr):=scr mode:=8
ic(k):=0, typ(k):='disc', typ(k+1):='disc'
(nr1)
/* Check nr1 connection */
(0)
else
(ad(nr1))
else
(0)
Release TO DECPID (dec(ad(nr1)))
sat(nr1):=0, ic(scr+1):=0
dec(ad(nr1)):=0
T1510760-92
33

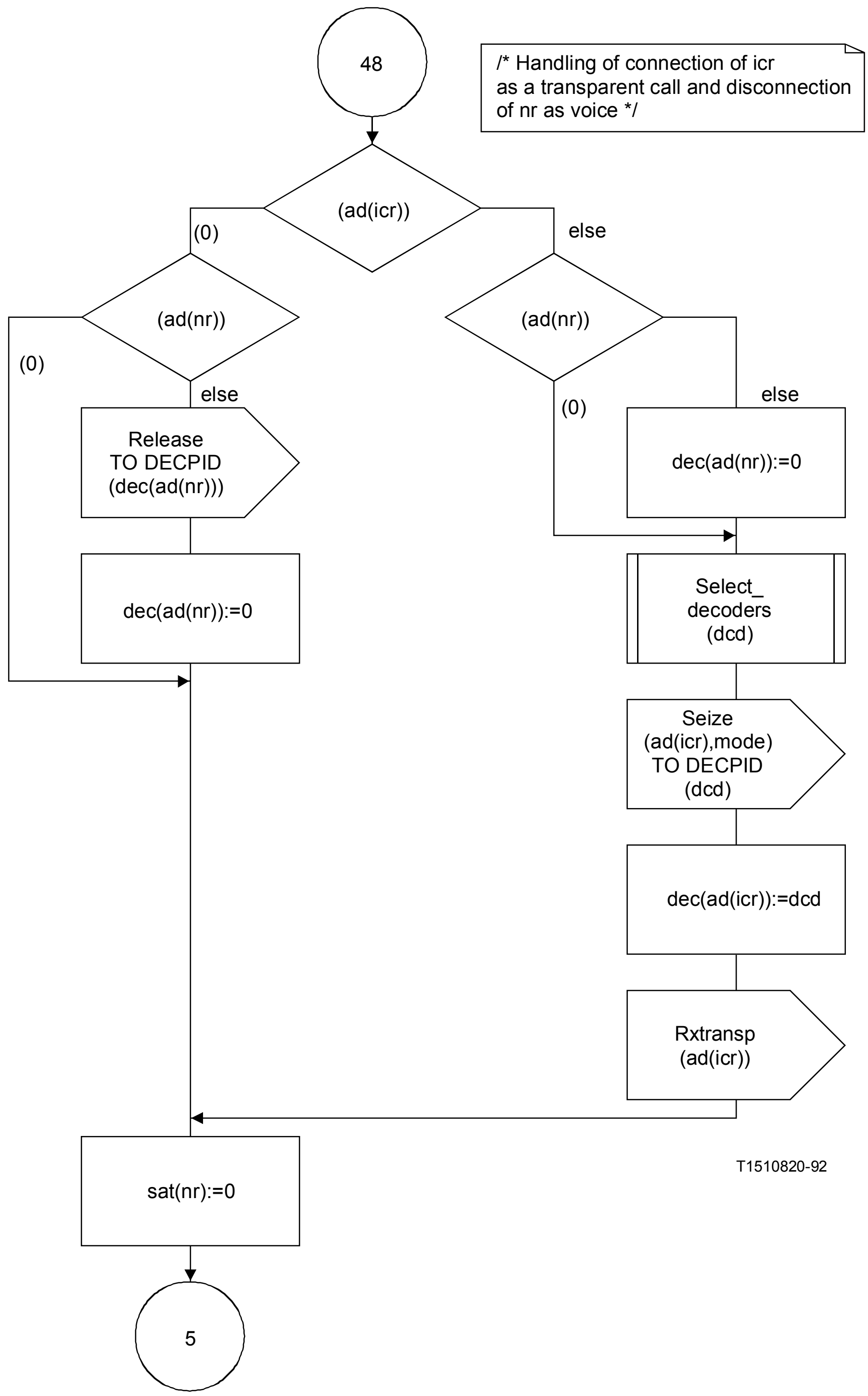

/* Connect if addressed to this unit */
33
(ad(icr))
else
(0)
Seize (ad(icr),mode) TO DECPID (dec(ad(icr)))
T1510770-92
5

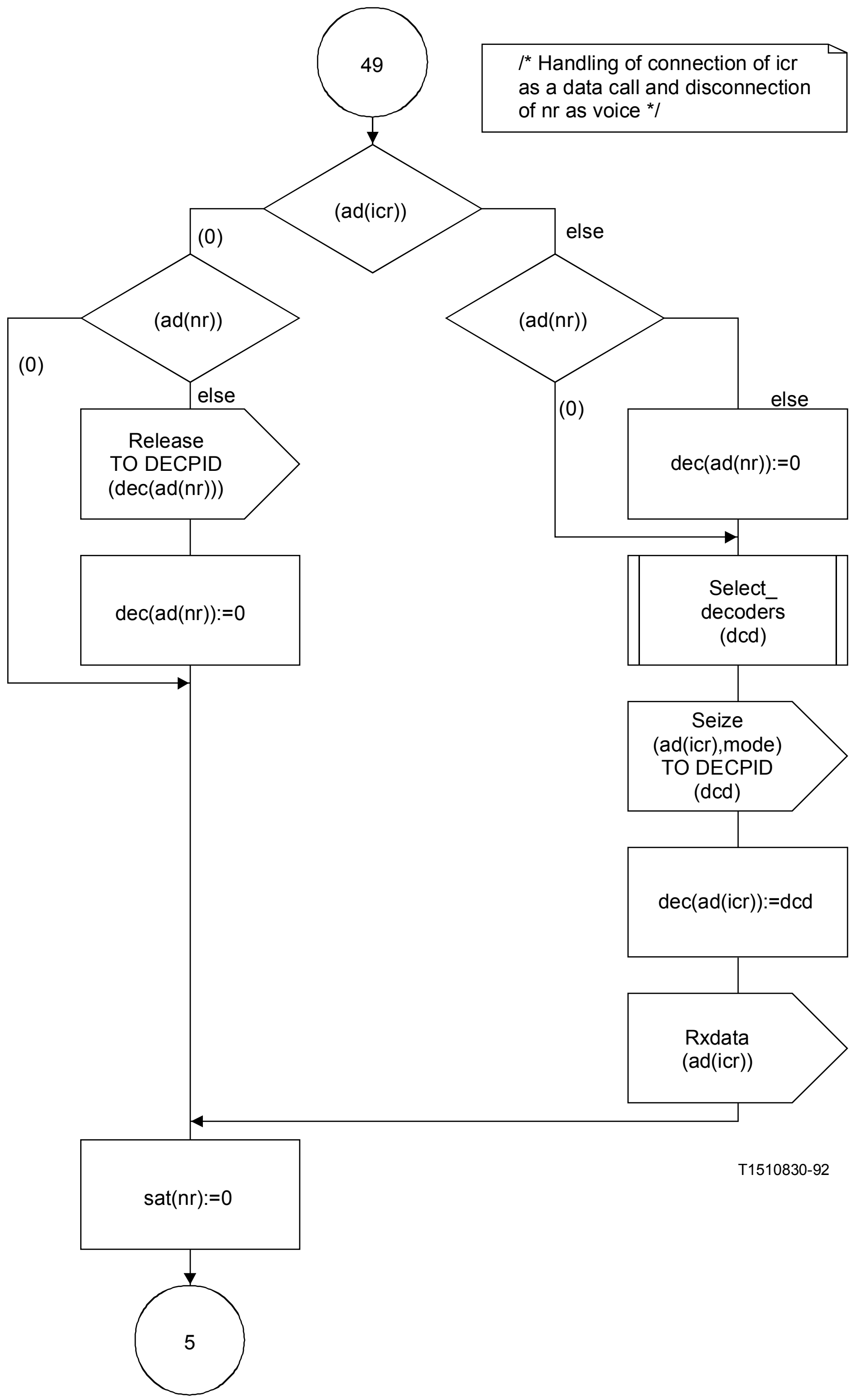

36
/* Check of typ(scr+1) before continuing with reassignment of previous voice call on top of a bitbank as a transparent call */
(typ(scr+1))
('bank')
('data')
else
Remove_from_ bitbankarray (scr+1,btot)
Remove_from_ voicelist(scr+1)
37
41
/* Check of typ(scr+1) before continuing with reassignment of previous data call on top of a bitbank as a transparent call */
(typ(scr+1))
('bank')
('data')
else
Remove_from_ bitbankarray (scr+1,btot)
Remove_from_ voicelist(scr+1)
T1510780-92
42

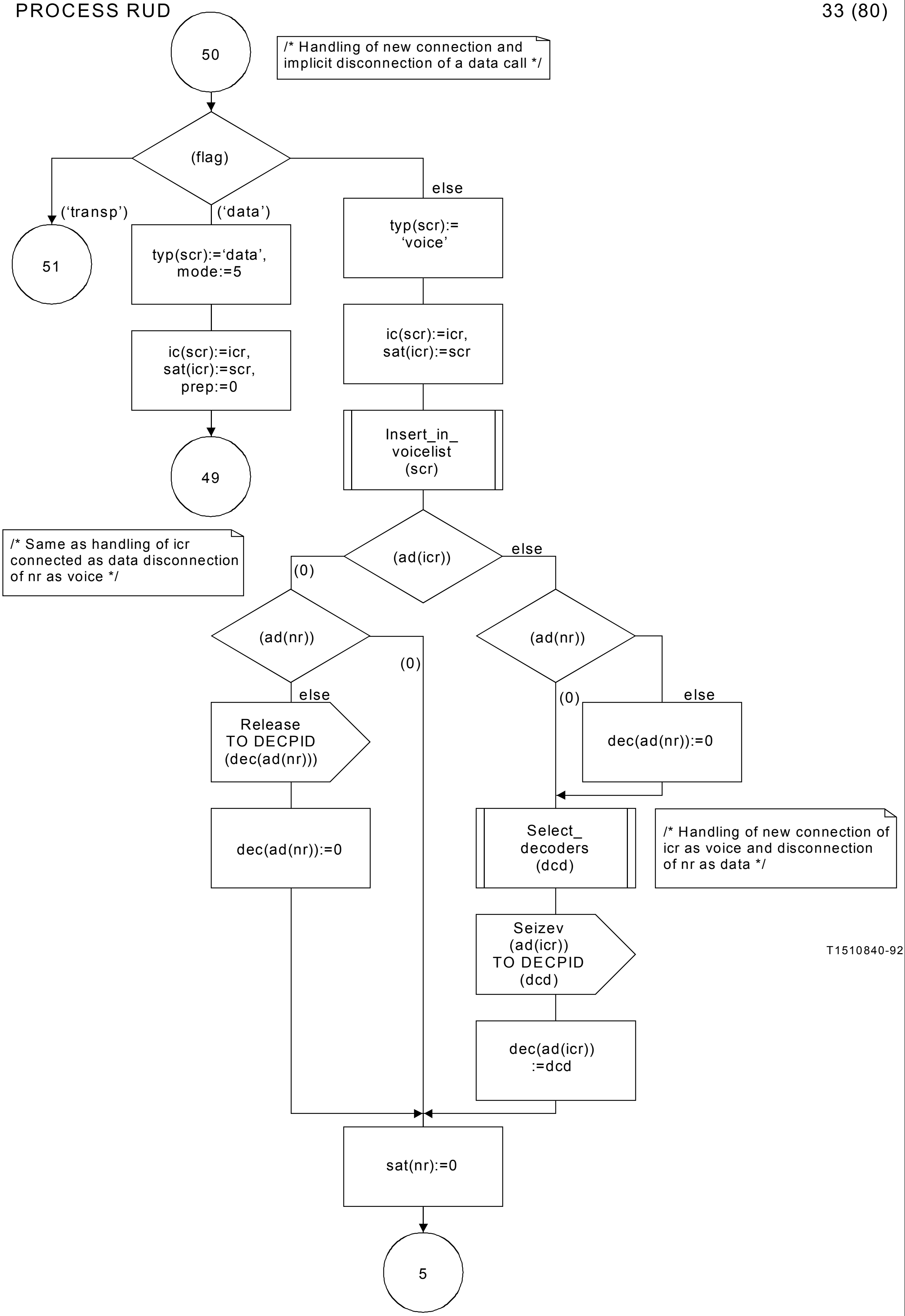

43
/* Reassignment of an IC connected previously as data */
Insert2_in_ voicelist (scr–1,scr)
typ(scr–1):= 'disc', nr:=ic(scr–1)
sat(nr):=0, ic(scr–1):=0
(ad(nr))
else
(0)
Release TO DECPID (dec(ad(nr)))
dec(ad(nr)):=0
Rxtransprel (ad(nr))
26
/* Continues as reassignment of previously connected data call to an unused SC */
38
/* Reassignment of an IC connected previously as voice */
Insert2_in_ voicelist (scr–1,scr)
typ(scr–1):= 'disc', nr:=ic(scr–1)
sat(nr):=0, ic(scr–1):=0
(ad(nr))
else
(0)
Release TO DECPID (dec(ad(nr)))
dec(ad(nr)):=0
Rxtransprel (ad(nr))
22
/* Continues as reassignment of previously connected voice call to an unused SC */
T1510790-92

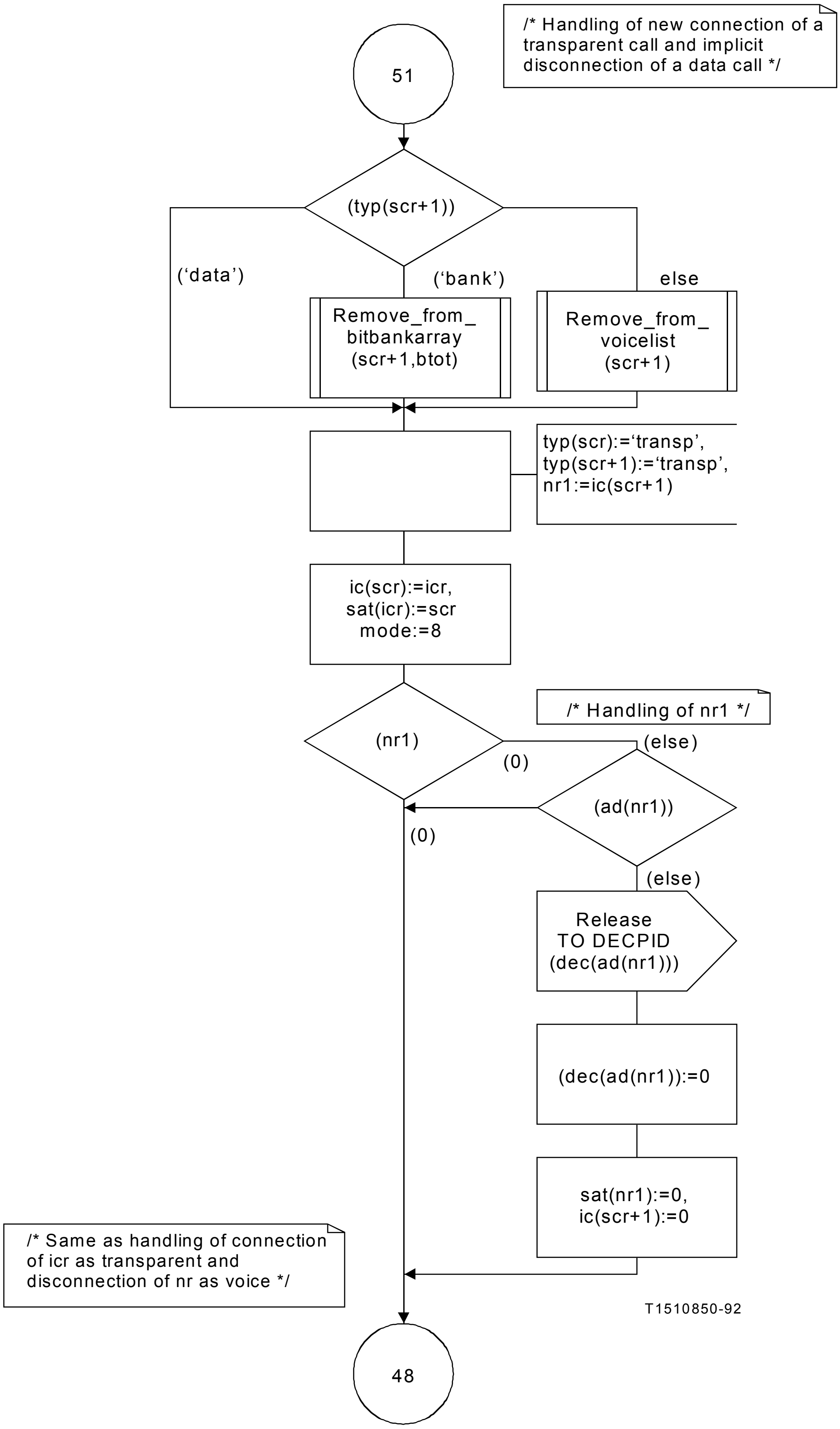

/* New connection and implicit disconnection of a voice call */
46
(flag)
('transp')
47
('data')
typ(scr):='data', mode:=5
ic(scr):=icr, sat(icr):=scr, prep:=0
Remove_from_ voicelist(scr)
49
else
typ(scr):= 'voice'
ic(scr):=icr, sat(icr):=scr
(ad(icr))
(0)
else
(ad(nr))
(0)
else
Release TO DECPID (dec(ad(nr)))
dec(ad(nr)):=0
(ad(nr))
(0)
else
dec(ad(nr)):=0
Select_ decoders (dcd)
Seizev (ad(icr)) TO DECPID (dcd)
dec(ad(icr)):=dcd
sat(nr):=0
T1510800-92
5
/* Handling of connection of icr as voice and disconnection of voice nr */

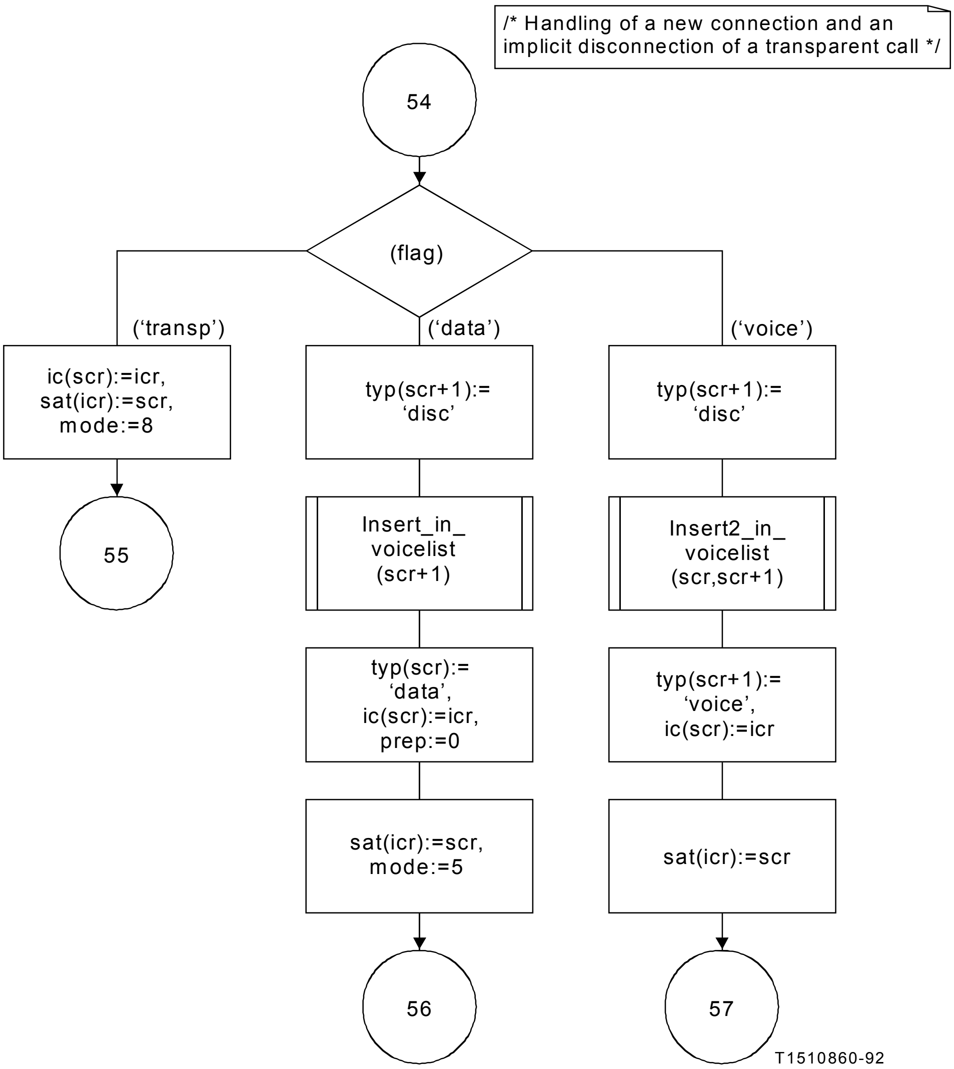

47
/* Continuation of new connection of a transparent call and implicit disconnection of a voice call */
(typ(scr+1))
('data')
('bank')
else
Remove_from_ voicelist (scr)
Remove_from_ voicelist (scr)
Remove2_from_ voicelist (scr,scr+1)
Remove_from_ bitbankarray (scr+1,btot)
typ(scr):='transp', typ(scr+1):='transp', nr1:=ic(scr+1)
ic(scr):=icr, sat(icr):=scr, mode:=8
(nr1)
(0)
else
/* Handling of connection to scr+1 */
(ad(nr1))
else
(0)
Release TO DECPID (dec(ad(nr1)))
(dec(ad(nr1)):=0
sat(nr1):=0, ic(scr+1):=0
T1510810-92
48

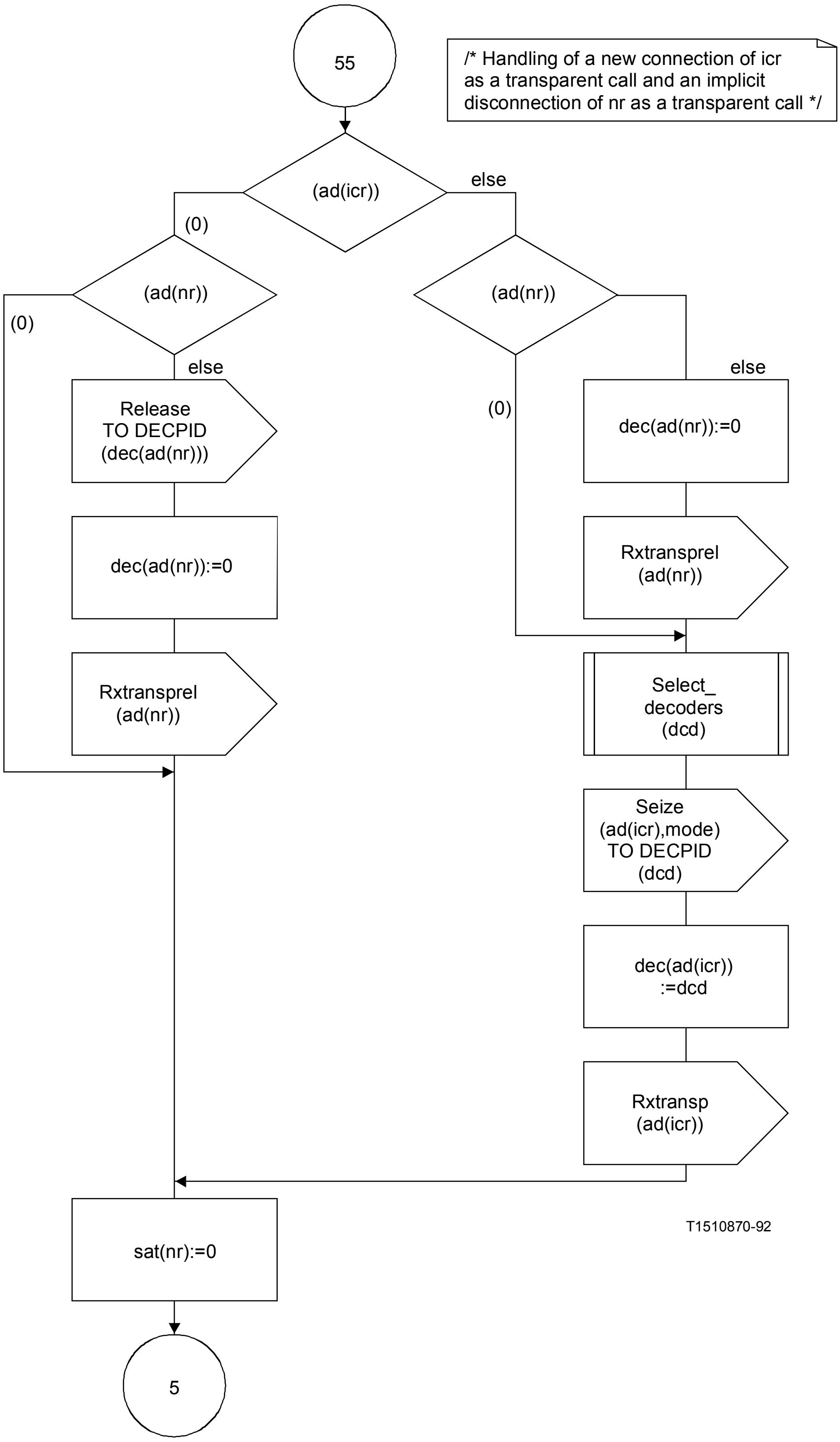

48
/* Handling of connection of icr
as a transparent call and disconnection
of nr as voice */
(ad(icr))
(0)
else
(ad(nr))
(0)
else
Release
TO DECPID
(dec(ad(nr)))
dec(ad(nr)):=0
(ad(nr))
(0)
else
dec(ad(nr)):=0
Select_
decoders
(dcd)
Seize
(ad(icr),mode)
TO DECPID
(dcd)
dec(ad(icr)):=dcd
Rxtransp
(ad(icr))
sat(nr):=0
5
T1510820-92

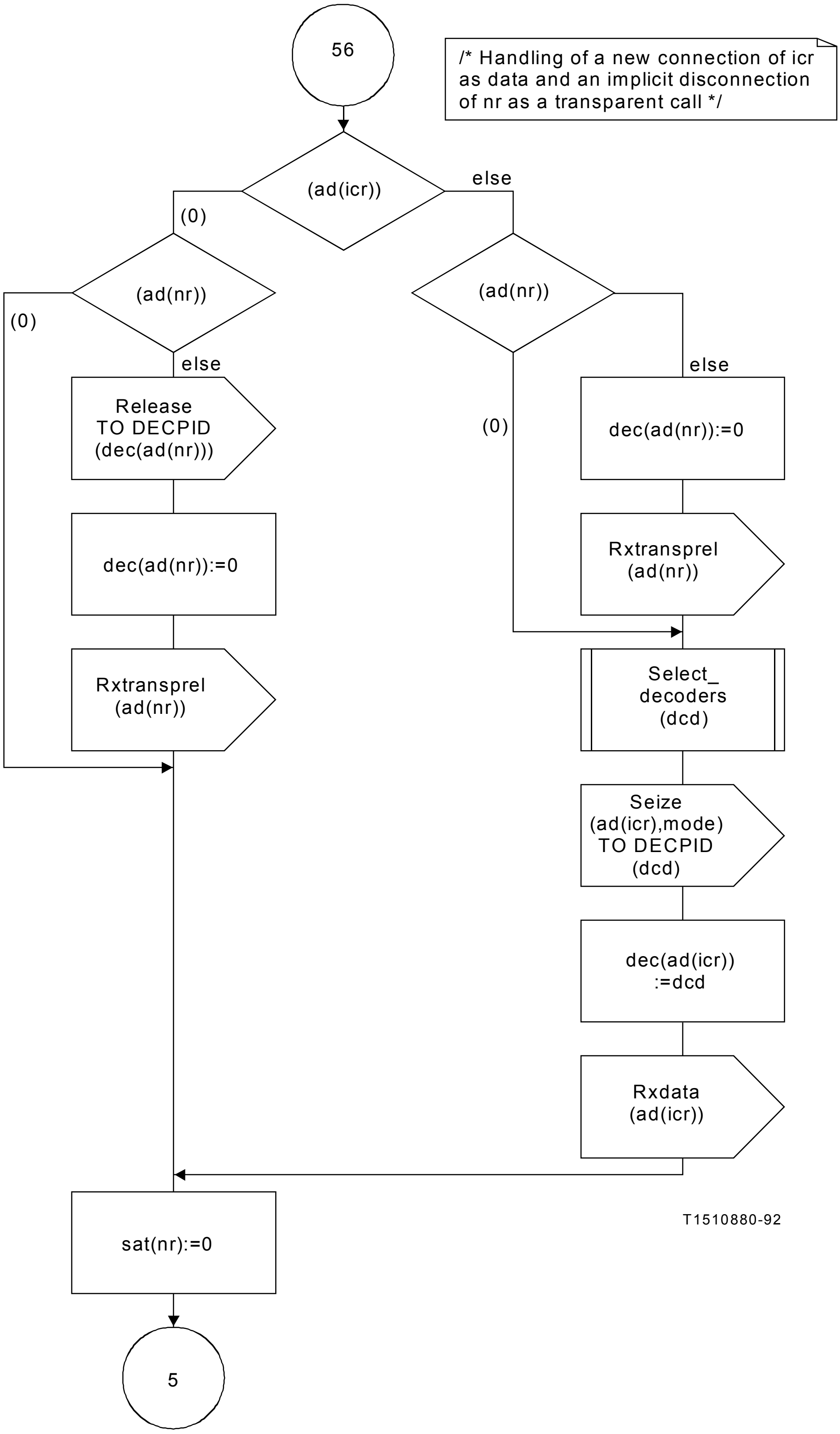

49
/* Handling of connection of icr as a data call and disconnection of nr as voice */
(ad(icr))
(0)
else
(ad(nr))
(ad(nr))
(0)
else
(0)
else
Release TO DECPID (dec(ad(nr)))
dec(ad(nr)):=0
dec(ad(nr)):=0
Select_ decoders (dcd)
Seize (ad(icr),mode) TO DECPID (dcd)
dec(ad(icr)):=dcd
Rxdata (ad(icr))
sat(nr):=0
5
T1510830-92

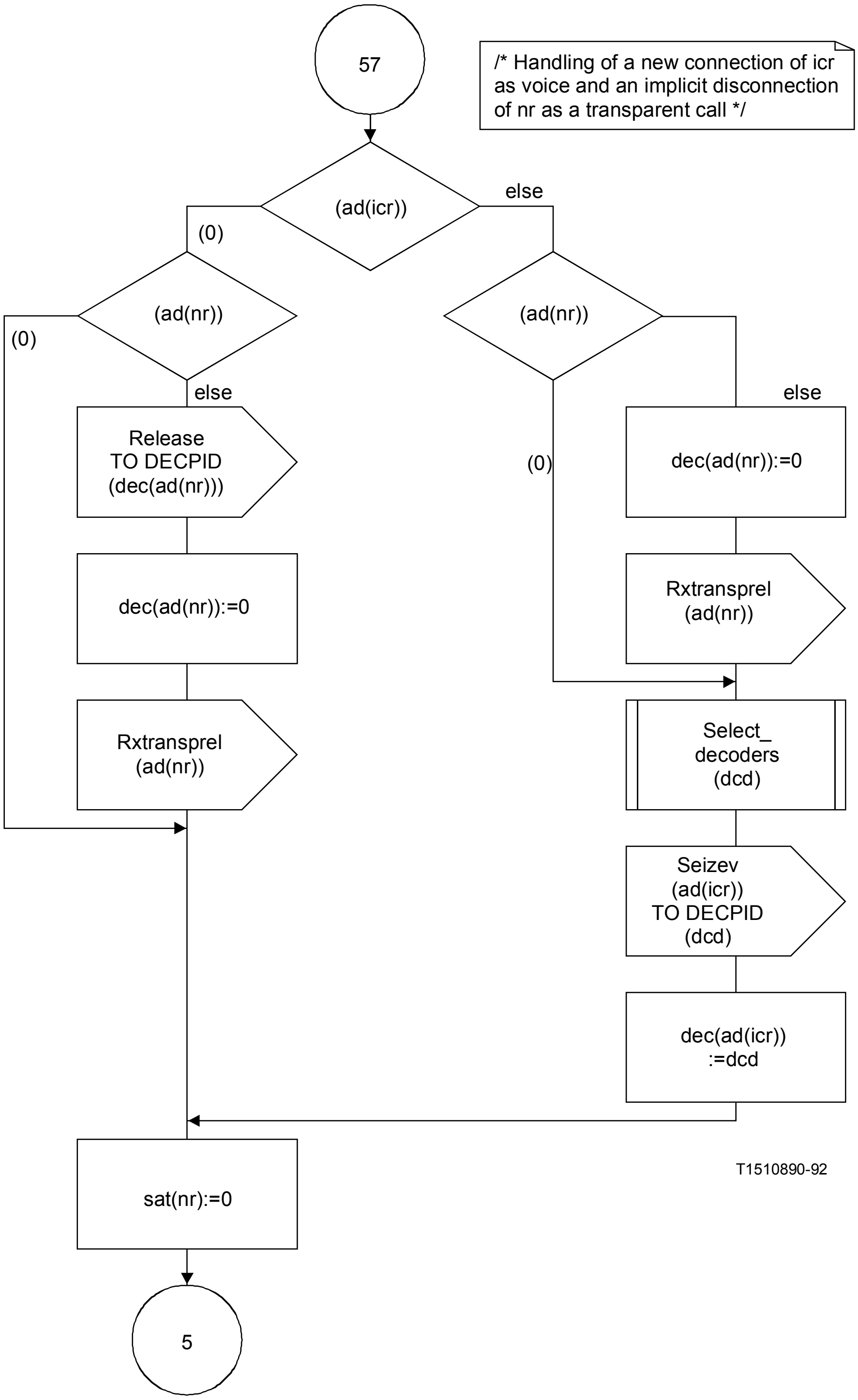

50
/* Handling of new connection and implicit disconnection of a data call */
(flag)
('transp')
('data')
else
51
typ(scr):='data', mode:=5
typ(scr):= 'voice'
ic(scr):=icr, sat(icr):=scr, prep:=0
ic(scr):=icr, sat(icr):=scr
Insert_in_ voicelist (scr)
49
/* Same as handling of icr connected as data disconnection of nr as voice */
(ad(icr))
(0)
else
(ad(nr))
(0)
else
(ad(nr))
(0)
else
Release TO DECPID (dec(ad(nr)))
dec(ad(nr)):=0
dec(ad(nr)):=0
Select_ decoders (dcd)
/* Handling of new connection of icr as voice and disconnection of nr as data */
Seizev (ad(icr)) TO DECPID (dcd)
dec(ad(icr)) :=dcd
sat(nr):=0
5
T1510840-92

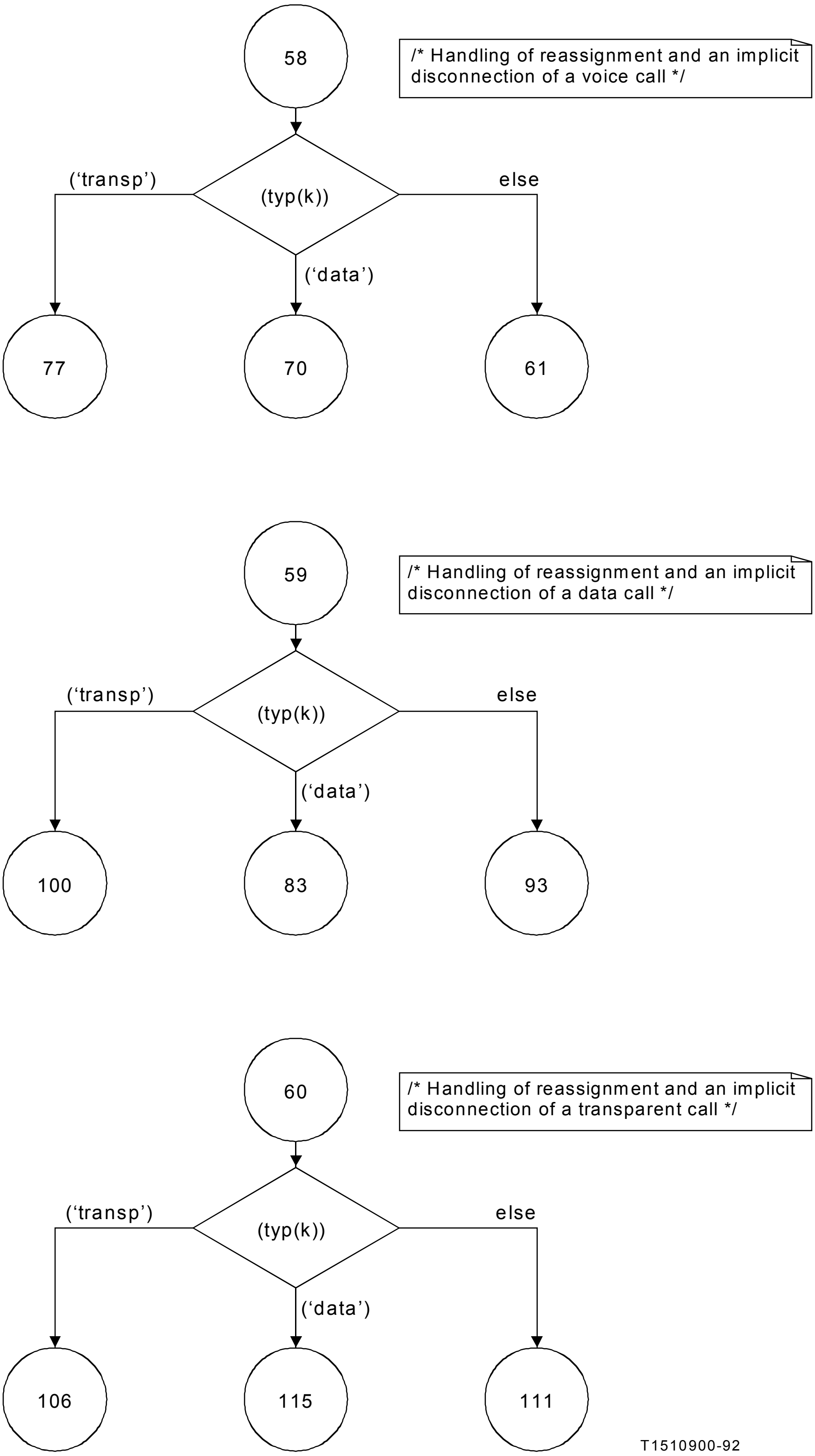

/* Handling of new connection of a transparent call and implicit disconnection of a data call */
51
(typ(scr+1))
('data')
('bank')
else
Remove_from_ bitbankarray (scr+1,btot)
Remove_from_ voicelist (scr+1)
typ(scr):='transp', typ(scr+1):='transp', nr1:=ic(scr+1)
ic(scr):=icr, sat(icr):=scr mode:=8
/* Handling of nr1 */
(nr1)
(else)
(0)
(ad(nr1))
(0)
(else)
Release TO DECPID (dec(ad(nr1)))
(dec(ad(nr1)):=0
sat(nr1):=0, ic(scr+1):=0
/* Same as handling of connection of icr as transparent and disconnection of nr as voice */
T1510850-92
48

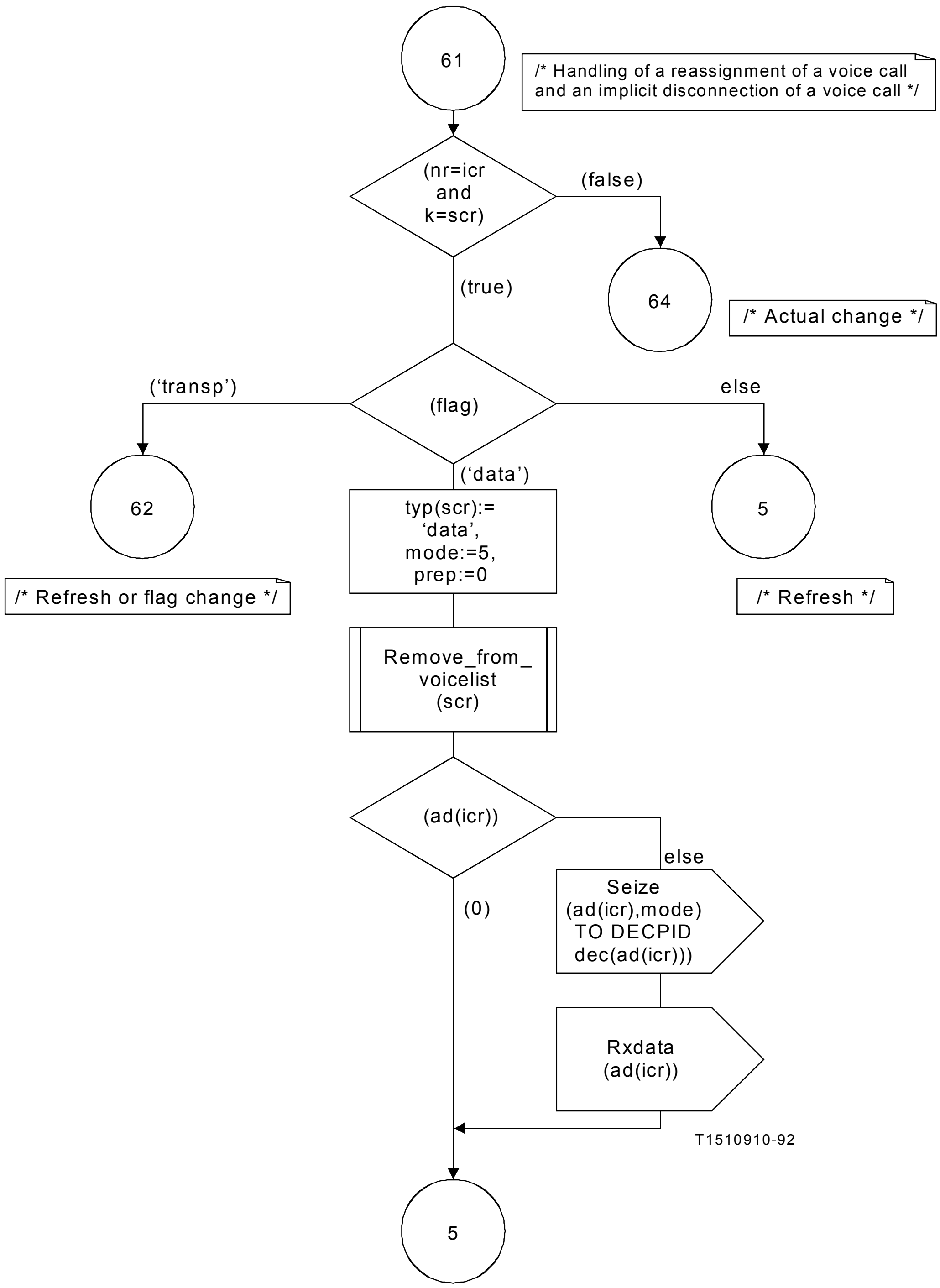

/* Handling of a new connection and an implicit disconnection of a transparent call */
54
(flag)
('transp')
('data')
('voice')
ic(scr):=icr, sat(icr):=scr, mode:=8
55
typ(scr+1):= 'disc'
Insert_in_ voicelist (scr+1)
typ(scr):= 'data', ic(scr):=icr, prep:=0
sat(icr):=scr, mode:=5
56
typ(scr+1):= 'disc'
Insert2_in_ voicelist (scr,scr+1)
typ(scr+1):= 'voice', ic(scr):=icr
sat(icr):=scr
57
T1510860-92

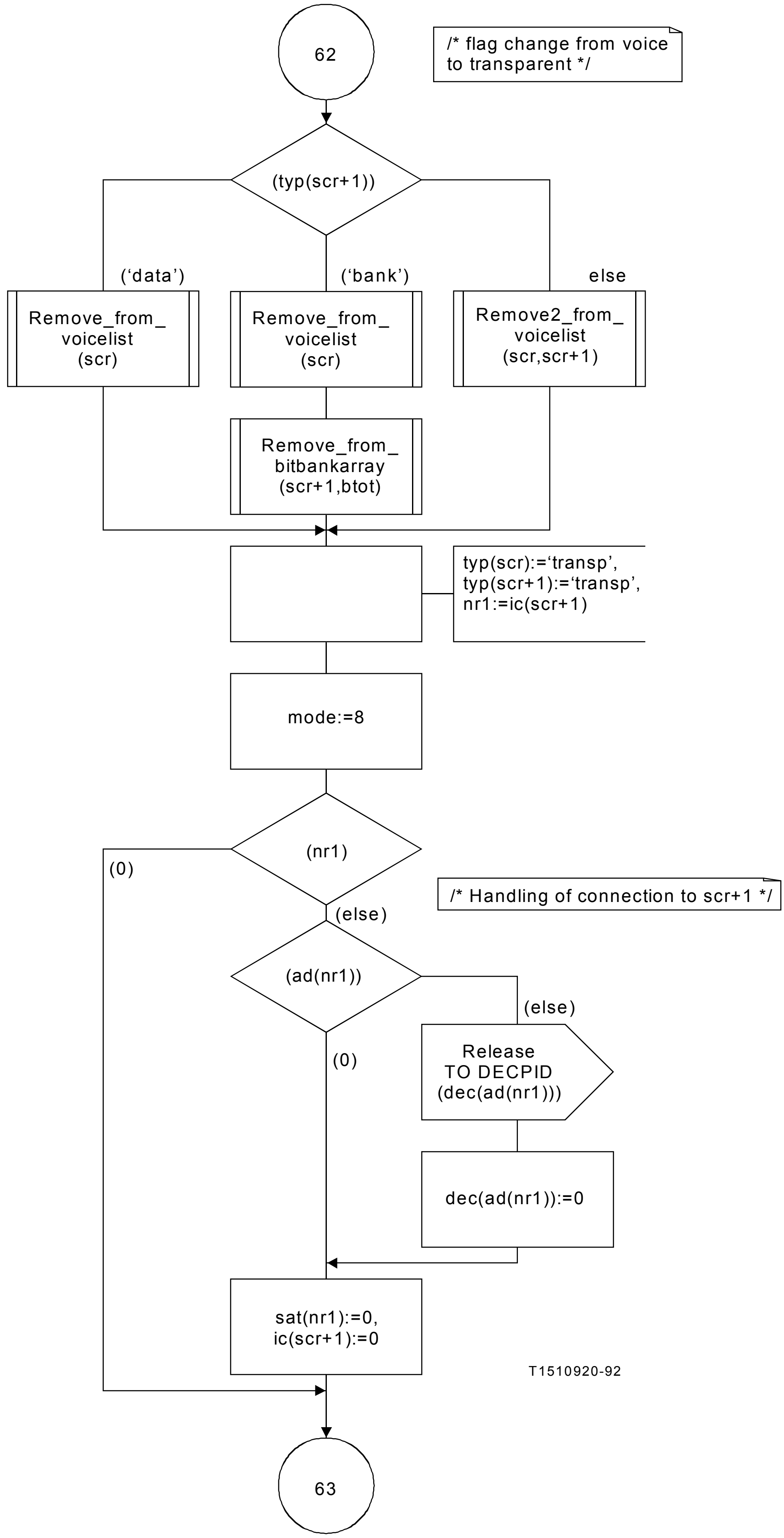

55
/* Handling of a new connection of icr as a transparent call and an implicit disconnection of nr as a transparent call */
(ad(icr))
else
(0)
(ad(nr))
(0)
else
Release TO DECPID (dec(ad(nr)))
dec(ad(nr)):=0
Rxtransprel (ad(nr))
(ad(nr))
else
(0)
dec(ad(nr)):=0
Rxtransprel (ad(nr))
Select_ decoders (dcd)
Seize (ad(icr),mode) TO DECPID (dcd)
dec(ad(icr)) :=dcd
Rxtransp (ad(icr))
sat(nr):=0
5
T1510870-92

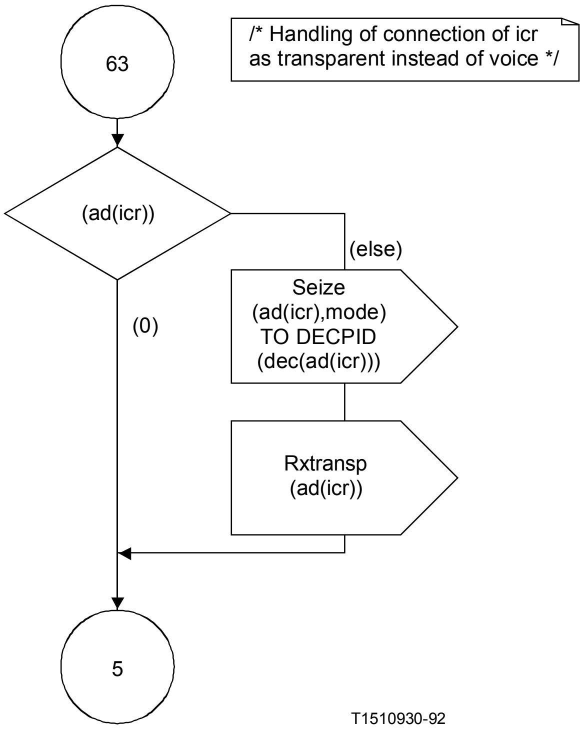

56
/* Handling of a new connection of icr as data and an implicit disconnection of nr as a transparent call */
(ad(icr))
(0)
else
(ad(nr))
(0)
else
Release TO DECPID (dec(ad(nr)))
dec(ad(nr)):=0
Rxtransprel (ad(nr))
(ad(nr))
(0)
else
dec(ad(nr)):=0
Rxtransprel (ad(nr))
Select_ decoders (dcd)
Seize (ad(icr),mode) TO DECPID (dcd)
dec(ad(icr)) :=dcd
Rxdata (ad(icr))
sat(nr):=0
5
T1510880-92

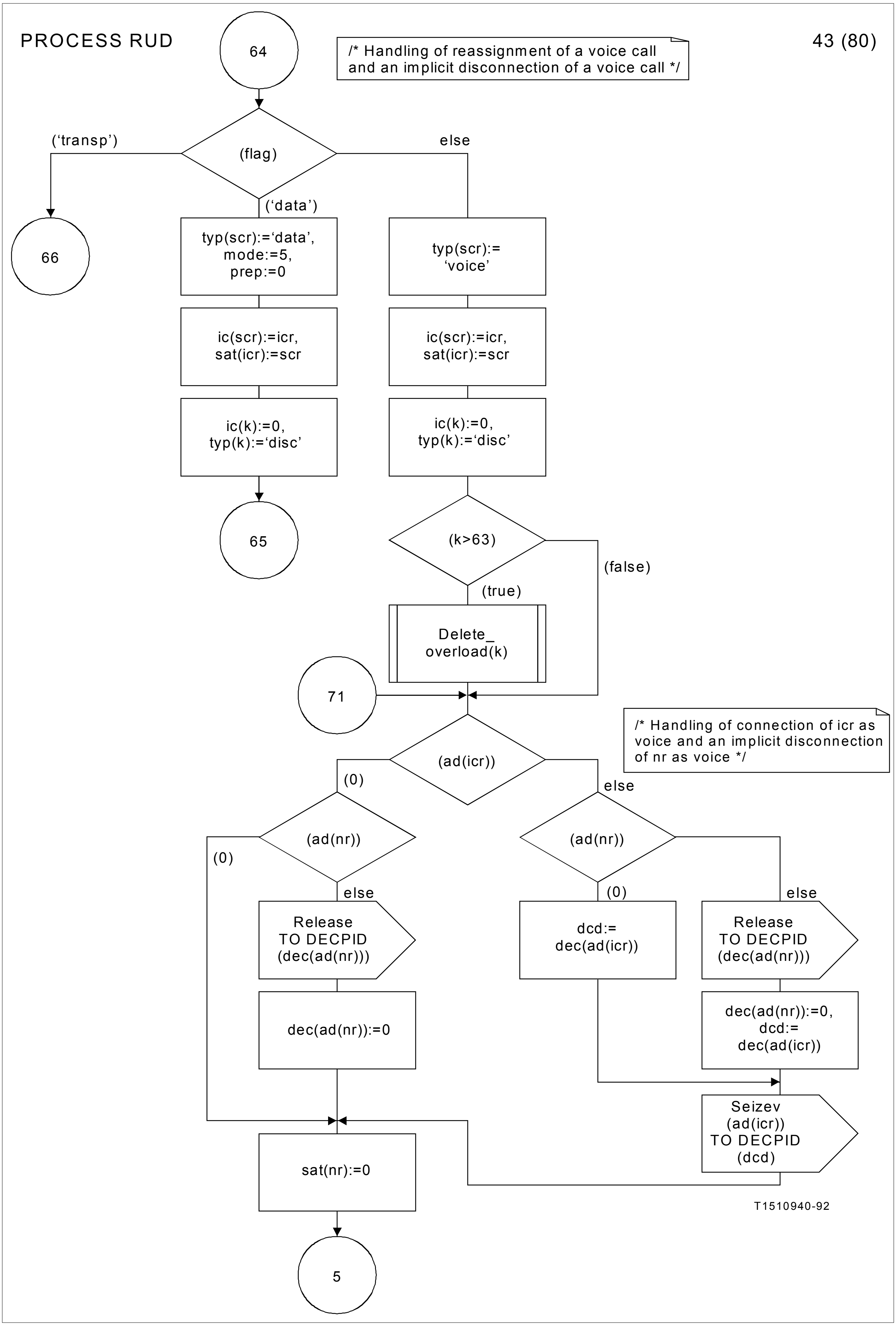

57
/* Handling of a new connection of icr as voice and an implicit disconnection of nr as a transparent call */
(ad(icr))
else
(0)
(ad(nr))
(0)
(ad(nr))
else
else
Release TO DECPID (dec(ad(nr)))
(0)
dec(ad(nr)):=0
dec(ad(nr)):=0
Rxtransprel (ad(nr))
Rxtransprel (ad(nr))
Select_ decoders (dcd)
Seizev (ad(icr)) TO DECPID (dcd)
dec(ad(icr)) :=dcd
sat(nr):=0
5
T1510890-92

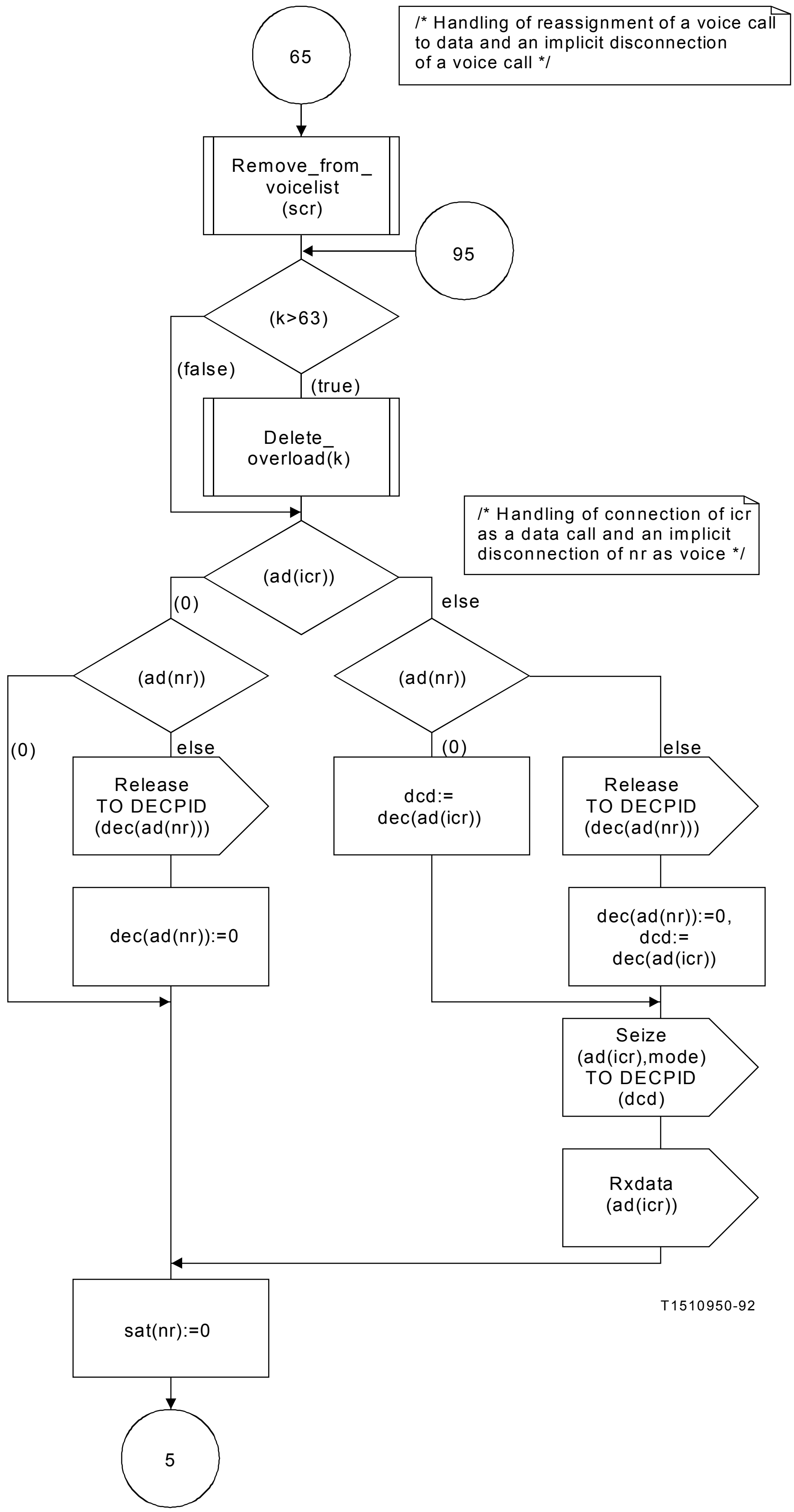

58
/* Handling of reassignment and an implicit disconnection of a voice call */
(typ(k))
('transp')
('data')
else
77
70
61
59
/* Handling of reassignment and an implicit disconnection of a data call */
(typ(k))
('transp')
('data')
else
100
83
93
60
/* Handling of reassignment and an implicit disconnection of a transparent call */
(typ(k))
('transp')
('data')
else
106
115
111
T1510900-92

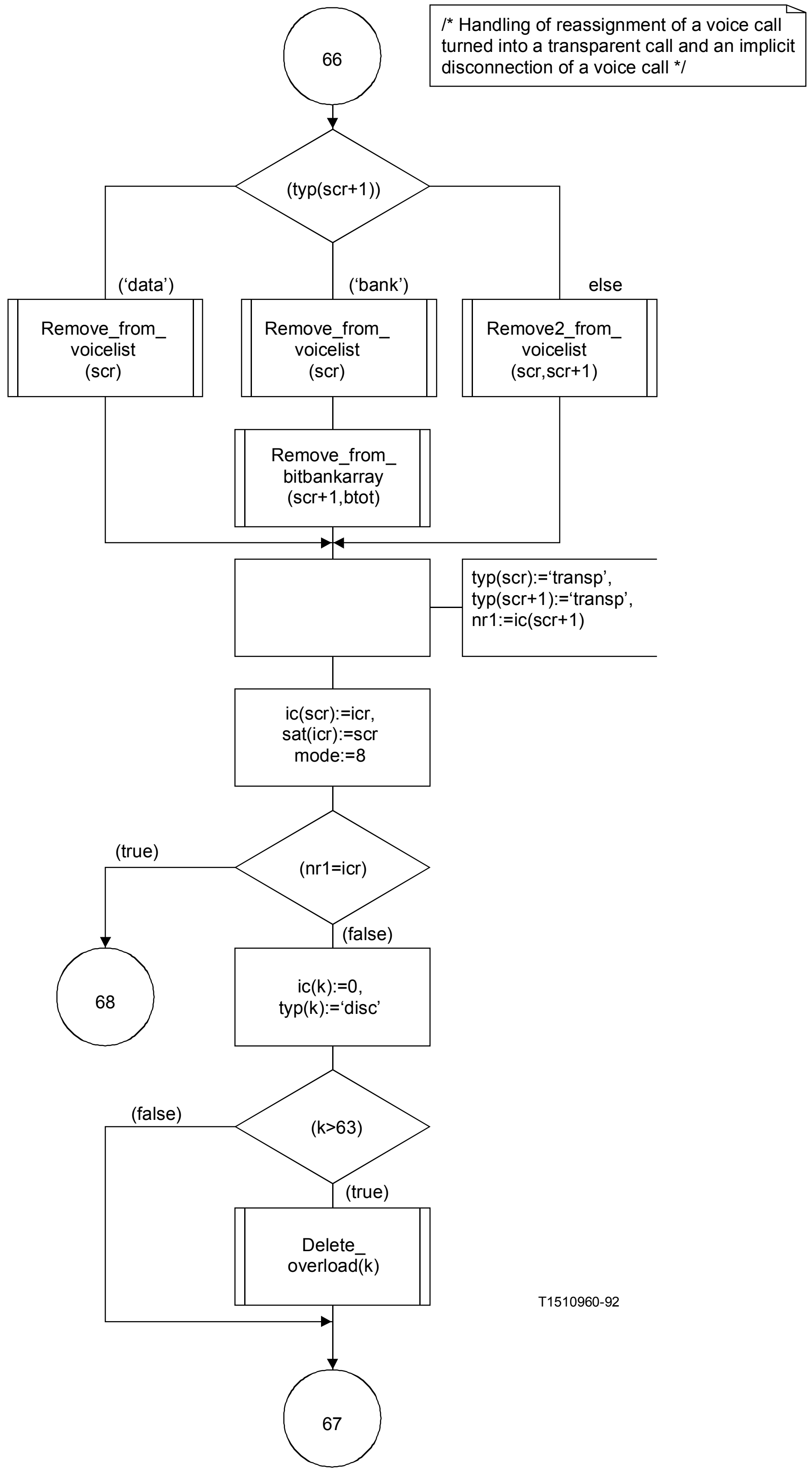

61
/* Handling of a reassignment of a voice call and an implicit disconnection of a voice call */
(nr=icr and k=scr)
(false)
64
/* Actual change */
(true)
(flag)
('transp')
else
('data')
62
/* Refresh or flag change */
typ(scr):= 'data', mode:=5, prep:=0
5
/* Refresh */
Remove_from_ voicelist (scr)
(ad(icr))
else
(0)
Seize (ad(icr),mode) TO DECPID dec(ad(icr)))
Rxdata (ad(icr))
T1510910-92
5

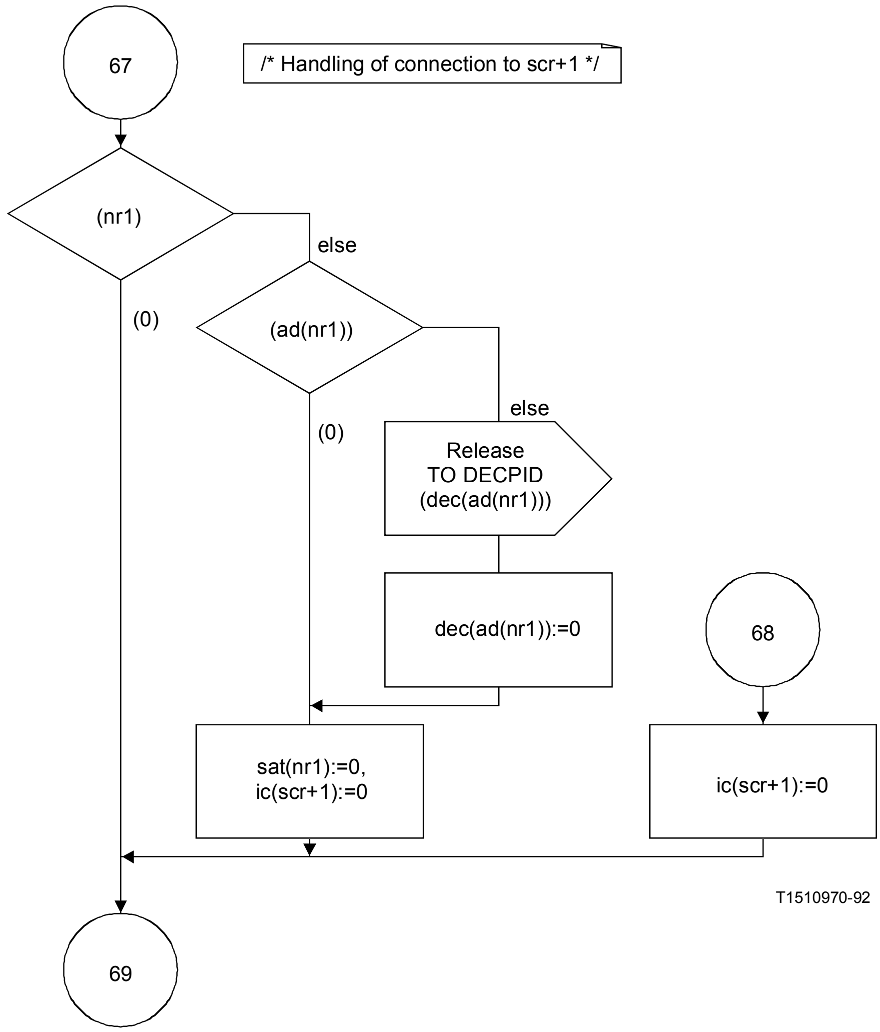

62
/* flag change from voice to transparent */
(typ(scr+1))
('data')
('bank')
else
Remove_from_ voicelist (scr)
Remove_from_ voicelist (scr)
Remove2_from_ voicelist (scr,scr+1)
Remove_from_ bitbankarray (scr+1,btot)
typ(scr):='transp', typ(scr+1):='transp', nr1:=ic(scr+1)
mode:=8
(nr1)
(0)
/* Handling of connection to scr+1 */
(else)
(ad(nr1))
(else)
(0)
Release TO DECPID (dec(ad(nr1)))
dec(ad(nr1)):=0
sat(nr1):=0, ic(scr+1):=0
T1510920-92
63

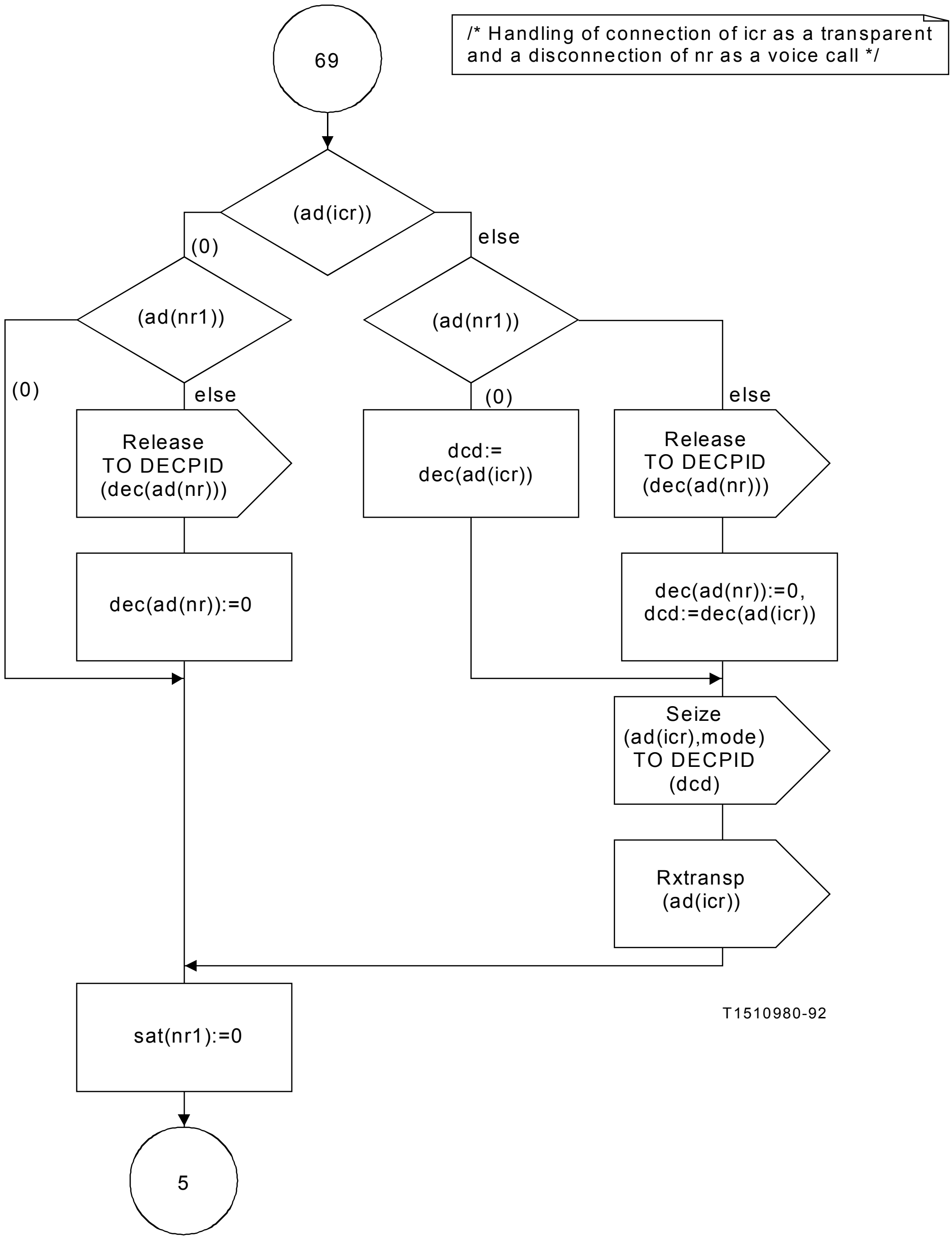

63
/* Handling of connection of icr as transparent instead of voice */
(ad(icr))
(else)
Seize (ad(icr),mode) TO DECPID (dec(ad(icr)))
(0)
Rxtransp (ad(icr))
5
T1510930-92


PROCESS RUD
43 (80)
64
/* Handling of reassignment of a voice call and an implicit disconnection of a voice call */
('transp')
(flag)
else
('data')
66
typ(scr):='data', mode:=5, prep:=0
typ(scr):= 'voice'
ic(scr):=icr, sat(icr):=scr
ic(scr):=icr, sat(icr):=scr
ic(k):=0, typ(k):='disc'
ic(k):=0, typ(k):='disc'
65
(k>63)
(false)
(true)
Delete_ overload(k)
71
/* Handling of connection of icr as voice and an implicit disconnection of nr as voice */
(ad(icr))
(0)
else
(ad(nr))
(0)
(ad(nr))
else
(0)
else
Release TO DECPID (dec(ad(nr)))
dcd:= dec(ad(icr))
Release TO DECPID (dec(ad(nr)))
dec(ad(nr)):=0
dec(ad(nr)):=0, dcd:= dec(ad(icr))
Seizev (ad(icr)) TO DECPID (dcd)
sat(nr):=0
T1510940-92
5

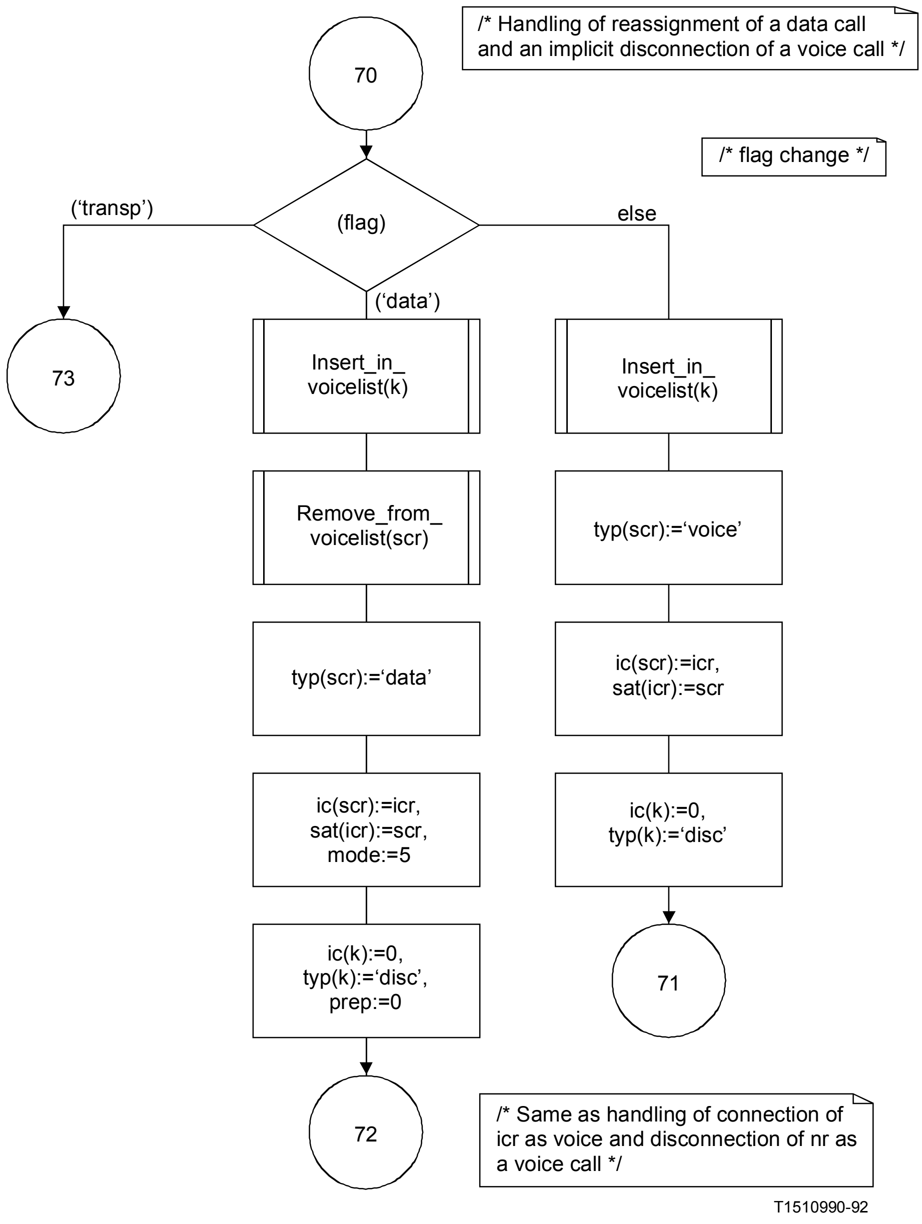

65
/* Handling of reassignment of a voice call to data and an implicit disconnection of a voice call */
Remove_from_ voicelist (scr)
95
(k>63)
(false)
(true)
Delete_ overload(k)
/* Handling of connection of icr as a data call and an implicit disconnection of nr as voice */
(ad(icr))
(0)
else
(ad(nr))
(ad(nr))
(0)
else
(0)
else
Release TO DECPID (dec(ad(nr)))
dcd:= dec(ad(icr))
Release TO DECPID (dec(ad(nr)))
dec(ad(nr)):=0
dec(ad(nr)):=0, dcd:= dec(ad(icr))
Seize (ad(icr),mode) TO DECPID (dcd)
Rxdata (ad(icr))
sat(nr):=0
5
T1510950-92

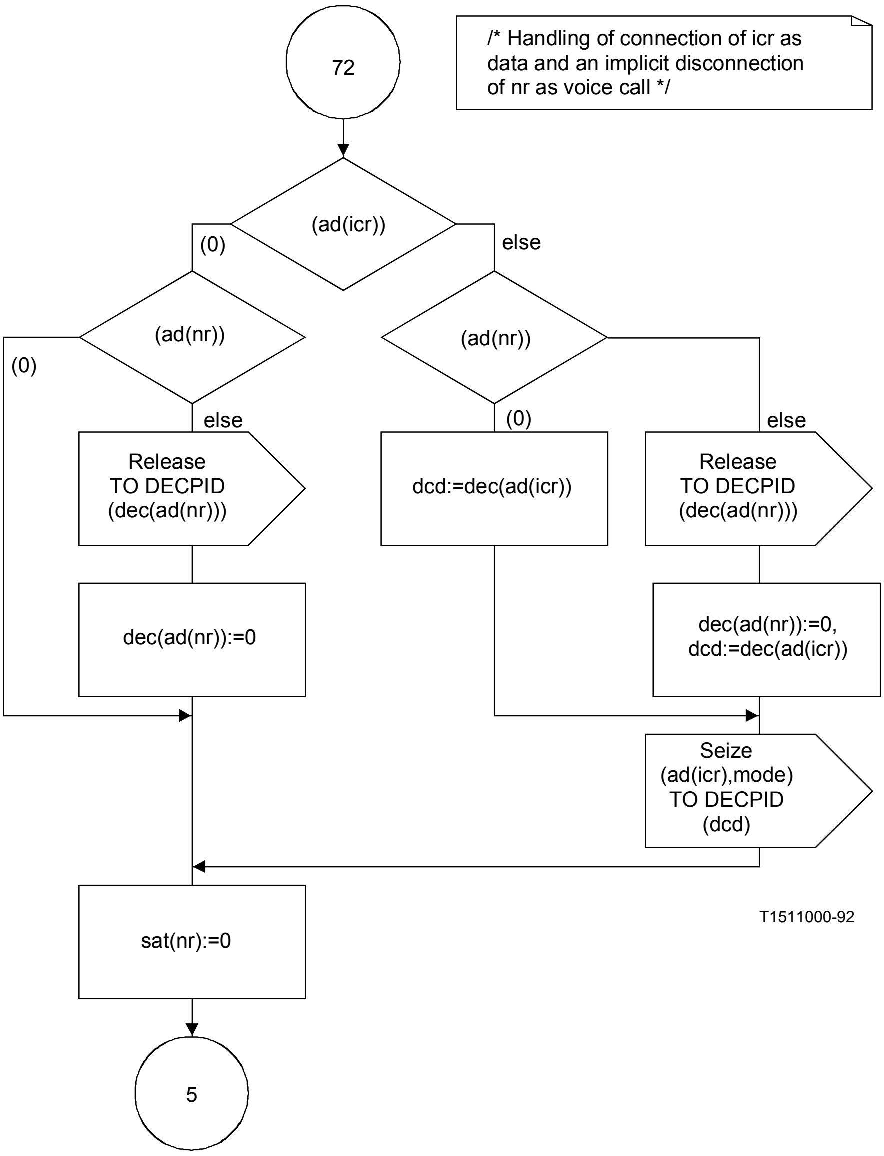

/* Handling of reassignment of a voice call turned into a transparent call and an implicit disconnection of a voice call */
66
(typ(scr+1))
('data')
('bank')
else
Remove_from_ voicelist (scr)
Remove_from_ voicelist (scr)
Remove2_from_ voicelist (scr,scr+1)
Remove_from_ bitbankarray (scr+1,btot)
typ(scr):='transp', typ(scr+1):='transp', nr1:=ic(scr+1)
ic(scr):=icr, sat(icr):=scr mode:=8
(true)
(nr1=icr)
(false)
68
ic(k):=0, typ(k):='disc'
(false)
(k>63)
(true)
Delete_ overload(k)
T1510960-92
67

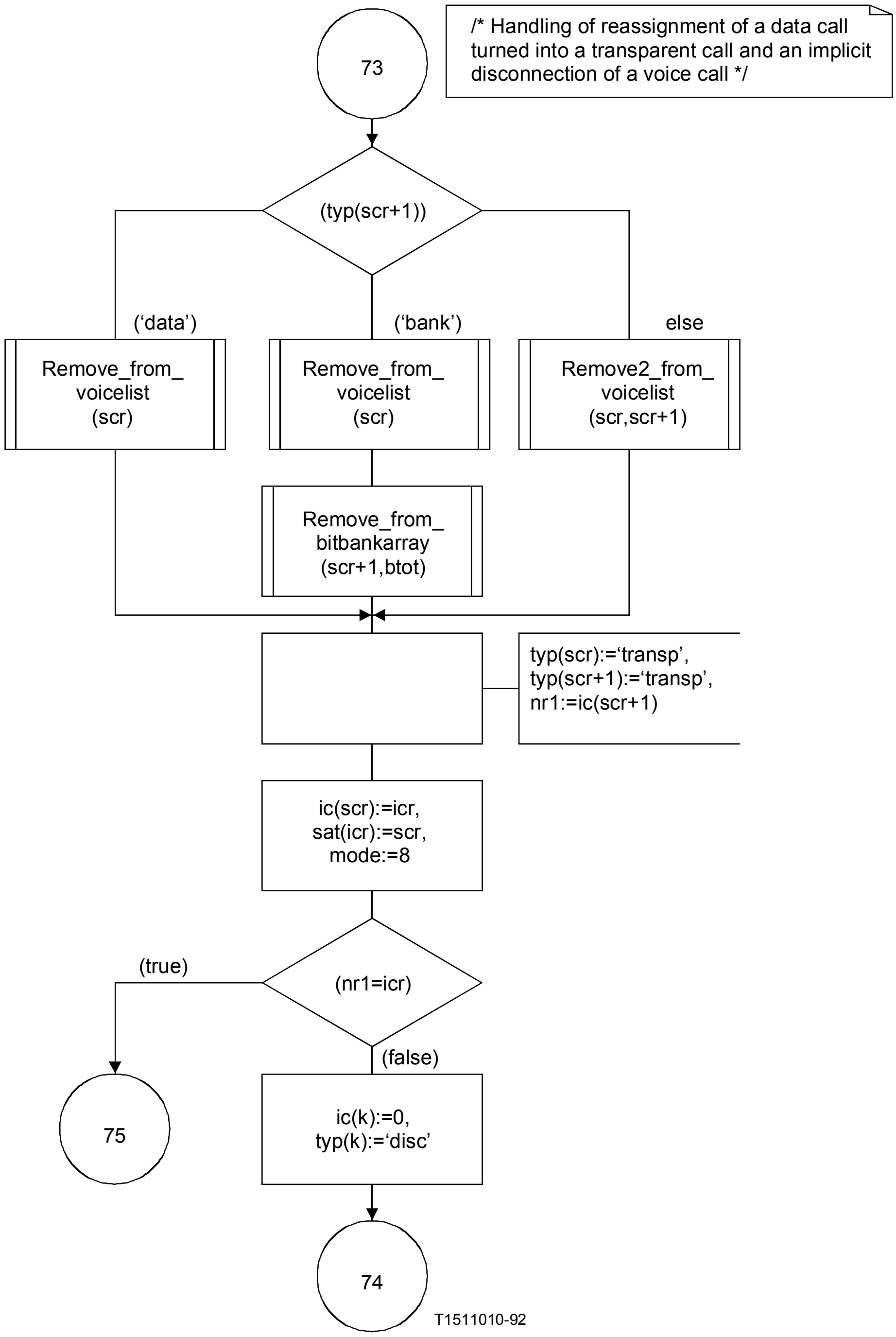

67
/* Handling of connection to scr+1 */
(nr1)
else
(0)
(ad(nr1))
else
(0)
Release
TO DECPID
(dec(ad(nr1)))
dec(ad(nr1)):=0
68
sat(nr1):=0,
ic(scr+1):=0
ic(scr+1):=0
T1510970-92
69



69
/* Handling of connection of icr as a transparent and a disconnection of nr as a voice call */
(ad(icr))
(0)
else
(ad(nr1))
(ad(nr1))
(0)
else
(0)
else
Release TO DECPID (dec(ad(nr)))
dcd:= dec(ad(icr))
Release TO DECPID (dec(ad(nr)))
dec(ad(nr)):=0
dec(ad(nr)):=0, dcd:=dec(ad(icr))
Seize (ad(icr),mode) TO DECPID (dcd)
Rxtransp (ad(icr))
sat(nr1):=0
T1510980-92
5

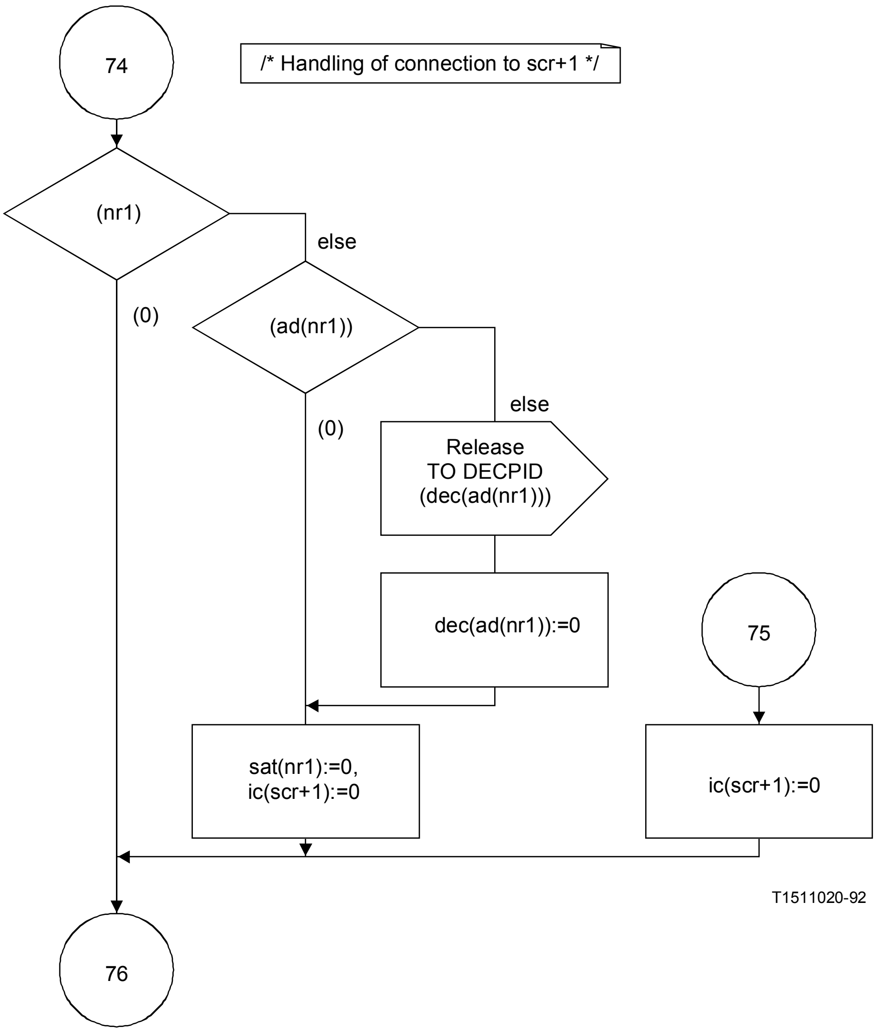

/* Handling of reassignment of a data call and an implicit disconnection of a voice call */
70
/* flag change */
('transp')
(flag)
else
('data')
73
Insert_in_ voicelist(k)
Insert_in_ voicelist(k)
Remove_from_ voicelist(scr)
typ(scr):='voice'
typ(scr):='data'
ic(scr):=icr, sat(icr):=scr
ic(scr):=icr, sat(icr):=scr, mode:=5
ic(k):=0, typ(k):='disc'
ic(k):=0, typ(k):='disc', prep:=0
71
72
/* Same as handling of connection of icr as voice and disconnection of nr as a voice call */
T1510990-92


72
/* Handling of connection of icr as data and an implicit disconnection of nr as voice call */
(ad(icr))
(0)
else
(ad(nr))
(0)
(ad(nr))
else
(0)
else
Release
TO DECPID
(dec(ad(nr)))
dcd:=dec(ad(icr))
Release
TO DECPID
(dec(ad(nr)))
dec(ad(nr)):=0
dec(ad(nr)):=0,
dcd:=dec(ad(icr))
Seize
(ad(icr),mode)
TO DECPID
(dcd)
sat(nr):=0
5
T1511000-92

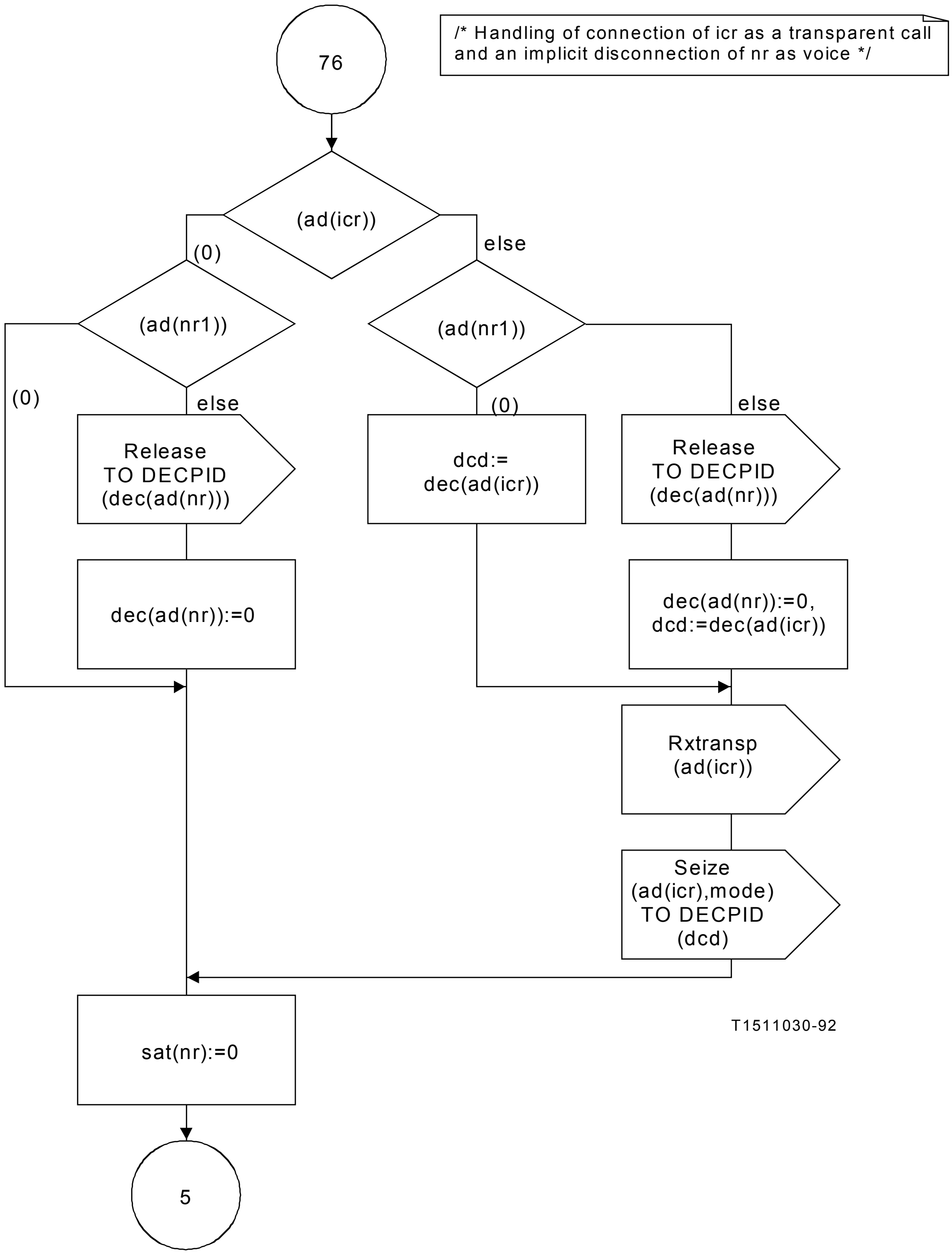


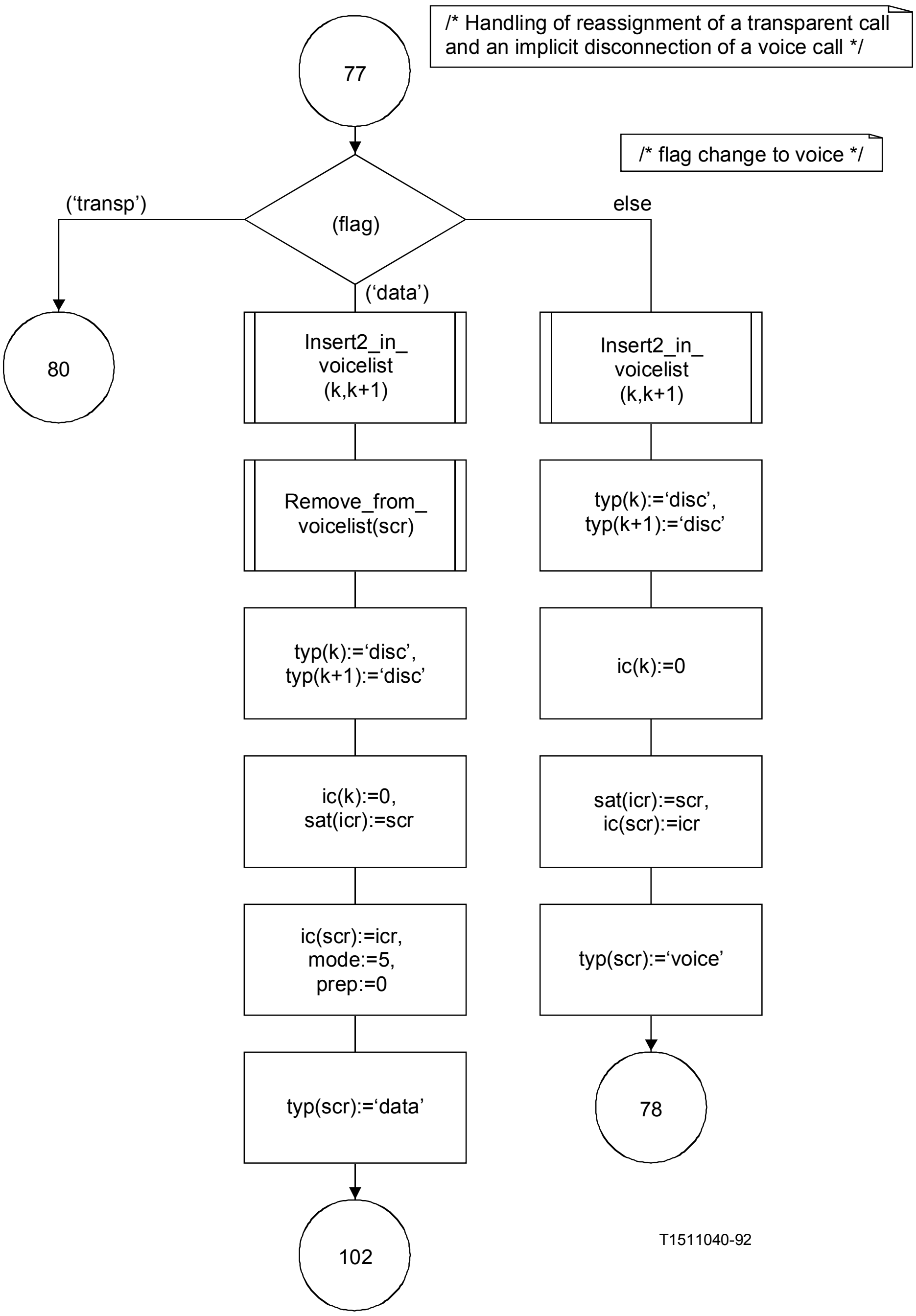
73
/* Handling of reassignment of a data call turned into a transparent call and an implicit disconnection of a voice call */
(typ(scr+1))
('data')
('bank')
else
Remove_from_ voicelist (scr)
Remove_from_ voicelist (scr)
Remove2_from_ voicelist (scr,scr+1)
Remove_from_ bitbankarray (scr+1,btot)
typ(scr):='transp', typ(scr+1):='transp', nr1:=ic(scr+1)
ic(scr):=icr, sat(icr):=scr, mode:=8
(true)
(nr1=icr)
(false)
75
ic(k):=0, typ(k):='disc'
74

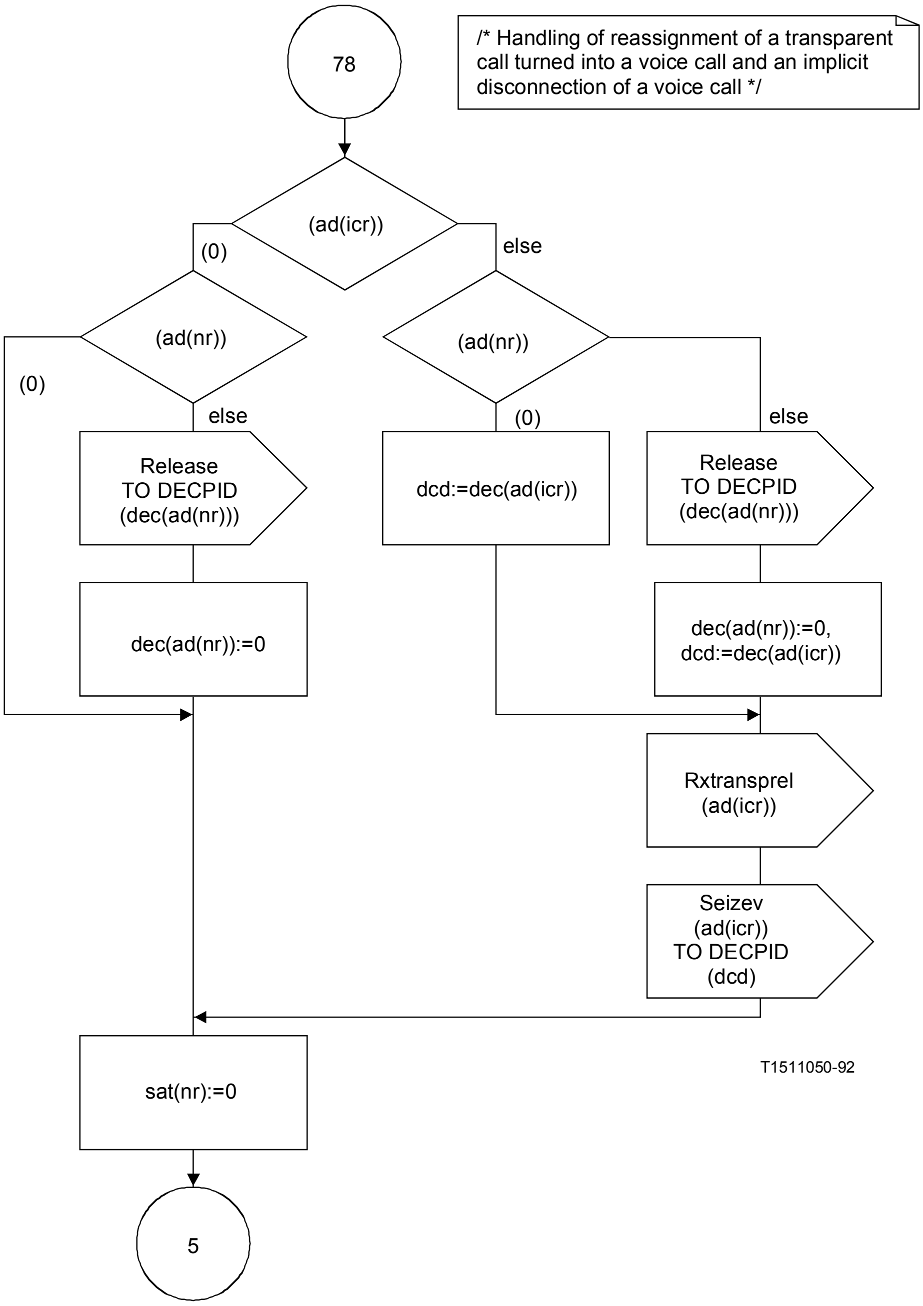
T1511010-92

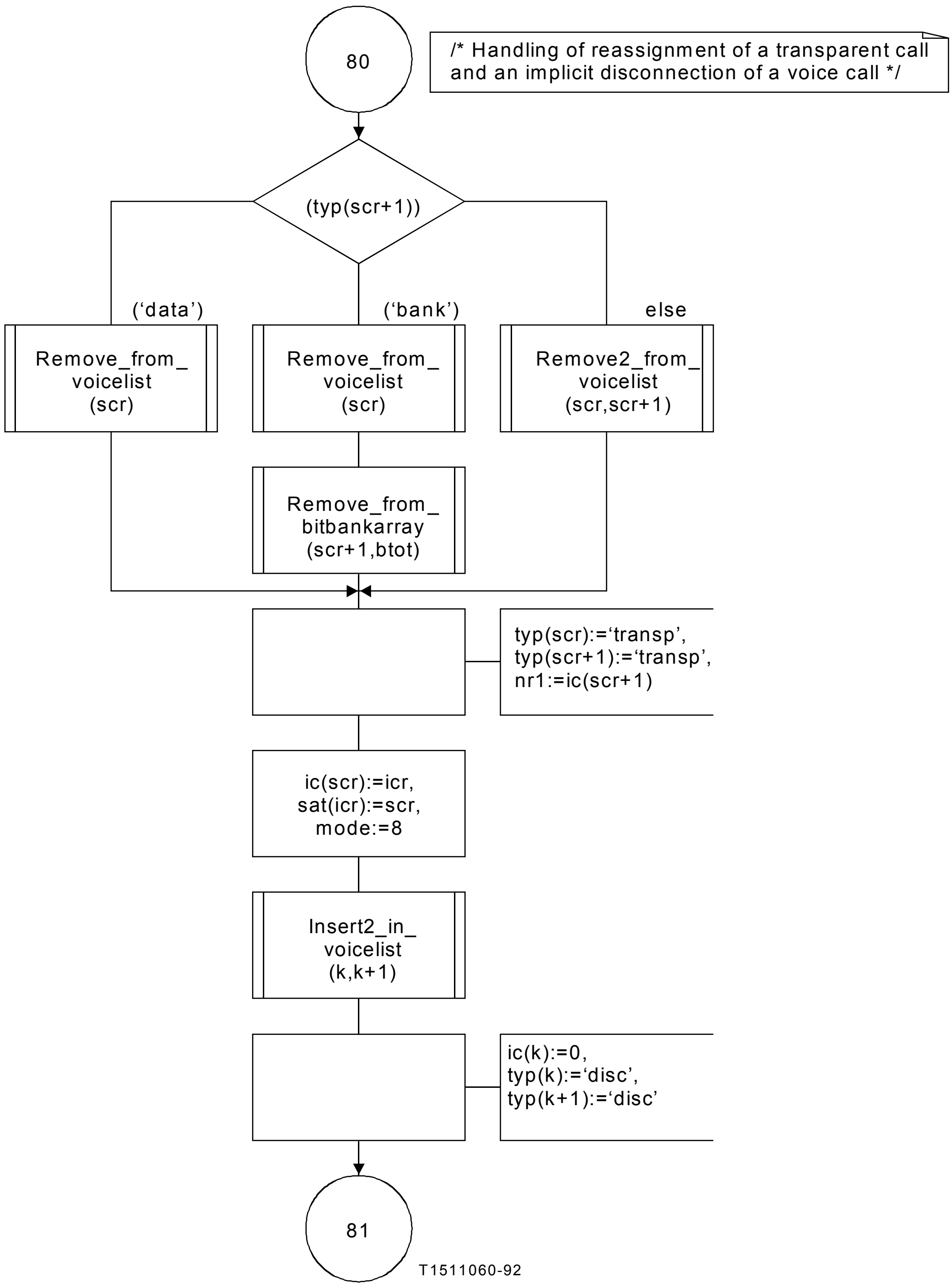

74
/* Handling of connection to scr+1 */
(nr1)
else
(0)
(ad(nr1))
else
(0)
Release
TO DECPID
(dec(ad(nr1)))
dec(ad(nr1)):=0
75
sat(nr1):=0,
ic(scr+1):=0
ic(scr+1):=0
76

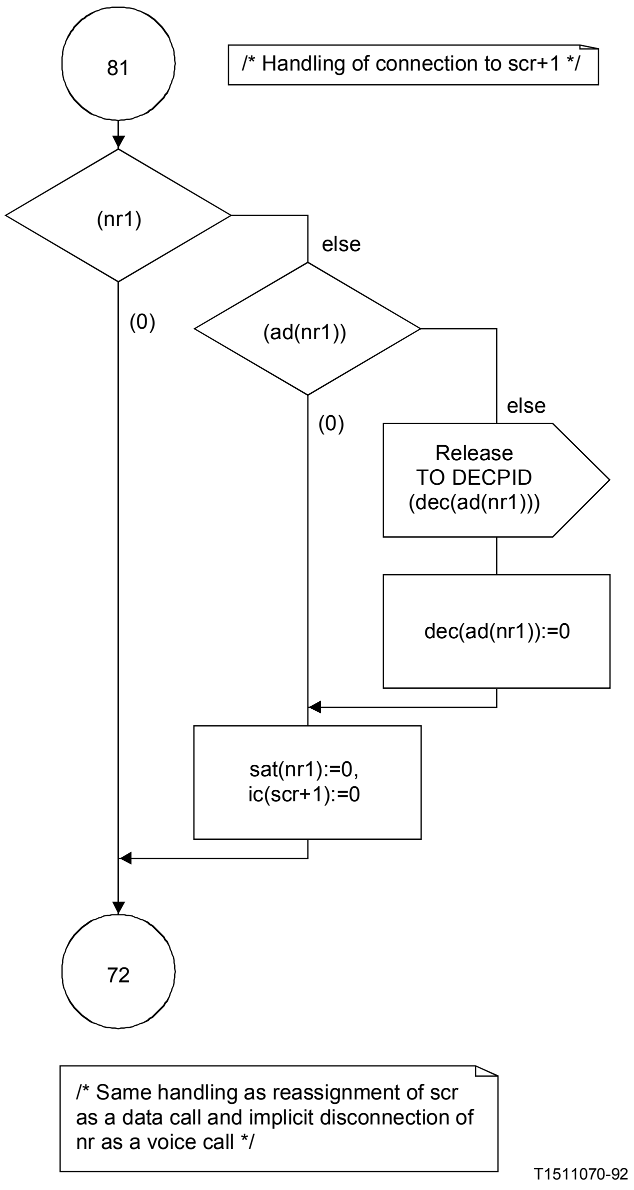
T1511020-92

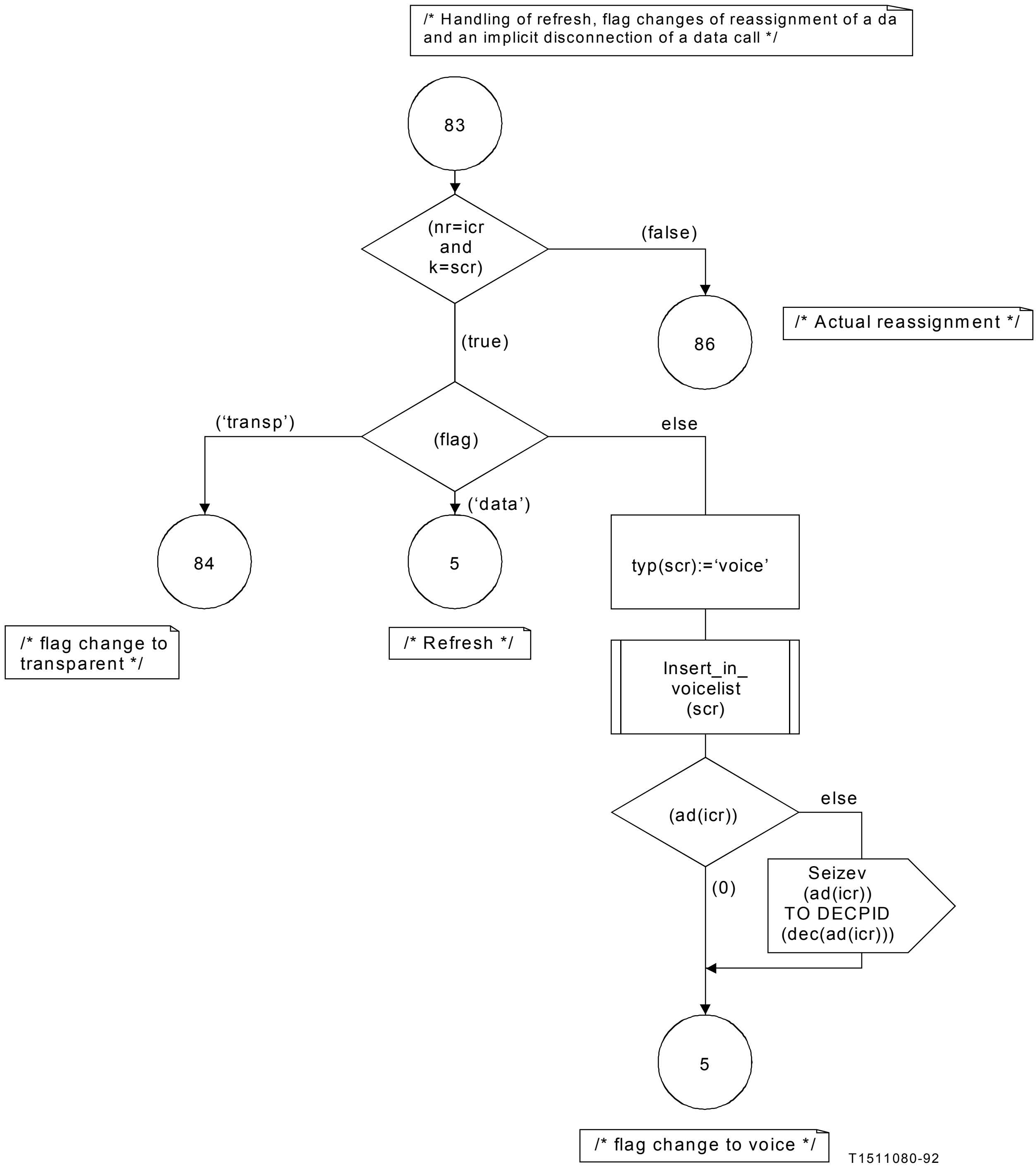

/* Handling of connection of icr as a transparent call and an implicit disconnection of nr as voice */
76
(ad(icr))
(0)
else
(ad(nr1))
(ad(nr1))
(0)
else
(0)
else
Release TO DECPID (dec(ad(nr)))
dcd:= dec(ad(icr))
Release TO DECPID (dec(ad(nr)))
dec(ad(nr)):=0
dec(ad(nr)):=0, dcd:=dec(ad(icr))
Rxtransp (ad(icr))
Seize (ad(icr),mode) TO DECPID (dcd)
sat(nr):=0
5
T1511030-92

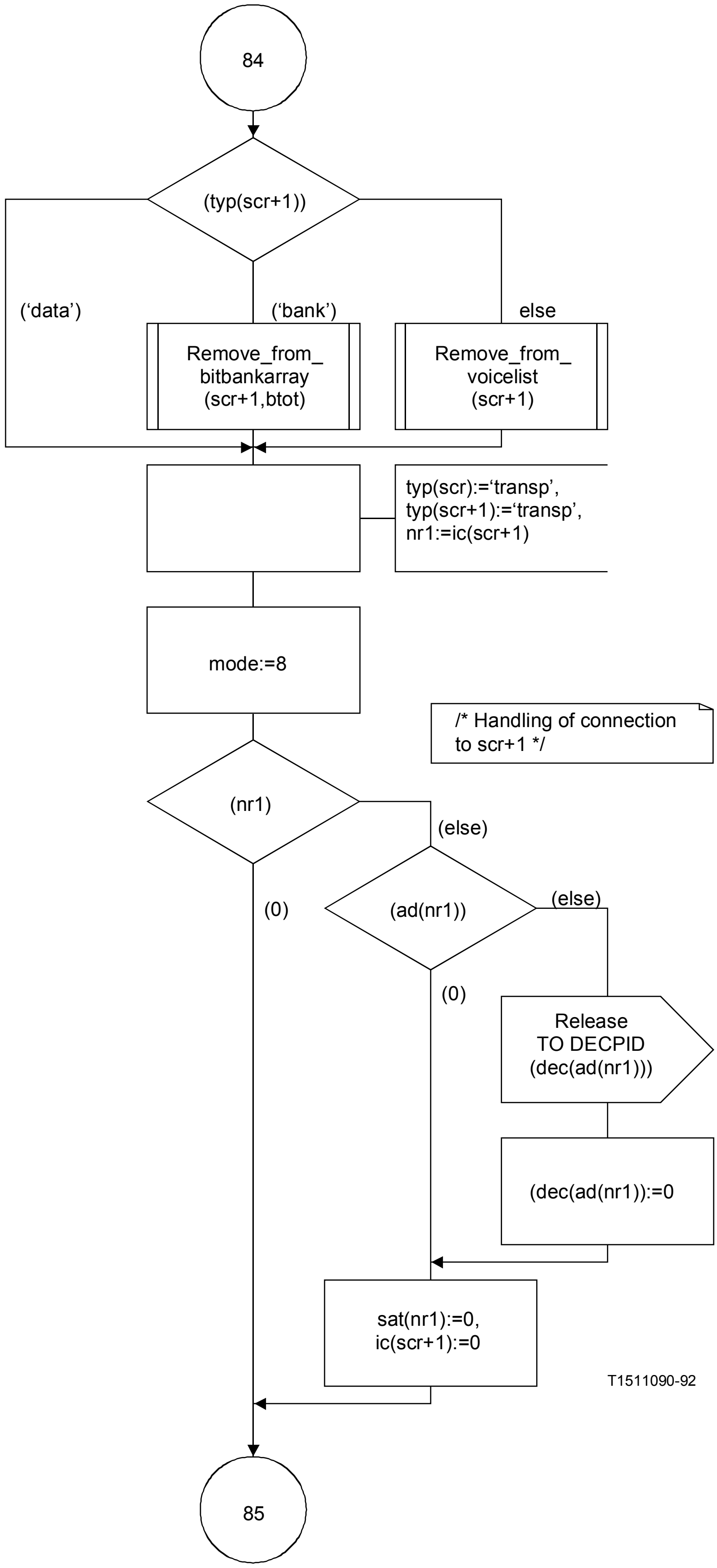

/* Handling of reassignment of a transparent call and an implicit disconnection of a voice call */
77
/* flag change to voice */
(flag)
('transp')
else
('data')
80
Insert2_in_ voicelist (k,k+1)
Remove_from_ voicelist(scr)
typ(k):='disc', typ(k+1):='disc'
ic(k):=0, sat(icr):=scr
ic(scr):=icr, mode:=5, prep:=0
typ(scr):='data'
102
Insert2_in_ voicelist (k,k+1)
typ(k):='disc', typ(k+1):='disc'
ic(k):=0
sat(icr):=scr, ic(scr):=icr
typ(scr):='voice'
78
T1511040-92

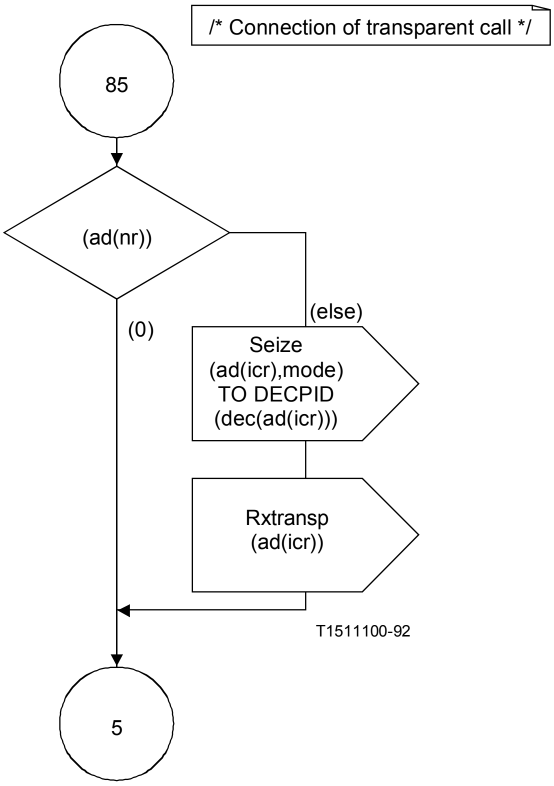

78
/* Handling of reassignment of a transparent call turned into a voice call and an implicit disconnection of a voice call */
(ad(icr))
(0)
else
(ad(nr))
(0)
else
(ad(nr))
(0)
else
Release TO DECPID (dec(ad(nr)))
dcd:=dec(ad(icr))
Release TO DECPID (dec(ad(nr)))
dec(ad(nr)):=0
dec(ad(nr)):=0, dcd:=dec(ad(icr))
Rxtransprel (ad(icr))
Seizev (ad(icr)) TO DECPID (dcd)
sat(nr):=0
5
T1511050-92

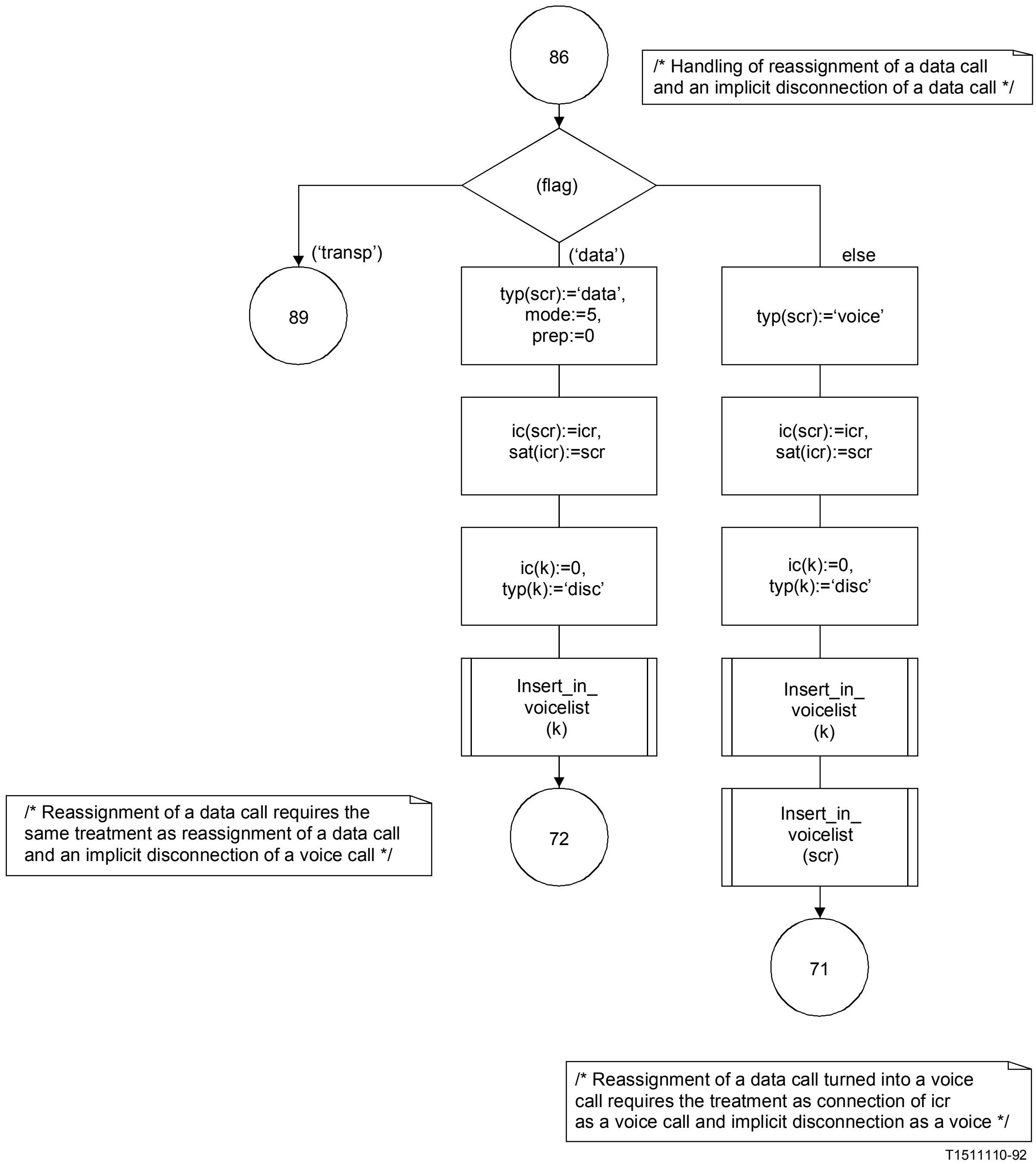

80
/* Handling of reassignment of a transparent call and an implicit disconnection of a voice call */
(typ(scr+1))
('data')
('bank')
else
Remove_from_ voicelist (scr)
Remove_from_ voicelist (scr)
Remove2_from_ voicelist (scr,scr+1)
Remove_from_ bitbankarray (scr+1,btot)
typ(scr):='transp',
typ(scr+1):='transp',
nr1:=ic(scr+1)
ic(scr):=icr,
sat(icr):=scr,
mode:=8
Insert2_in_ voicelist (k,k+1)
ic(k):=0,
typ(k):='disc',
typ(k+1):='disc'
81
T1511060-92


81
/* Handling of connection to scr+1 */
(nr1)
else
(0)
(ad(nr1))
else
(0)
Release
TO DECPID
(dec(ad(nr1)))
dec(ad(nr1)):=0
sat(nr1):=0,
ic(scr+1):=0
72
/* Same handling as reassignment of scr
as a data call and implicit disconnection of
nr as a voice call */
T1511070-92

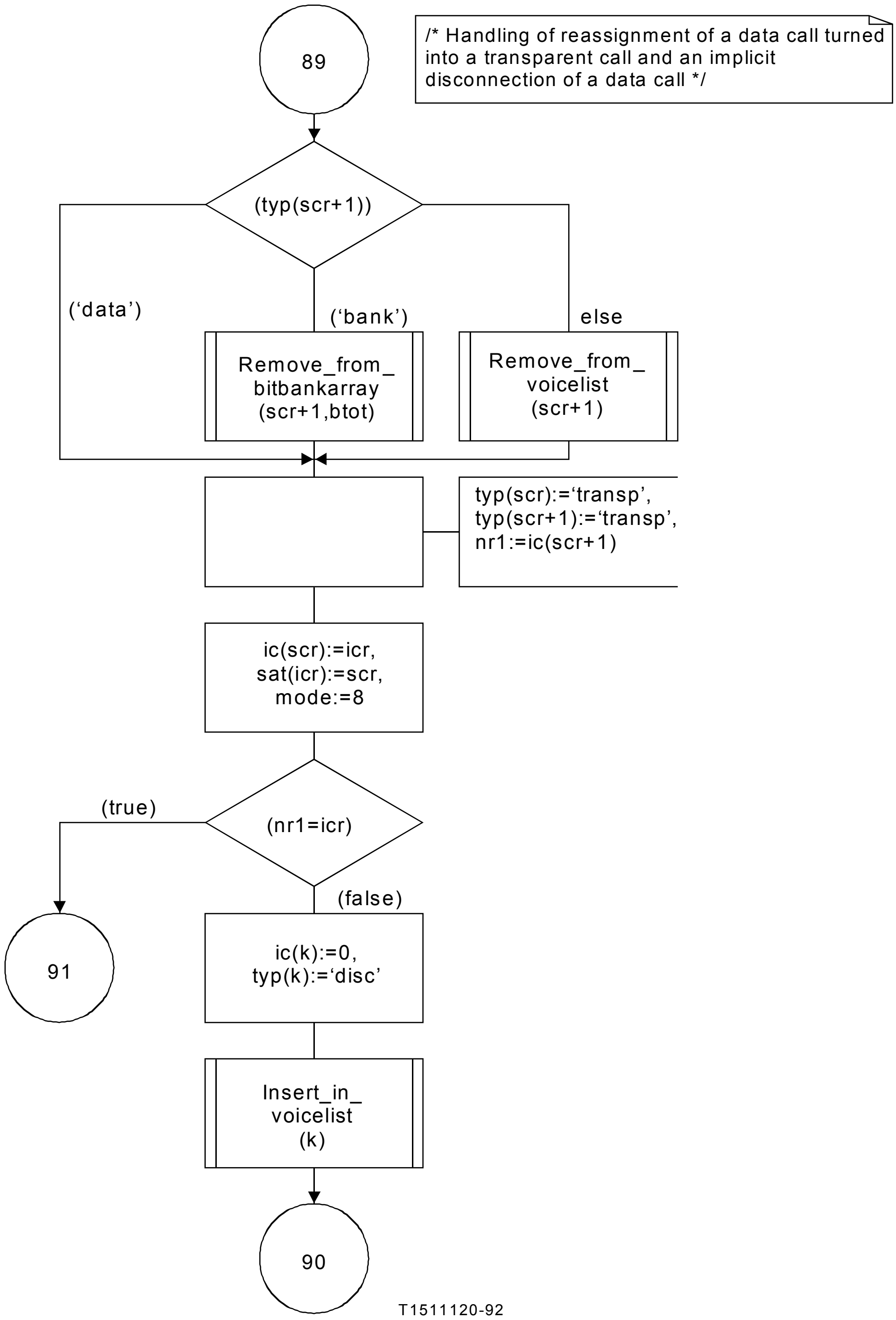

/* Handling of refresh, flag changes of reassignment of a da
and an implicit disconnection of a data call */
83
(nr=icr
and
k=scr)
(false)
86
/* Actual reassignment */
(true)
('transp')
(flag)
else
('data')
84
5
typ(scr):='voice'
/* flag change to
transparent */
/* Refresh */
Insert_in_
voicelist
(scr)
(ad(icr))
else
(0)
Seizev
(ad(icr))
TO DECPID
(dec(ad(icr)))
5
/* flag change to voice */
T1511080-92


84
(typ(scr+1))
('data')
('bank')
else
Remove_from_ bitbankarray (scr+1,btot)
Remove_from_ voicelist (scr+1)
typ(scr):='transp', typ(scr+1):='transp', nr1:=ic(scr+1)
mode:=8
/* Handling of connection to scr+1 */
(nr1)
(else)
(0)
(ad(nr1))
(else)
(0)
Release TO DECPID (dec(ad(nr1)))
(dec(ad(nr1)):=0
sat(nr1):=0, ic(scr+1):=0
T1511090-92
85

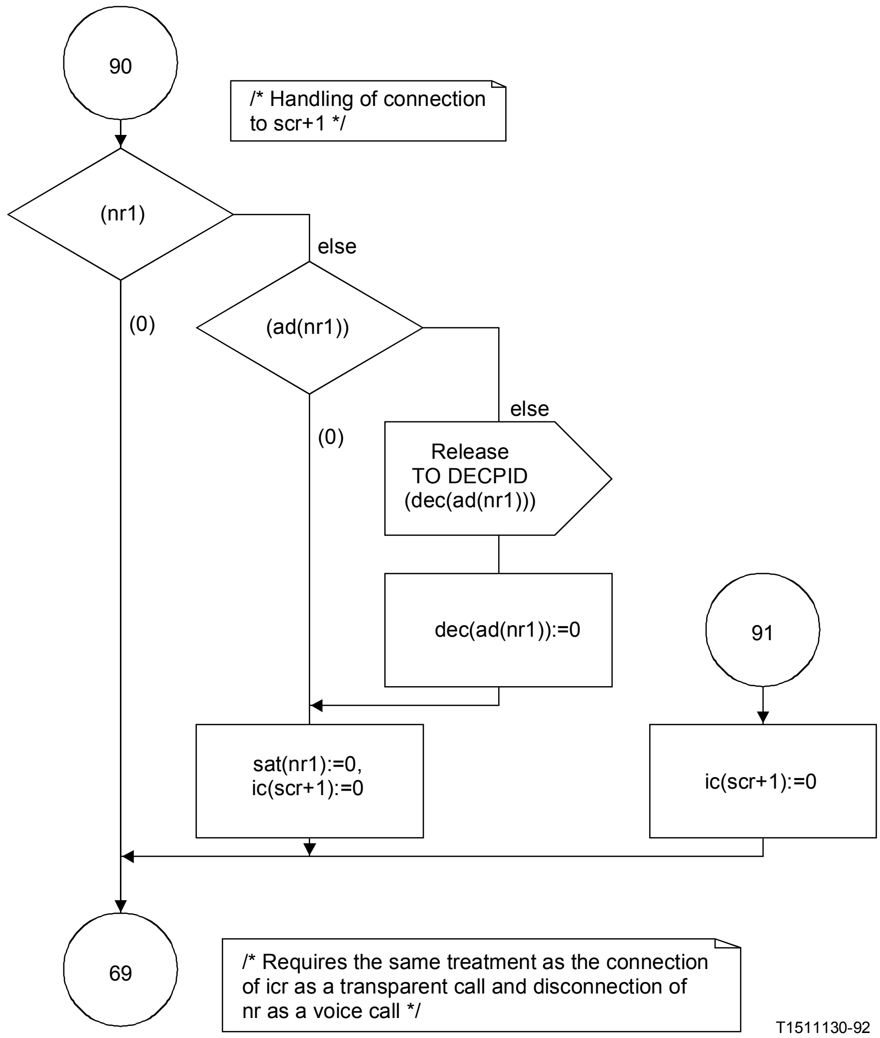

/* Connection of transparent call */
85
(ad(nr))
(0)
(else)
Seize
(ad(icr),mode)
TO DECPID
(dec(ad(icr)))
Rxtransp
(ad(icr))
5
T1511100-92

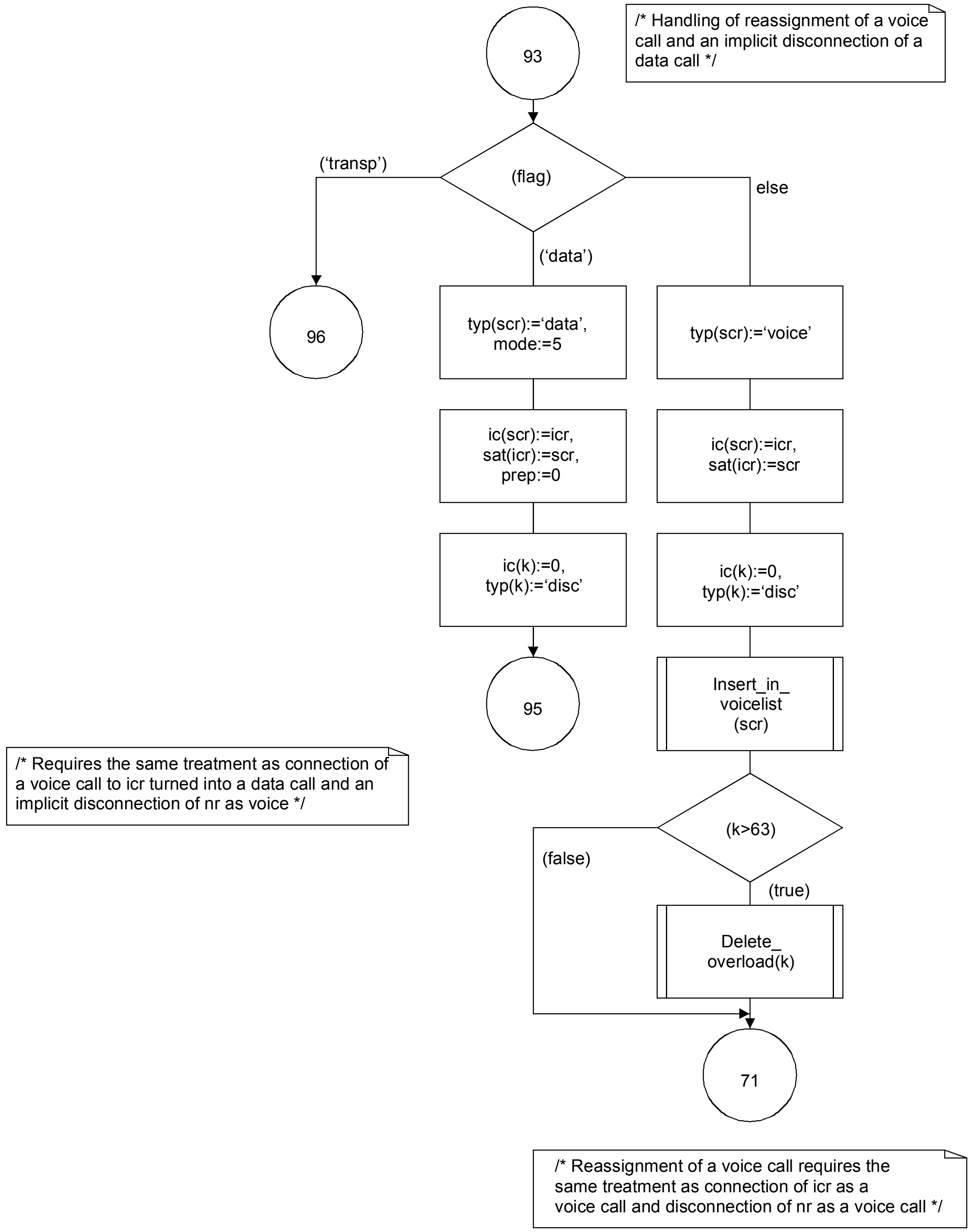


86
/* Handling of reassignment of a data call and an implicit disconnection of a data call */
(flag)
('transp')
('data')
else
89
typ(scr):='data', mode:=5, prep:=0
typ(scr):='voice'
ic(scr):=icr, sat(icr):=scr
ic(scr):=icr, sat(icr):=scr
ic(k):=0, typ(k):='disc'
ic(k):=0, typ(k):='disc'
Insert_in_ voicelist (k)
Insert_in_ voicelist (k)
/* Reassignment of a data call requires the same treatment as reassignment of a data call and an implicit disconnection of a voice call */
72
Insert_in_ voicelist (scr)
71
/* Reassignment of a data call turned into a voice call requires the treatment as connection of icr as a voice call and implicit disconnection as a voice */
T1511110-92


89
/* Handling of reassignment of a data call turned into a transparent call and an implicit disconnection of a data call */
(typ(scr+1))
('data')
('bank')
else
Remove_from_ bitbankarray (scr+1,btot)
Remove_from_ voicelist (scr+1)
typ(scr):='transp', typ(scr+1):='transp', nr1:=ic(scr+1)
ic(scr):=icr, sat(icr):=scr, mode:=8
(true)
(nr1=icr)
(false)
91
ic(k):=0, typ(k):='disc'
Insert_in_ voicelist (k)
90

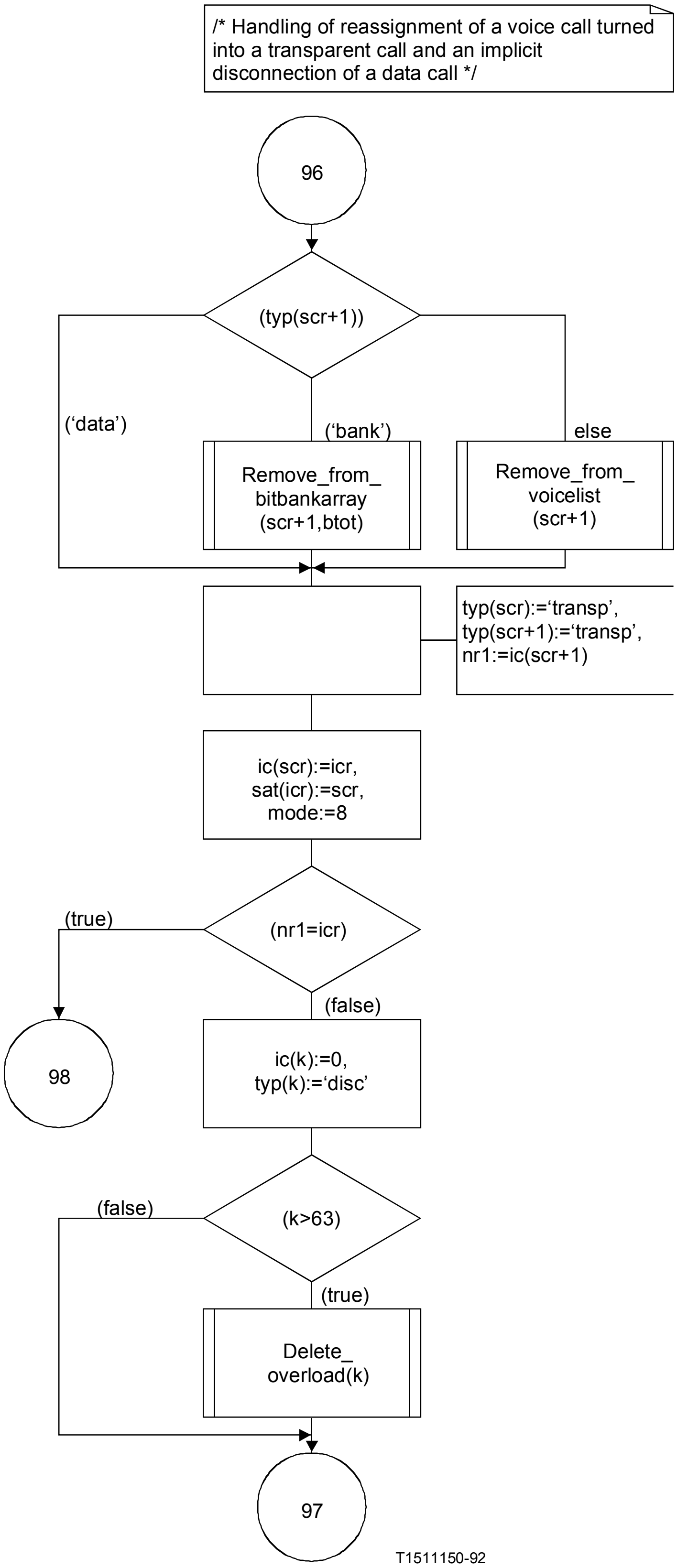
T1511120-92

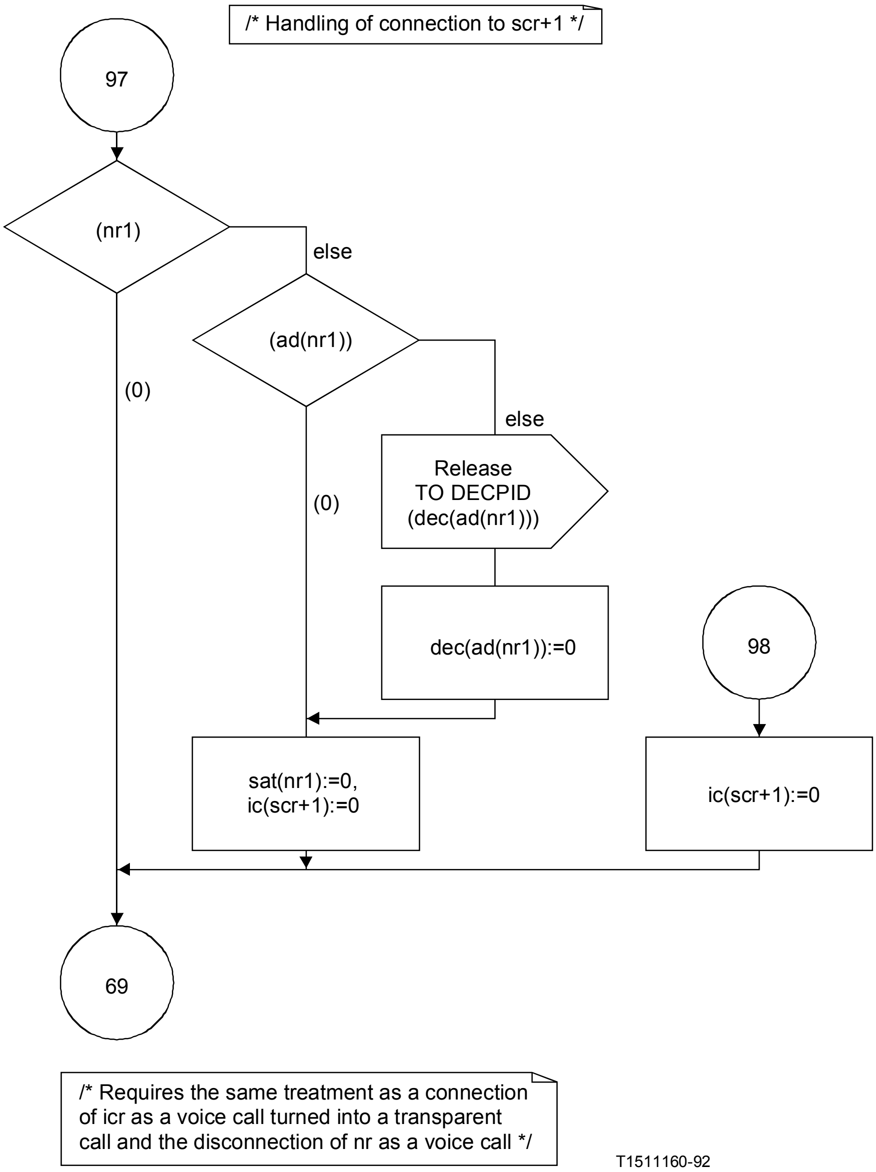

90
/* Handling of connection
to scr+1 */
(nr1)
else
(0)
(ad(nr1))
else
(0)
Release
TO DECPID
(dec(ad(nr1)))
dec(ad(nr1)):=0
91
sat(nr1):=0,
ic(scr+1):=0
ic(scr+1):=0
69
/* Requires the same treatment as the connection
of icr as a transparent call and disconnection of
nr as a voice call */
T1511130-92

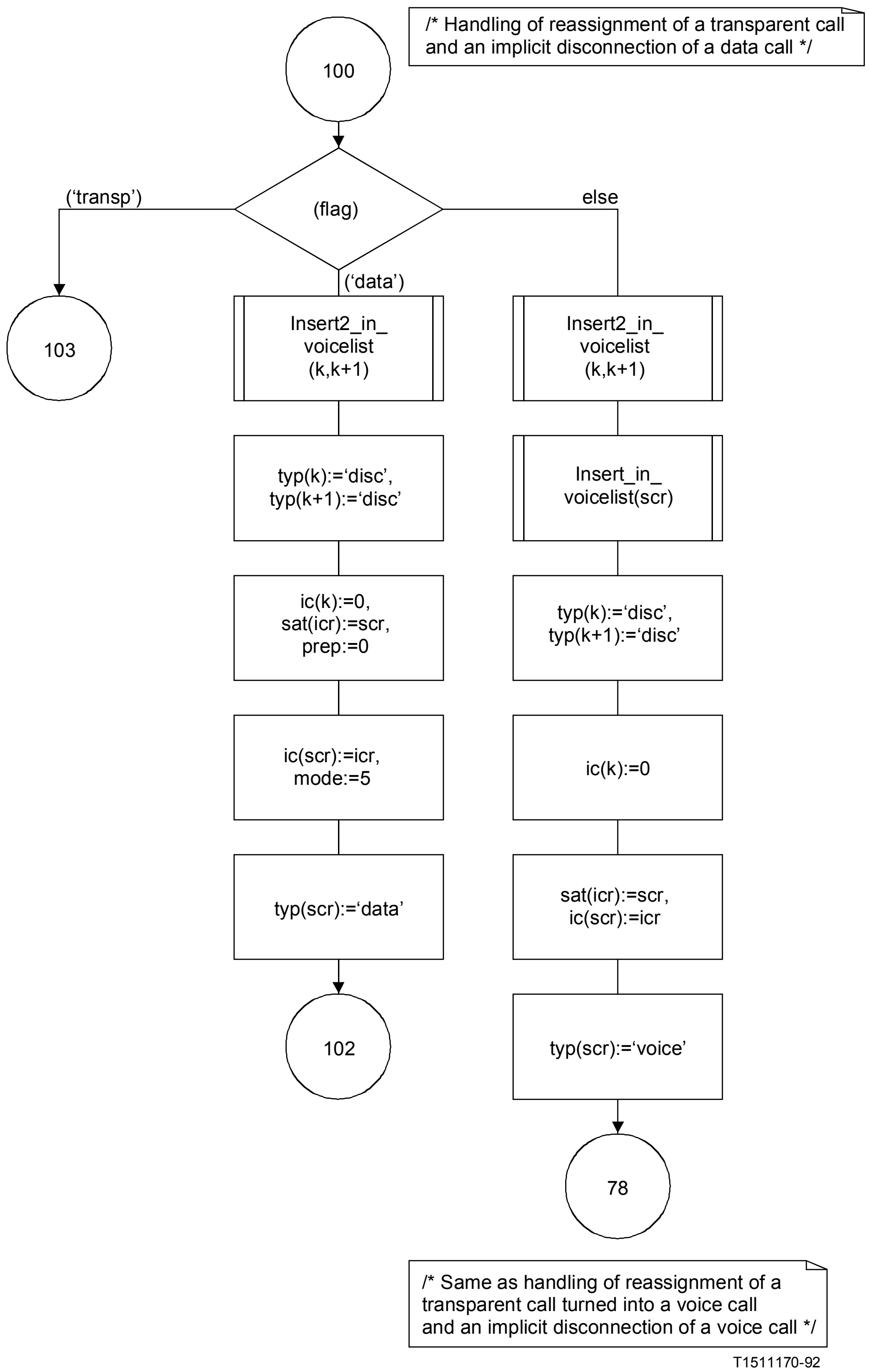


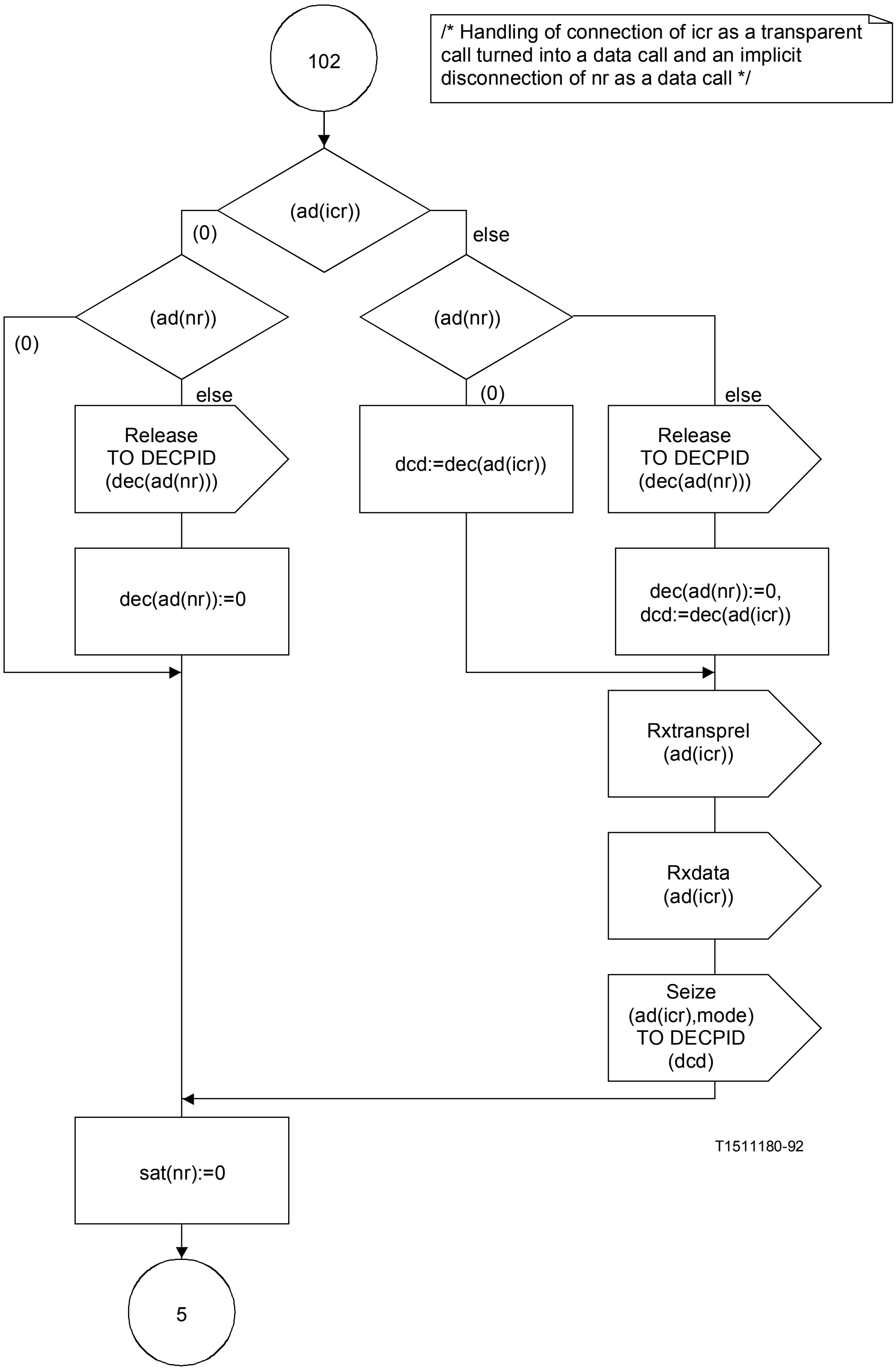
93
/* Handling of reassignment of a voice call and an implicit disconnection of a data call */
(flag)
('transp')
else
('data')
96
typ(scr):='data', mode:=5
typ(scr):='voice'
ic(scr):=icr, sat(icr):=scr, prep:=0
ic(scr):=icr, sat(icr):=scr
ic(k):=0, typ(k):='disc'
ic(k):=0, typ(k):='disc'
95
Insert_in_ voicelist (scr)
/* Requires the same treatment as connection of a voice call to icr turned into a data call and an implicit disconnection of nr as voice */
(k>63)
(false)
(true)
Delete_ overload(k)
71
/* Reassignment of a voice call requires the same treatment as connection of icr as a voice call and disconnection of nr as a voice call */

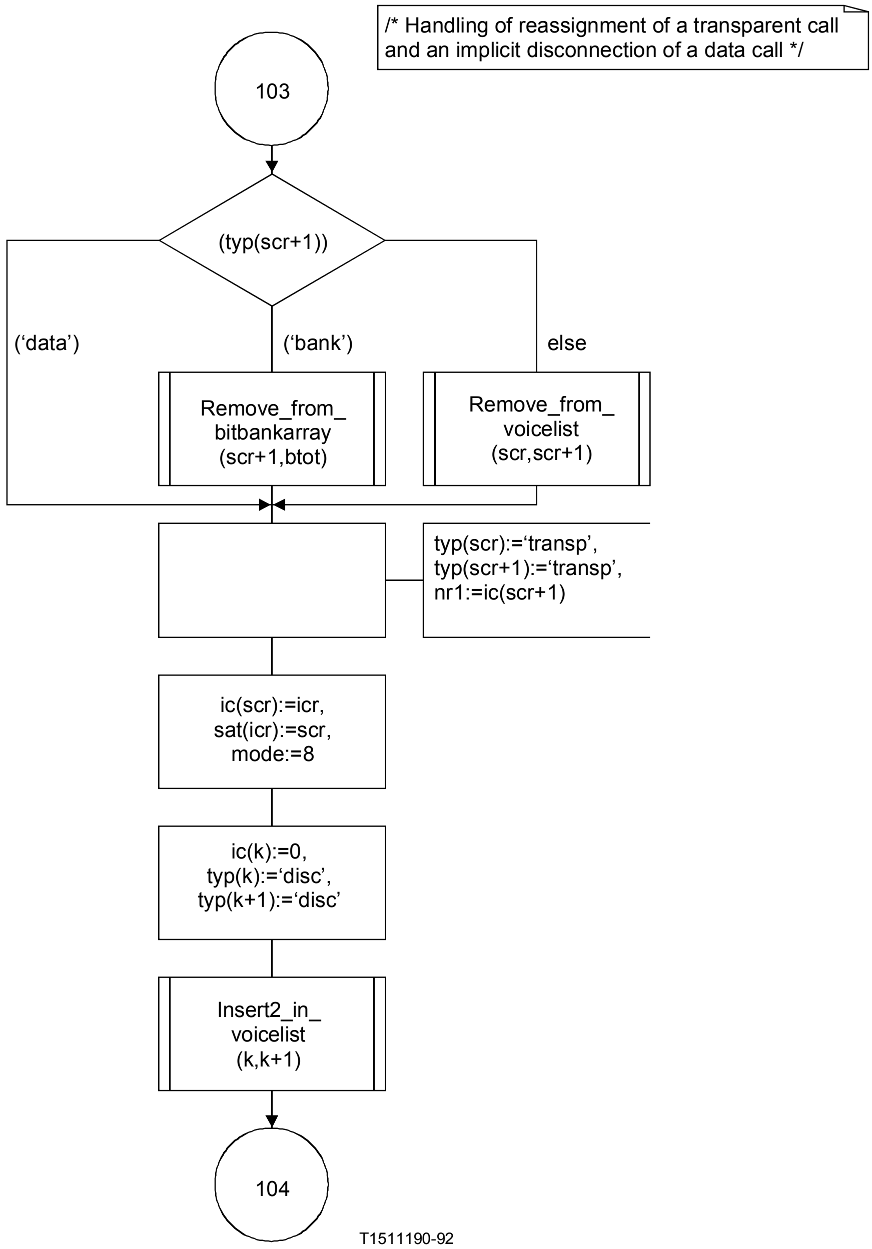
T1511140-92

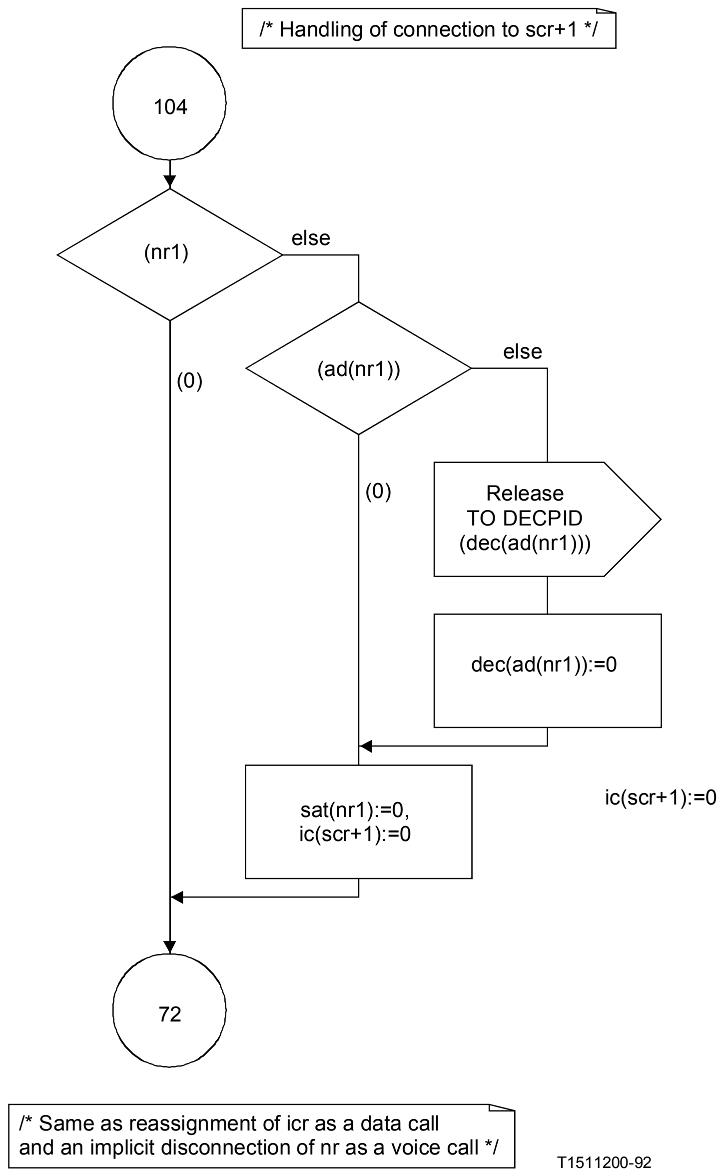

/* Handling of reassignment of a voice call turned into a transparent call and an implicit disconnection of a data call */
96
(typ(scr+1))
('data')
('bank')
else
Remove_from_ bitbankarray (scr+1,btot)
Remove_from_ voicelist (scr+1)
typ(scr):='transp', typ(scr+1):='transp', nr1:=ic(scr+1)
ic(scr):=icr, sat(icr):=scr, mode:=8
(true)
(nr1=icr)
(false)
98
ic(k):=0, typ(k):='disc'
(false)
(k>63)
(true)
Delete_ overload(k)
97
T1511150-92

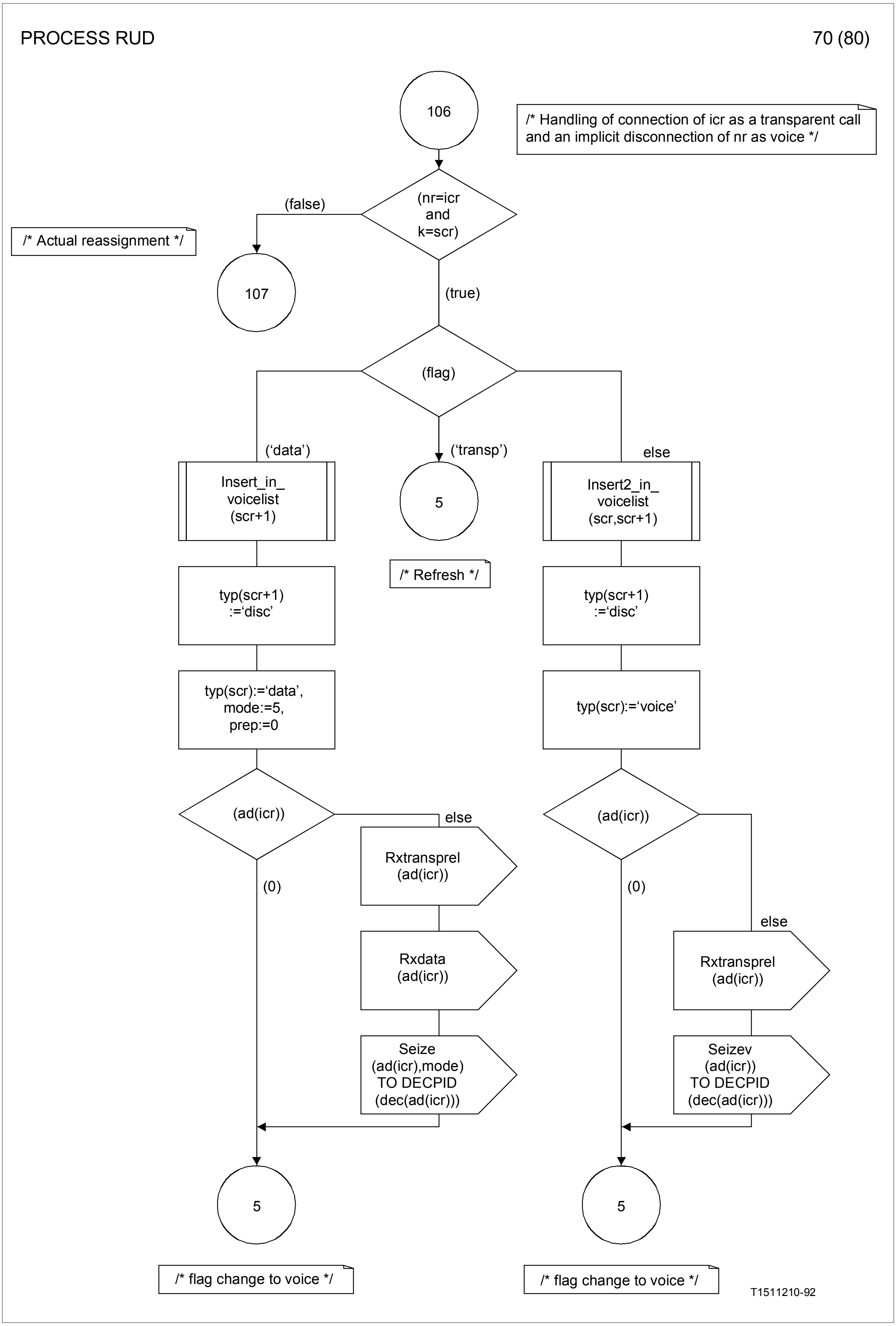

/* Handling of connection to scr+1 */
97
(nr1)
else
(0)
(ad(nr1))
else
(0)
Release
TO DECPID
(dec(ad(nr1)))
dec(ad(nr1)):=0
98
sat(nr1):=0,
ic(scr+1):=0
ic(scr+1):=0
69
/* Requires the same treatment as a connection
of icr as a voice call turned into a transparent
call and the disconnection of nr as a voice call */
T1511160-92


/* Handling of reassignment of a transparent call and an implicit disconnection of a data call */
100
(flag)
('transp')
else
('data')
103
Insert2_in_ voicelist (k,k+1)
typ(k):='disc', typ(k+1):='disc'
ic(k):=0, sat(icr):=scr, prep:=0
ic(scr):=icr, mode:=5
typ(scr):='data'
102
Insert2_in_ voicelist (k,k+1)
Insert_in_ voicelist(scr)
typ(k):='disc', typ(k+1):='disc'
ic(k):=0
sat(icr):=scr, ic(scr):=icr
typ(scr):='voice'
78
/* Same as handling of reassignment of a transparent call turned into a voice call and an implicit disconnection of a voice call */
T1511170-92

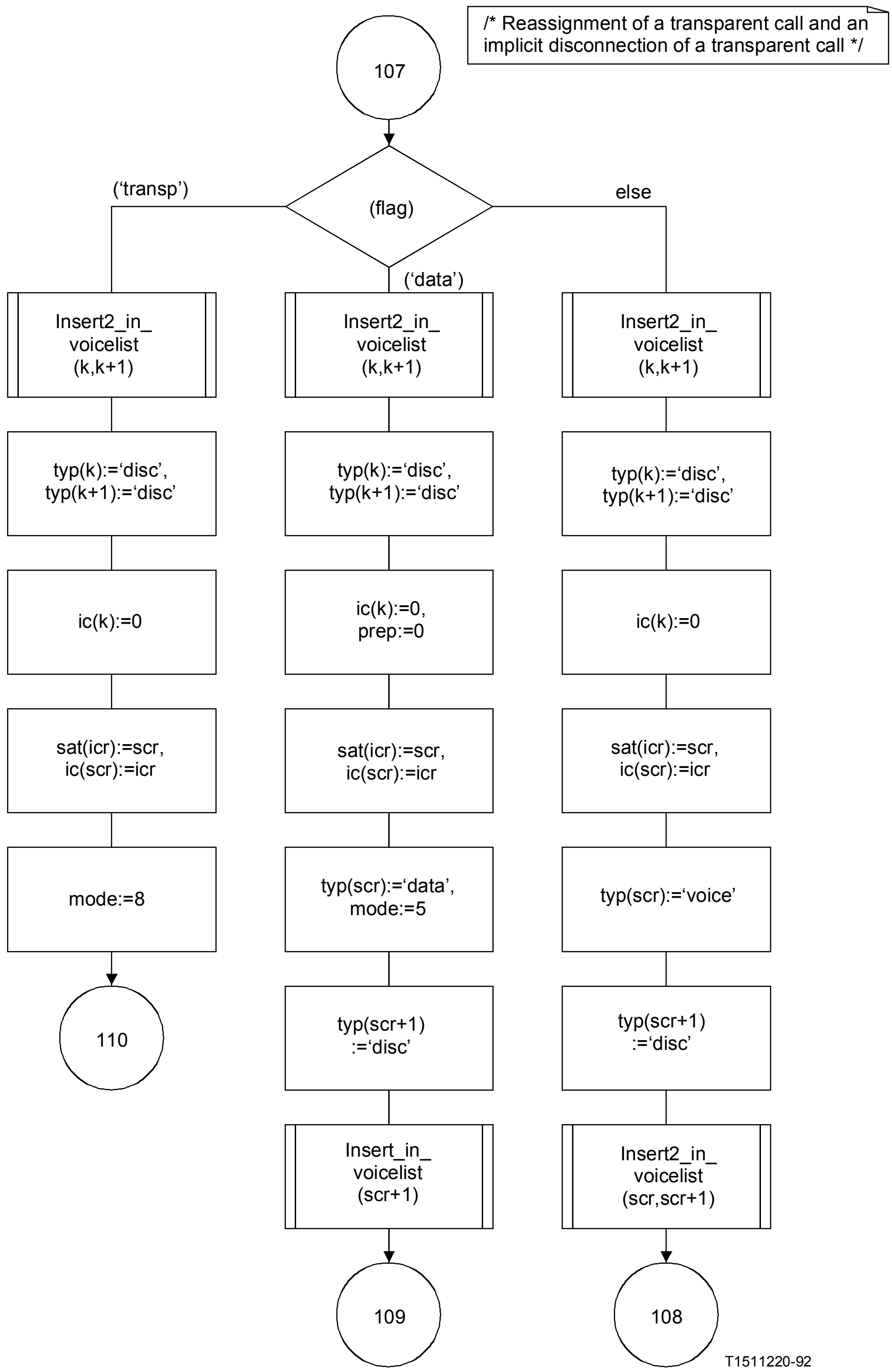

102
/* Handling of connection of icr as a transparent call turned into a data call and an implicit disconnection of nr as a data call */
(ad(icr))
(0)
else
(ad(nr))
(0)
else
Release TO DECPID (dec(ad(nr)))
dec(ad(nr)):=0
(ad(nr))
(0)
else
dcd:=dec(ad(icr))
Release TO DECPID (dec(ad(nr)))
dec(ad(nr)):=0, dcd:=dec(ad(icr))
Rxtransprel (ad(icr))
Rxdata (ad(icr))
Seize (ad(icr),mode) TO DECPID (dcd)
sat(nr):=0
5
T1511180-92

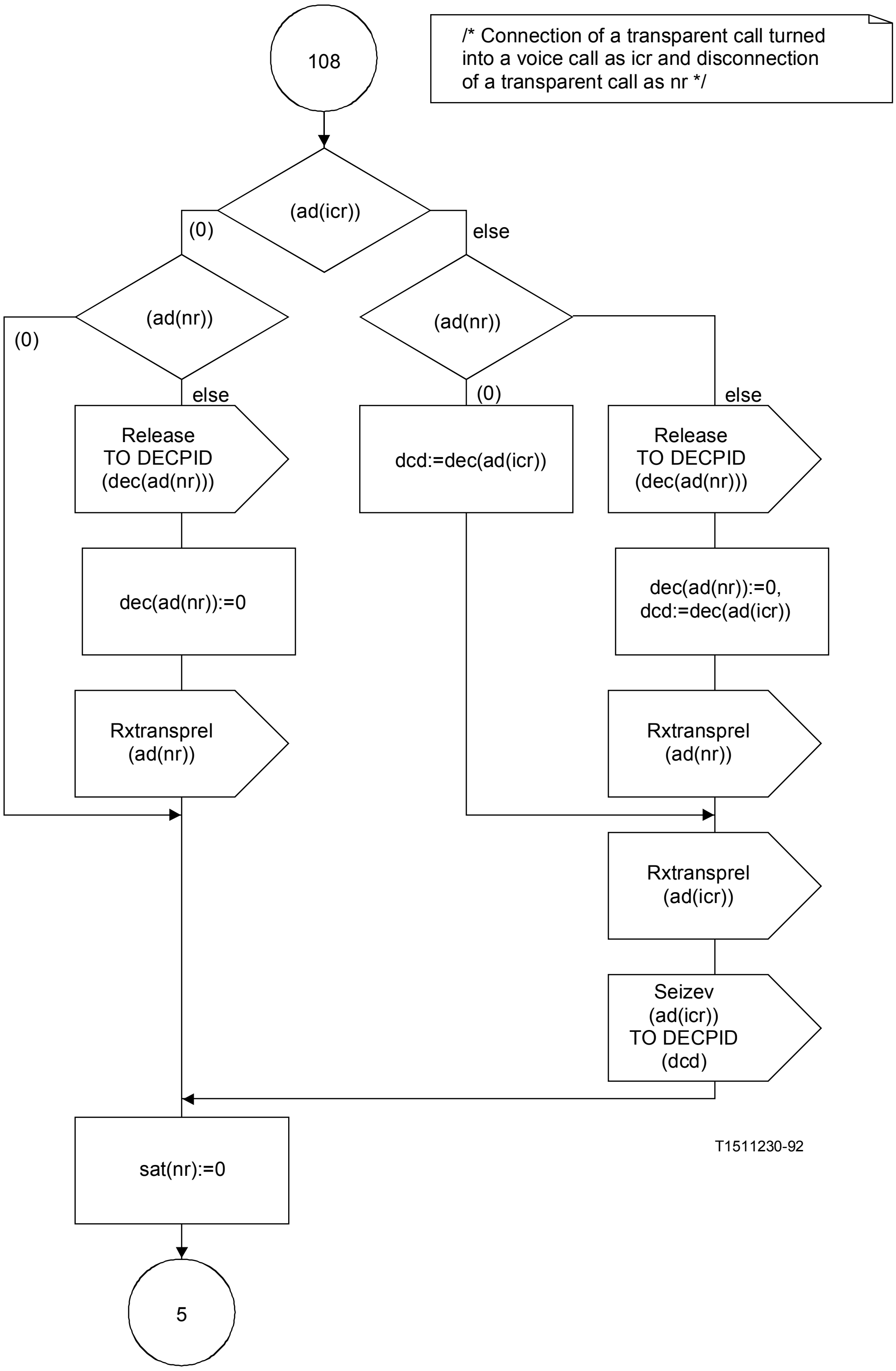

/* Handling of reassignment of a transparent call and an implicit disconnection of a data call */
103
(typ(scr+1))
('data')
('bank')
else
Remove_from_ bitbankarray (scr+1,btot)
Remove_from_ voicelist (scr,scr+1)
typ(scr):='transp', typ(scr+1):='transp', nr1:=ic(scr+1)
ic(scr):=icr, sat(icr):=scr, mode:=8
ic(k):=0, typ(k):='disc', typ(k+1):='disc'
Insert2_in_ voicelist (k,k+1)
104
T1511190-92

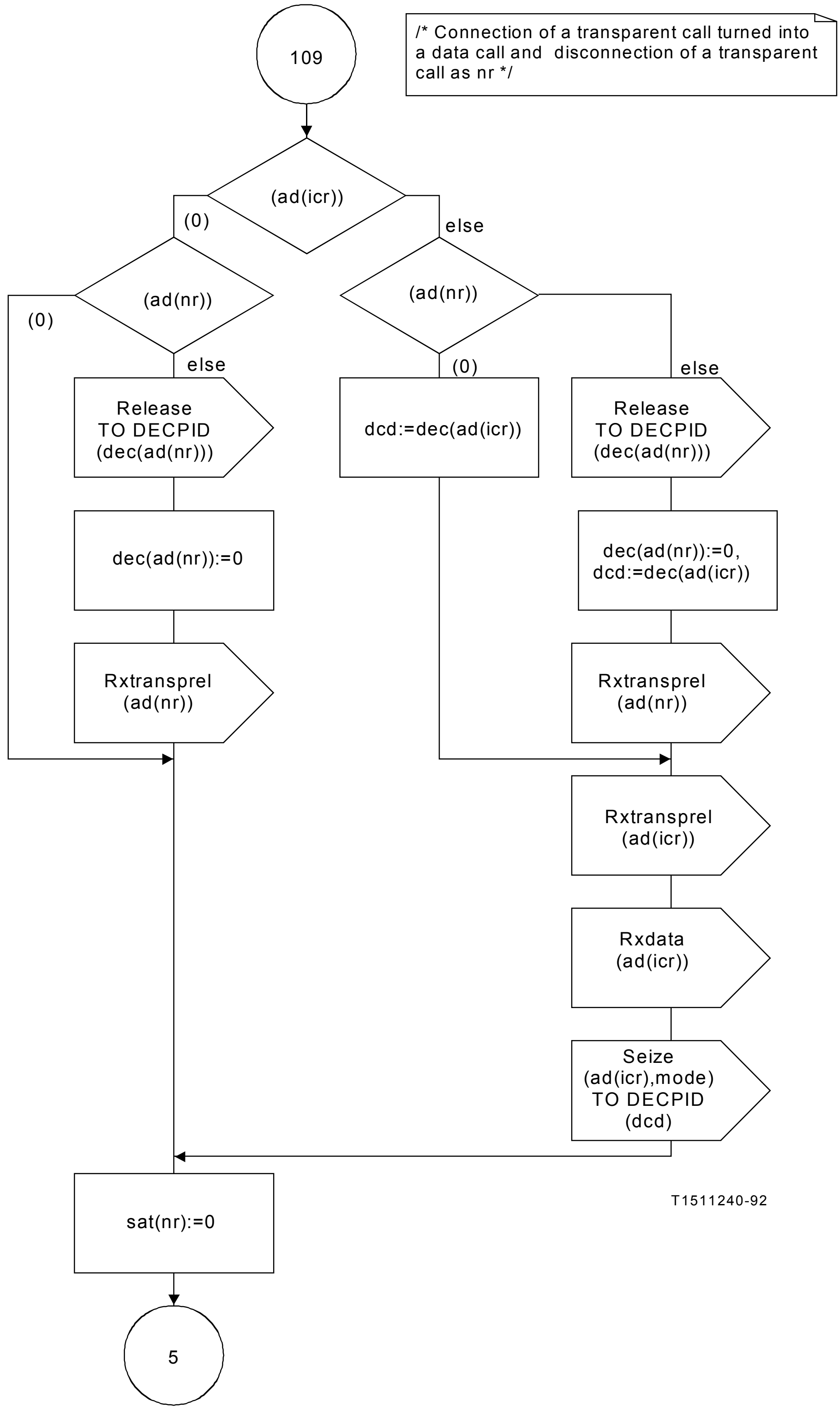

/* Handling of connection to scr+1 */
104
(nr1)
else
(0)
(ad(nr1))
else
(0)
Release
TO DECPID
(dec(ad(nr1)))
dec(ad(nr1)):=0
ic(scr+1):=0
sat(nr1):=0,
ic(scr+1):=0
72
/* Same as reassignment of icr as a data call
and an implicit disconnection of nr as a voice call */
T1511200-92

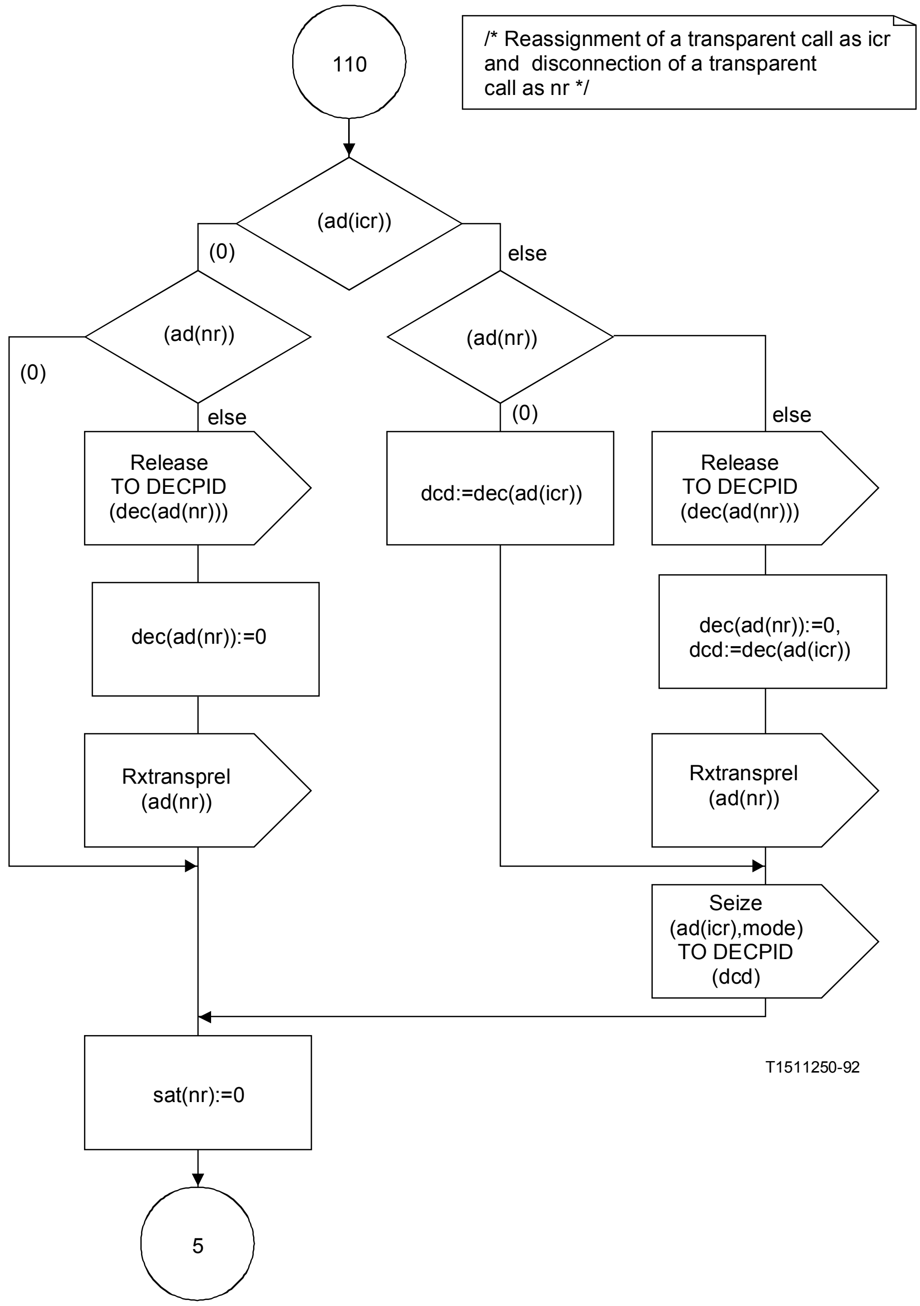


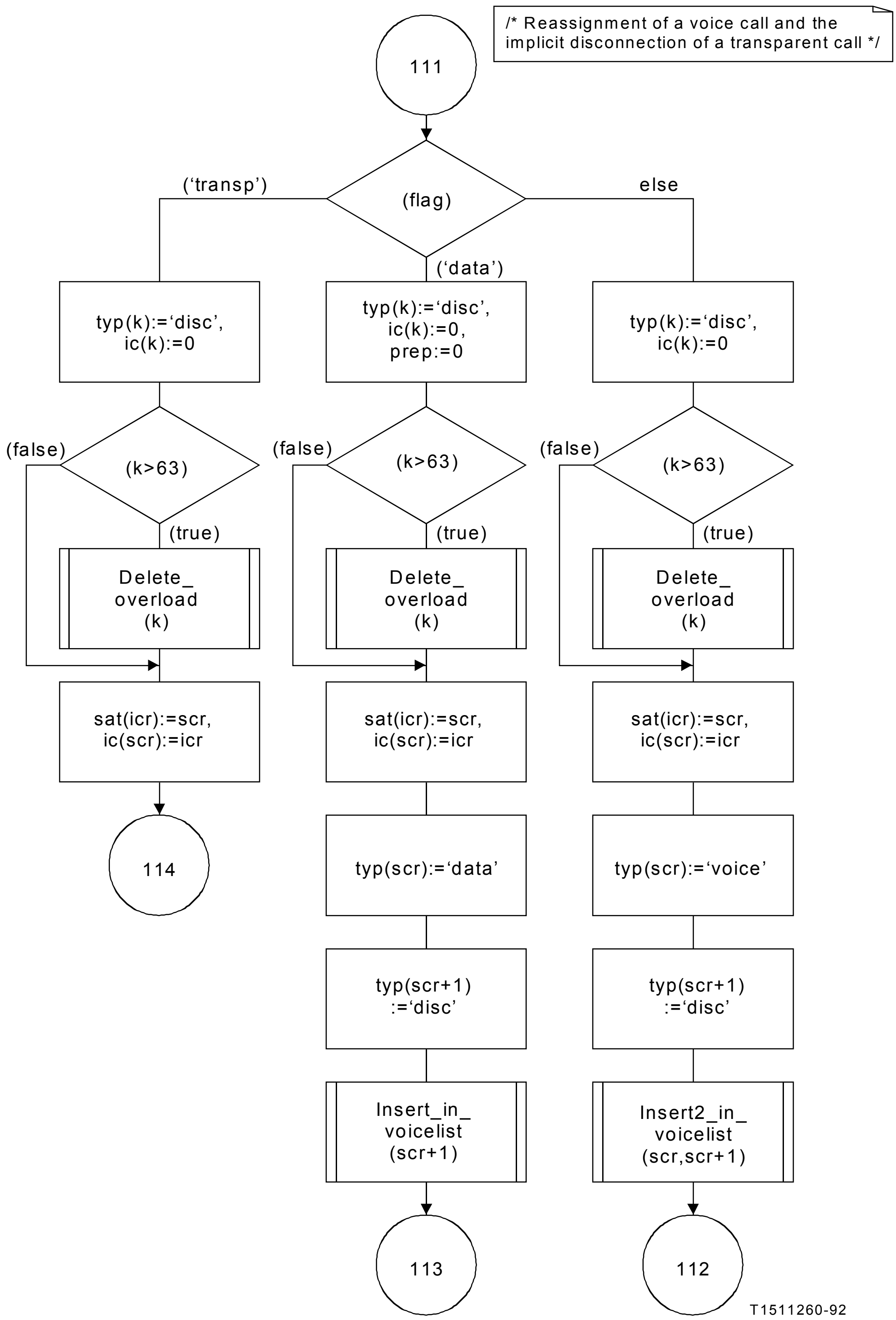
PROCESS RUD
70 (80)
106
/* Handling of connection of icr as a transparent call and an implicit disconnection of nr as voice */
(nr=icr and k=scr)
(false)
/* Actual reassignment */
107
(true)
(flag)
('data')
('transp')
else
Insert_in_ voicelist (scr+1)
5
Insert2_in_ voicelist (scr,scr+1)
/* Refresh */
typ(scr+1) :='disc'
typ(scr+1) :='disc'
typ(scr):='data', mode:=5, prep:=0
typ(scr):='voice'
(ad(icr))
else
(ad(icr))
Rxtransprel (ad(icr))
(0)
(0)
else
Rxdata (ad(icr))
Rxtransprel (ad(icr))
Seize (ad(icr),mode) TO DECPID (dec(ad(icr)))
Seizev (ad(icr)) TO DECPID (dec(ad(icr)))
5
5
/* flag change to voice */
/* flag change to voice */
T1511210-92

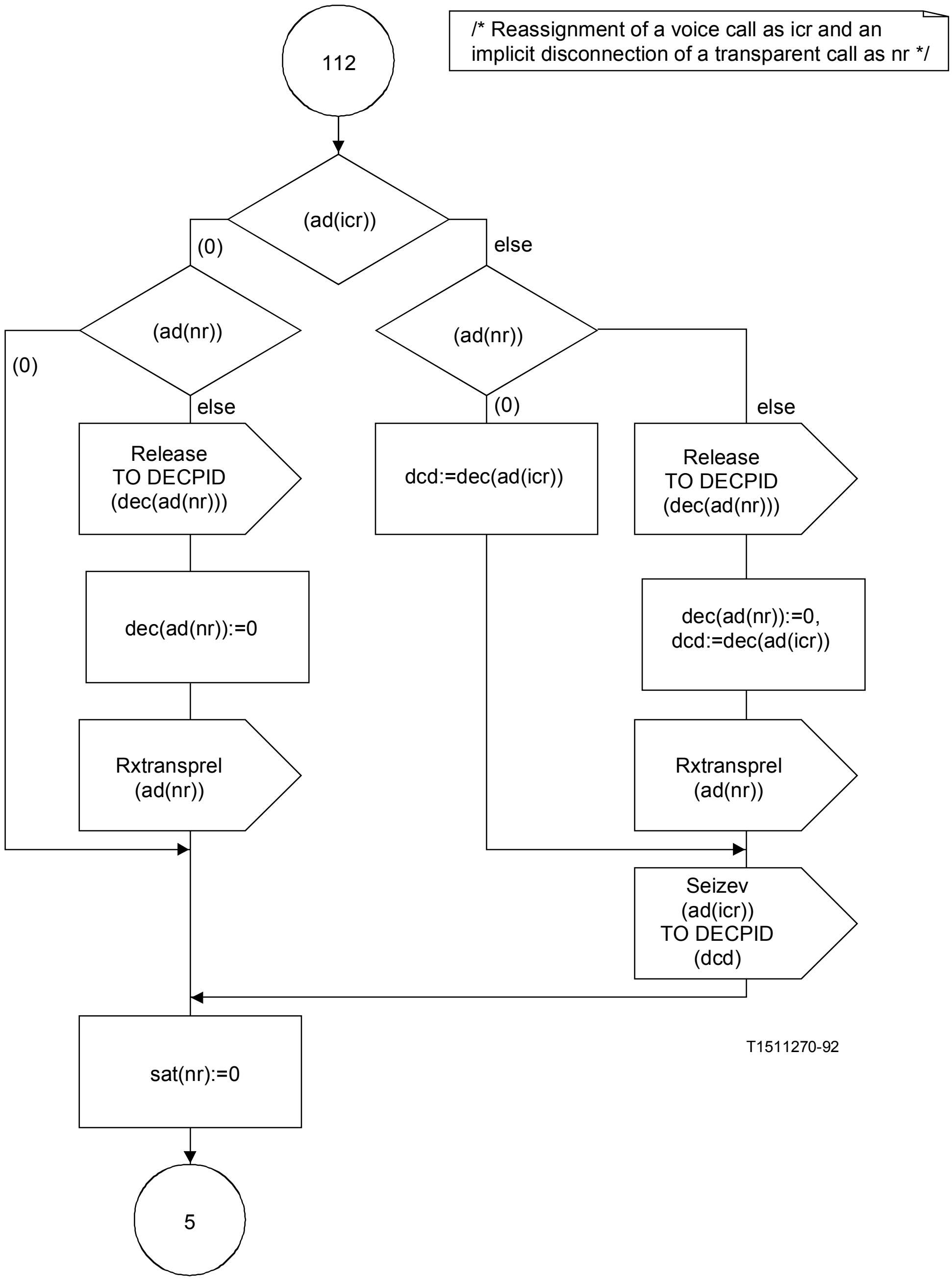

/* Reassignment of a transparent call and an implicit disconnection of a transparent call */
107
(flag)
('transp')
('data')
else
Insert2_in_ voicelist (k,k+1)
typ(k):='disc', typ(k+1):='disc'
ic(k):=0
sat(icr):=scr, ic(scr):=icr
mode:=8
110
Insert2_in_ voicelist (k,k+1)
typ(k):='disc', typ(k+1):='disc'
ic(k):=0, prep:=0
sat(icr):=scr, ic(scr):=icr
typ(scr):='data', mode:=5
typ(scr+1) :='disc'
Insert_in_ voicelist (scr+1)
109
Insert2_in_ voicelist (k,k+1)
typ(k):='disc', typ(k+1):='disc'
ic(k):=0
sat(icr):=scr, ic(scr):=icr
typ(scr):='voice'
typ(scr+1) :='disc'
Insert2_in_ voicelist (scr,scr+1)
108
T1511220-92

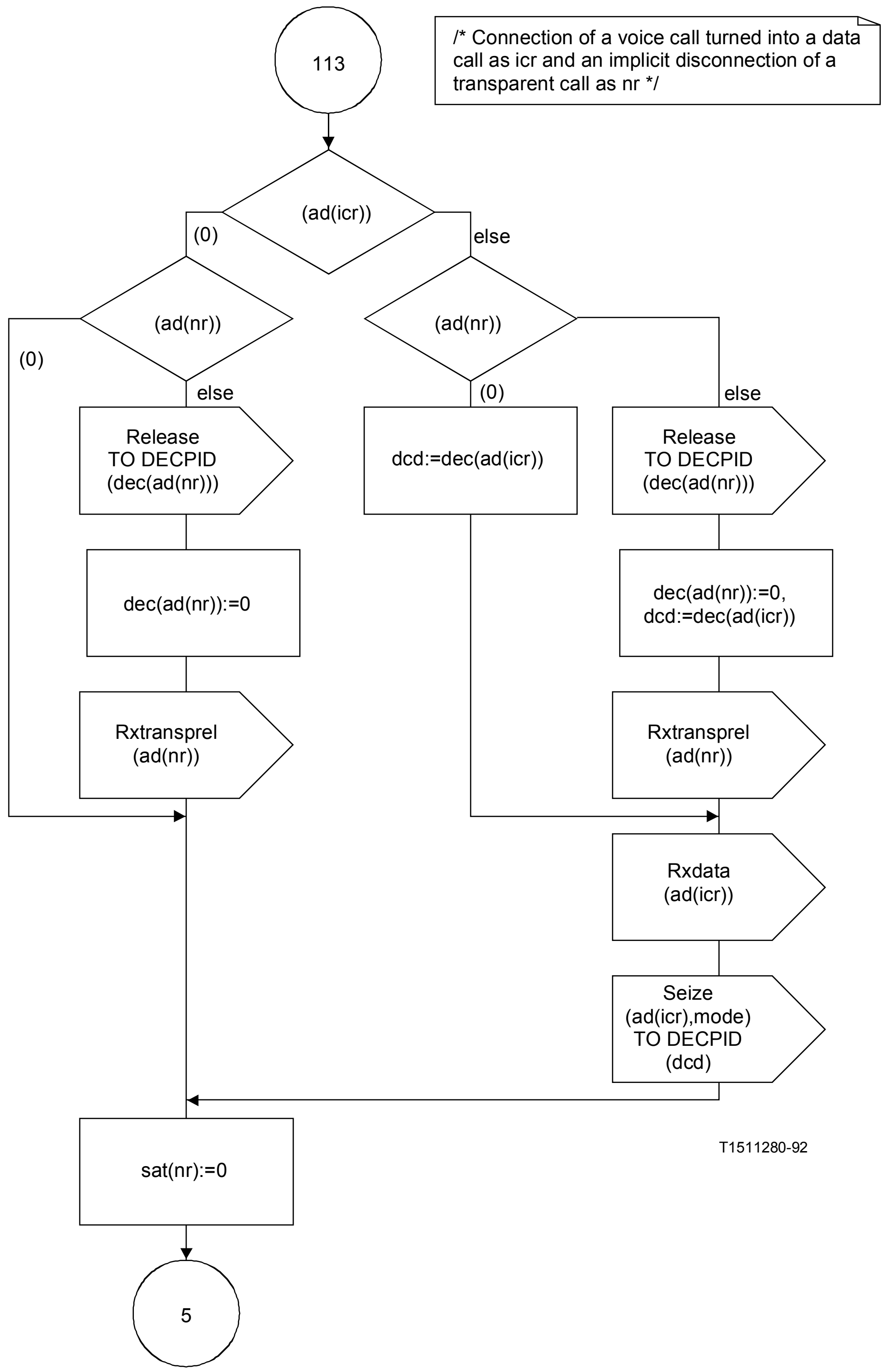

/* Connection of a transparent call turned into a voice call as icr and disconnection of a transparent call as nr */
108
(ad(icr))
(0)
else
(ad(nr))
(0)
else
(ad(nr))
(0)
else
Release TO DECPID (dec(ad(nr)))
dcd:=dec(ad(icr))
Release TO DECPID (dec(ad(nr)))
dec(ad(nr)):=0
dec(ad(nr)):=0, dcd:=dec(ad(icr))
Rxtransprel (ad(nr))
Rxtransprel (ad(nr))
Rxtransprel (ad(icr))
Seizev (ad(icr)) TO DECPID (dcd)
sat(nr):=0
5
T1511230-92


109
/* Connection of a transparent call turned into a data call and disconnection of a transparent call as nr */
(ad(icr))
(0)
else
(ad(nr))
(0)
else
(ad(nr))
(0)
else
Release TO DECPID (dec(ad(nr)))
dcd:=dec(ad(icr))
Release TO DECPID (dec(ad(nr)))
dec(ad(nr)):=0
dec(ad(nr)):=0, dcd:=dec(ad(icr))
Rxtransprel (ad(nr))
Rxtransprel (ad(nr))
Rxtransprel (ad(icr))
Rxdata (ad(icr))
Seize (ad(icr),mode) TO DECPID (dcd)
sat(nr):=0
5
T1511240-92


110
/* Reassignment of a transparent call as icr and disconnection of a transparent call as nr */
(ad(icr))
(0)
else
(ad(nr))
(0)
else
(ad(nr))
(0)
else
Release TO DECPID (dec(ad(nr)))
dcd:=dec(ad(icr))
Release TO DECPID (dec(ad(nr)))
dec(ad(nr)):=0
dec(ad(nr)):=0, dcd:=dec(ad(icr))
Rxtransprel (ad(nr))
Rxtransprel (ad(nr))
Seize (ad(icr),mode) TO DECPID (dcd)
sat(nr):=0
5
T1511250-92

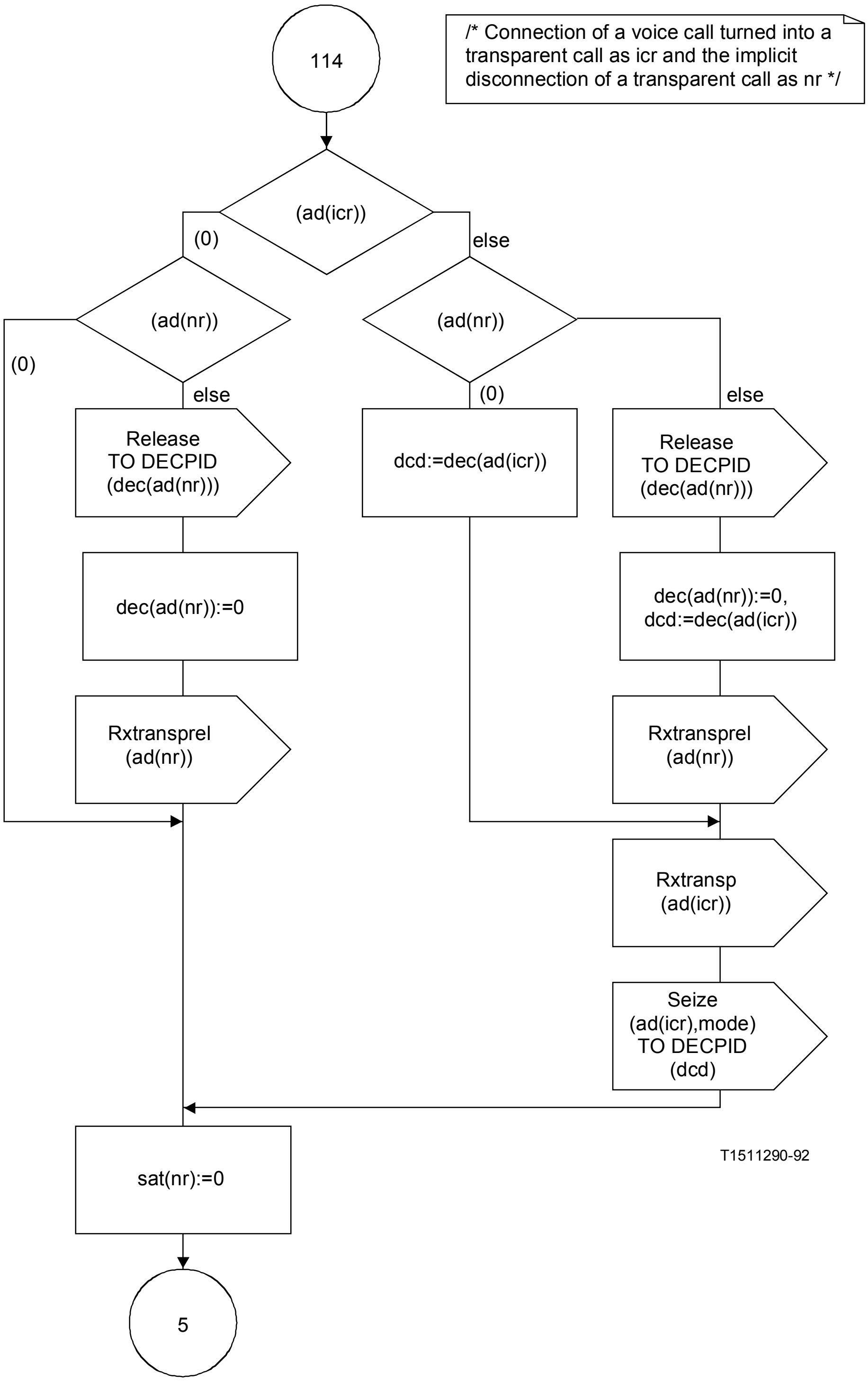

/* Reassignment of a voice call and the implicit disconnection of a transparent call */
111
(flag)
('transp')
('data')
else
typ(k):='disc', ic(k):=0
typ(k):='disc', ic(k):=0, prep:=0
typ(k):='disc', ic(k):=0
(k>63)
(false)
(true)
Delete_ overload (k)
sat(icr):=scr, ic(scr):=icr
114
typ(scr):='data'
typ(scr):='voice'
typ(scr+1) :='disc'
Insert_in_ voicelist (scr+1)
Insert2_in_ voicelist (scr,scr+1)
113
112
T1511260-92


/* Reassignment of a voice call as icr and an implicit disconnection of a transparent call as nr */
112
(ad(icr))
(0)
else
(ad(nr))
(0)
else
Release TO DECPID (dec(ad(nr)))
dec(ad(nr)):=0
Rxtransprel (ad(nr))
(ad(nr))
(0)
else
dcd:=dec(ad(icr))
Release TO DECPID (dec(ad(nr)))
dec(ad(nr)):=0, dcd:=dec(ad(icr))
Rxtransprel (ad(nr))
Seizev (ad(icr)) TO DECPID (dcd)
sat(nr):=0
5
T1511270-92

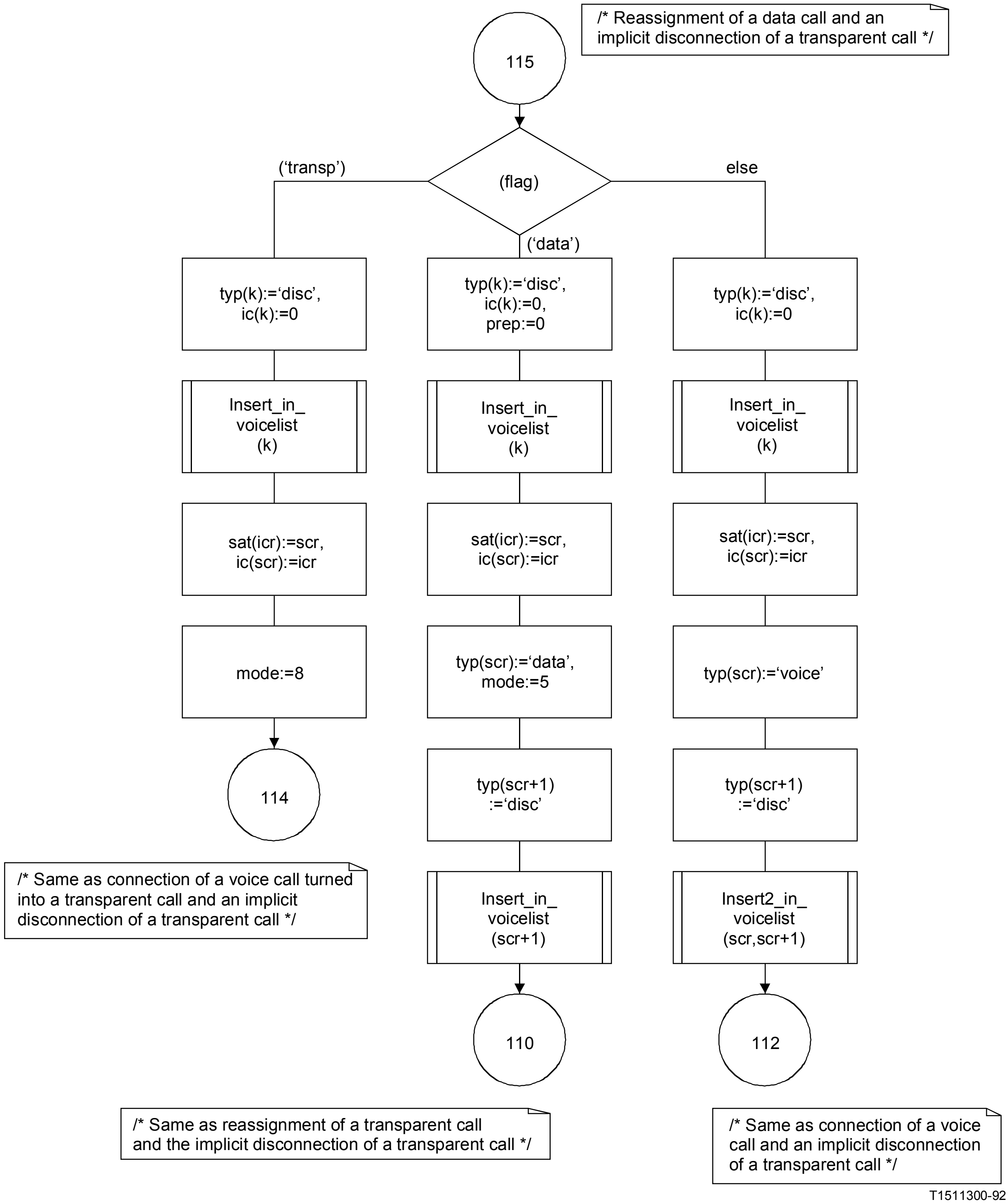


113
/* Connection of a voice call turned into a data call as icr and an implicit disconnection of a transparent call as nr */
(ad(icr))
(0)
else
(ad(nr))
(0)
else
Release TO DECPID (dec(ad(nr)))
dec(ad(nr)):=0
Rxtransprel (ad(nr))
(ad(nr))
(0)
else
dcd:=dec(ad(icr))
Release TO DECPID (dec(ad(nr)))
dec(ad(nr)):=0, dcd:=dec(ad(icr))
Rxtransprel (ad(nr))
Rxdata (ad(icr))
Seize (ad(icr),mode) TO DECPID (dcd)
sat(nr):=0
5

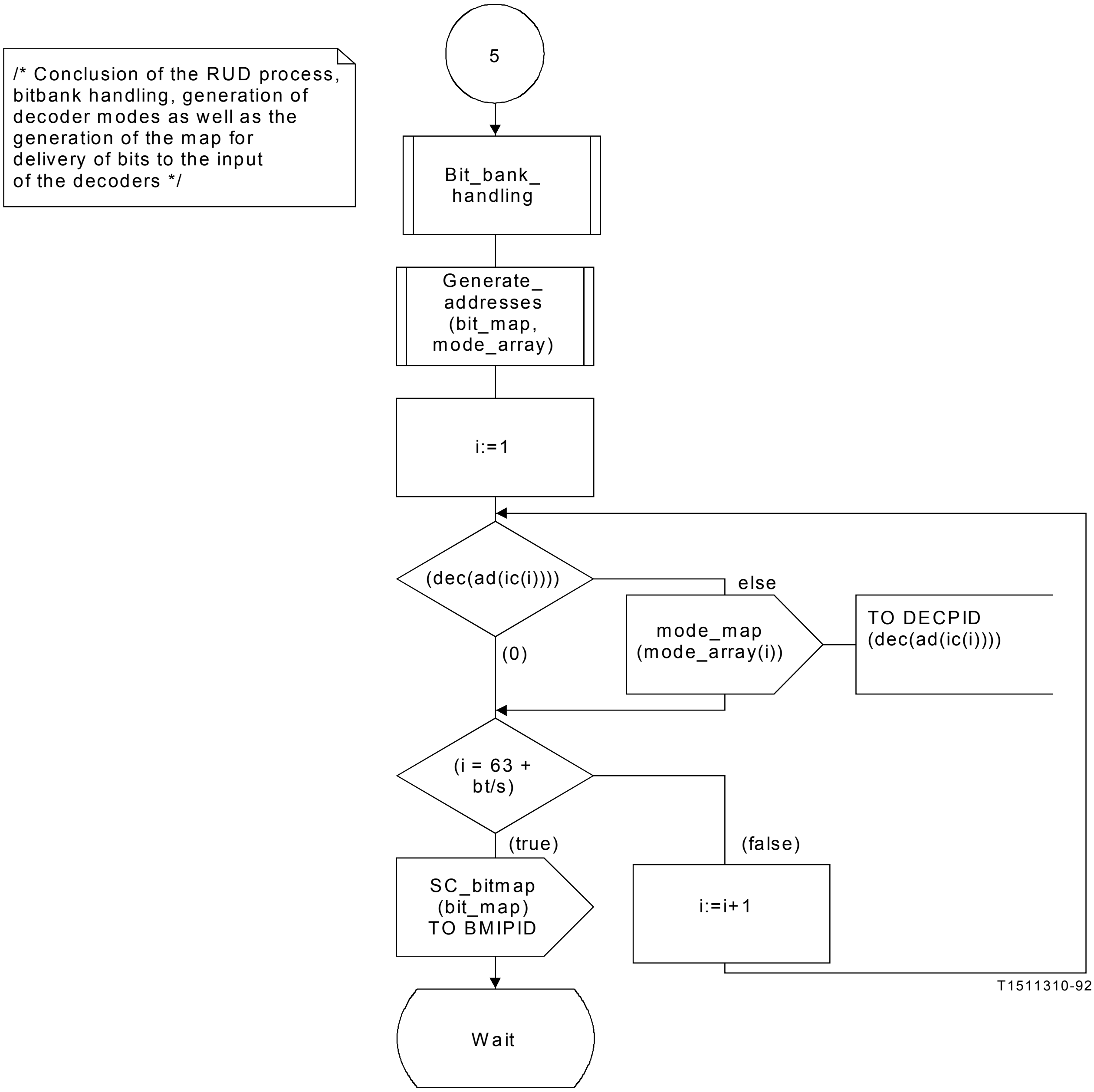
T1511280-92



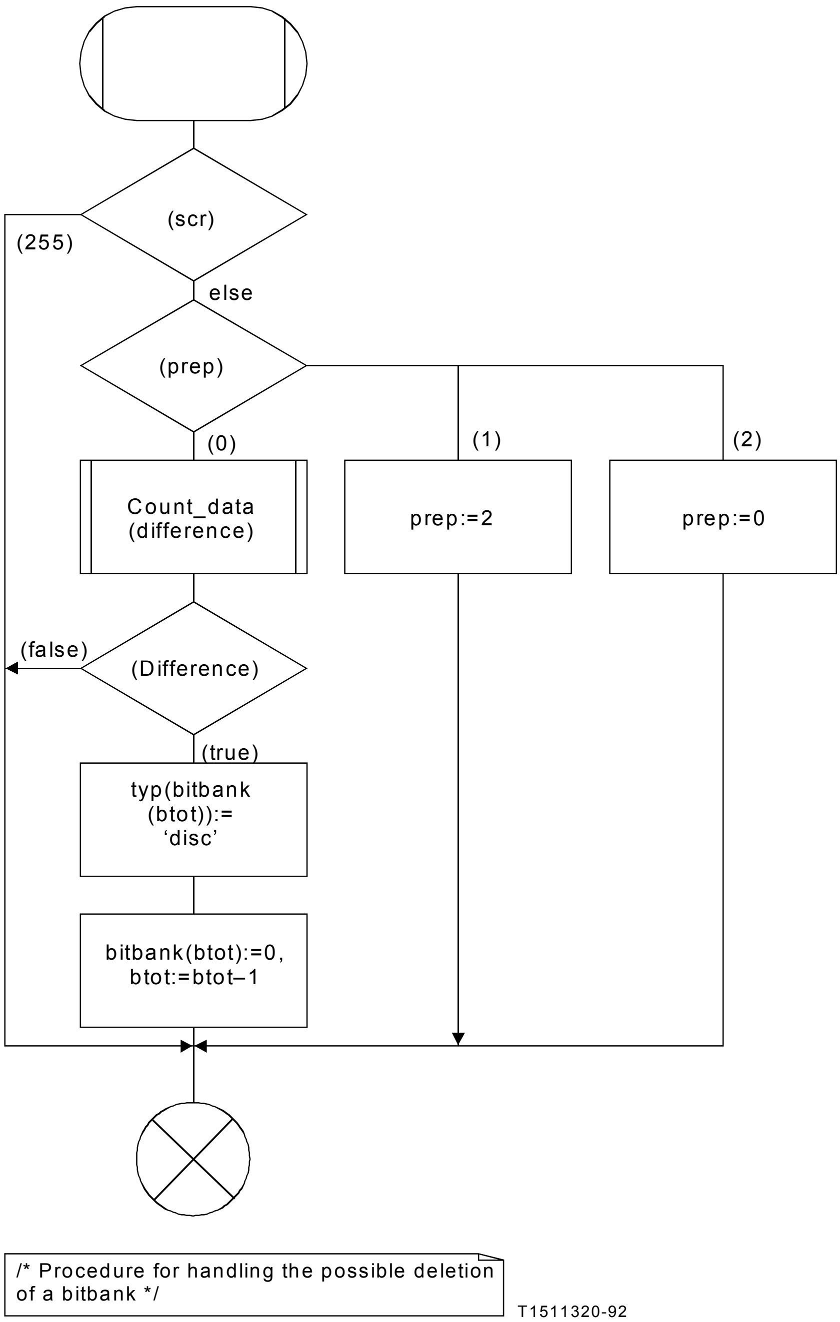
/* Connection of a voice call turned into a transparent call as icr and the implicit disconnection of a transparent call as nr */
114
(ad(icr))
(0)
else
(ad(nr))
(0)
else
Release TO DECPID (dec(ad(nr)))
dec(ad(nr)):=0
Rxtransprel (ad(nr))
(ad(nr))
(0)
else
dcd:=dec(ad(icr))
Release TO DECPID (dec(ad(nr)))
dec(ad(nr)):=0, dcd:=dec(ad(icr))
Rxtransprel (ad(nr))
Rxtransp (ad(icr))
Seize (ad(icr),mode) TO DECPID (dcd)
sat(nr):=0
5
T1511290-92

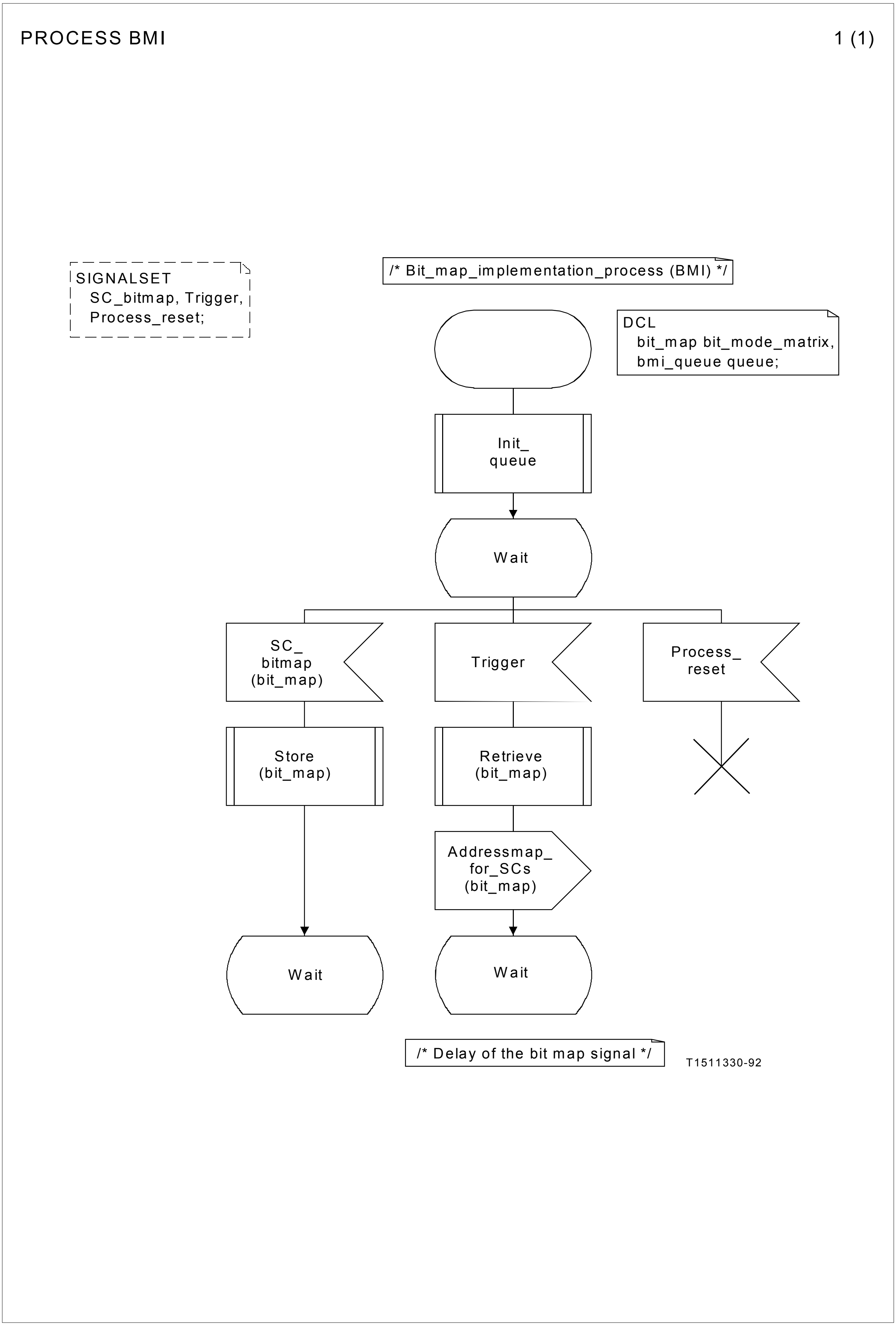


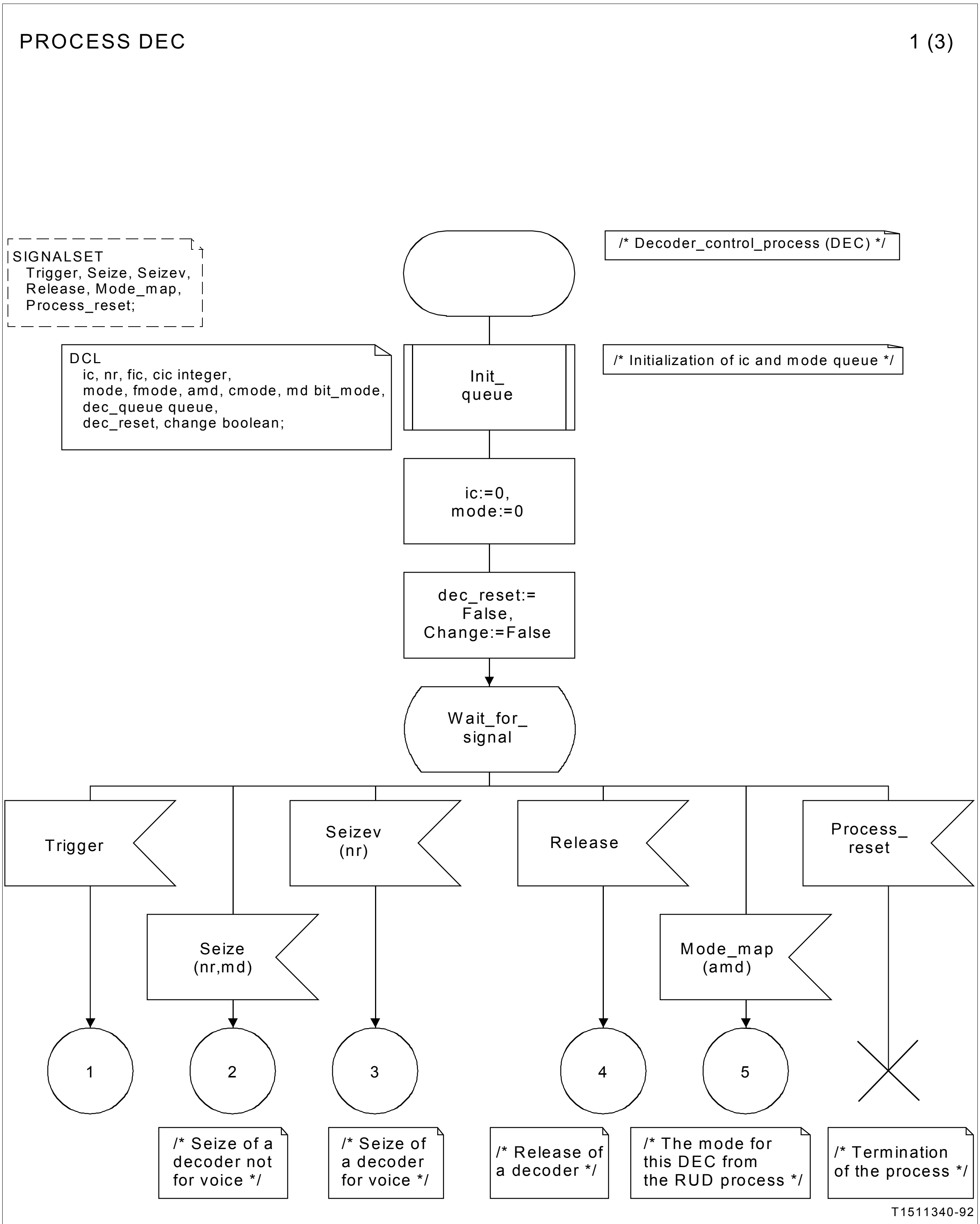
/* Reassignment of a data call and an implicit disconnection of a transparent call */
115
(flag)
('transp')
('data')
else
typ(k):='disc', ic(k):=0
typ(k):='disc', ic(k):=0, prep:=0
typ(k):='disc', ic(k):=0
Insert_in_ voicelist (k)
Insert_in_ voicelist (k)
Insert_in_ voicelist (k)
sat(icr):=scr, ic(scr):=icr
sat(icr):=scr, ic(scr):=icr
sat(icr):=scr, ic(scr):=icr
mode:=8
typ(scr):='data', mode:=5
typ(scr):='voice'
114
typ(scr+1) :='disc'
typ(scr+1) :='disc'
/* Same as connection of a voice call turned into a transparent call and an implicit disconnection of a transparent call */
Insert_in_ voicelist (scr+1)
Insert2_in_ voicelist (scr,scr+1)
110
112
/* Same as reassignment of a transparent call and the implicit disconnection of a transparent call */
/* Same as connection of a voice call and an implicit disconnection of a transparent call */
T1511300-92

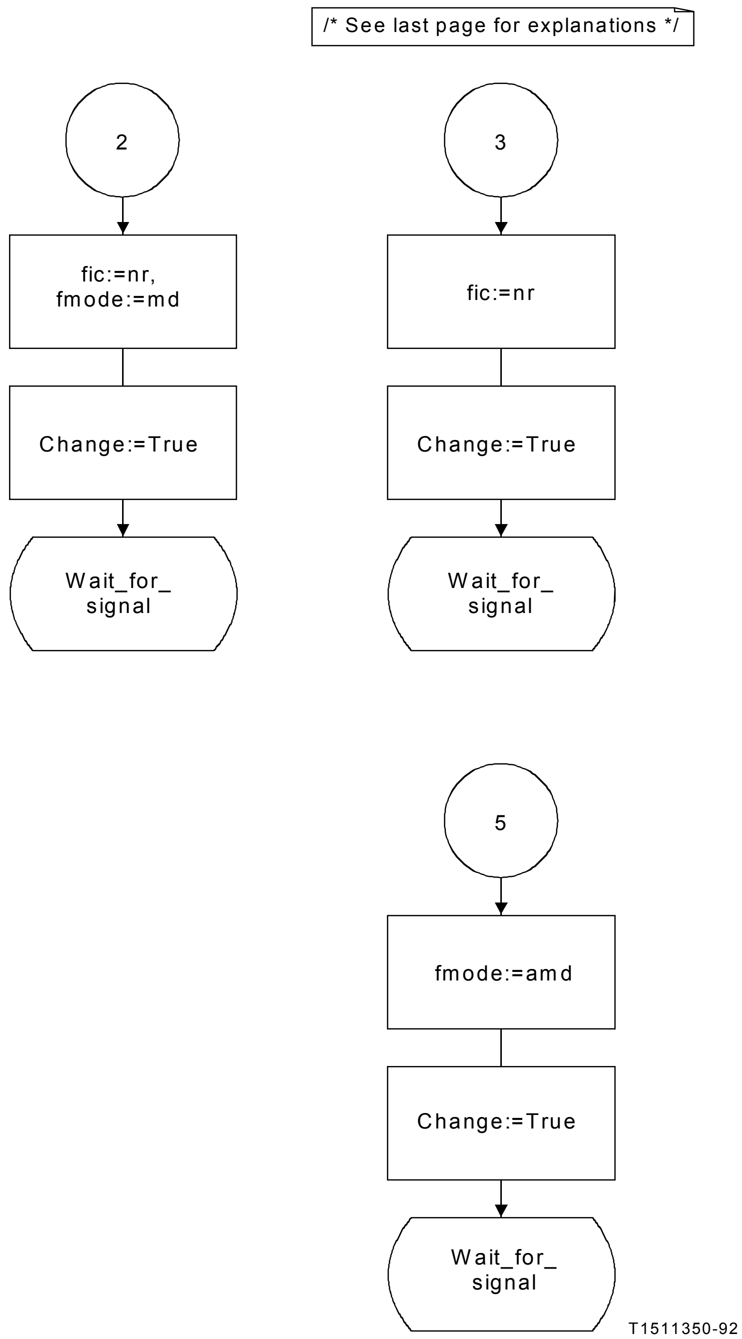

/* Conclusion of the RUD process, bitbank handling, generation of decoder modes as well as the generation of the map for delivery of bits to the input of the decoders */
5
Bit_bank_ handling
Generate_ addresses (bit_map, mode_array)
i:=1
(dec(ad(ic(i))))
else
mode_map (mode_array(i))
TO DECPID (dec(ad(ic(i))))
(0)
(i = 63 + bt/s)
(true)
(false)
SC_bitmap (bit_map) TO BMIPID
i:=i+1
Wait
T1511310-92

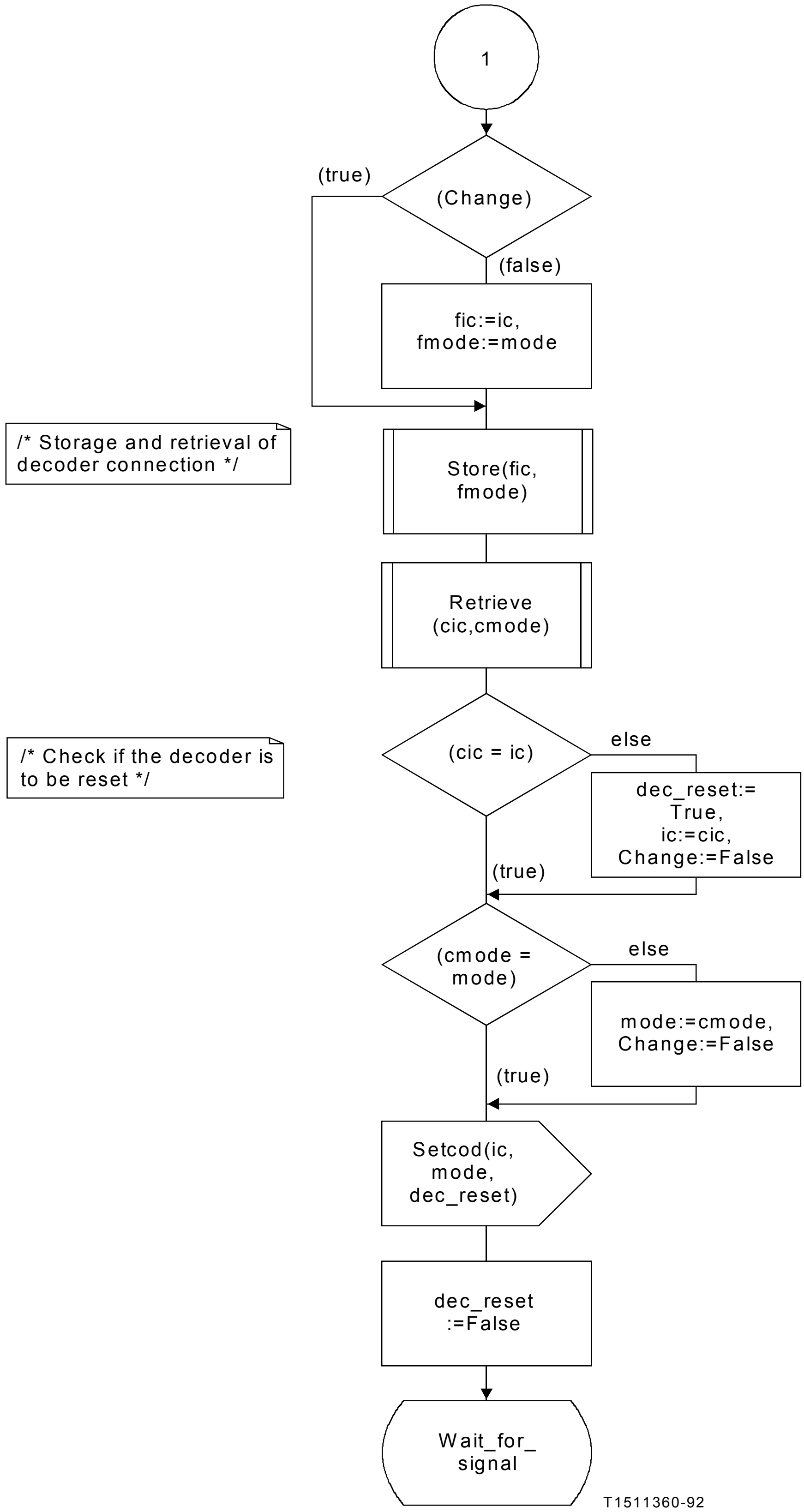
PROCEDURE BIT_BANK_HANDLING 1 (1)

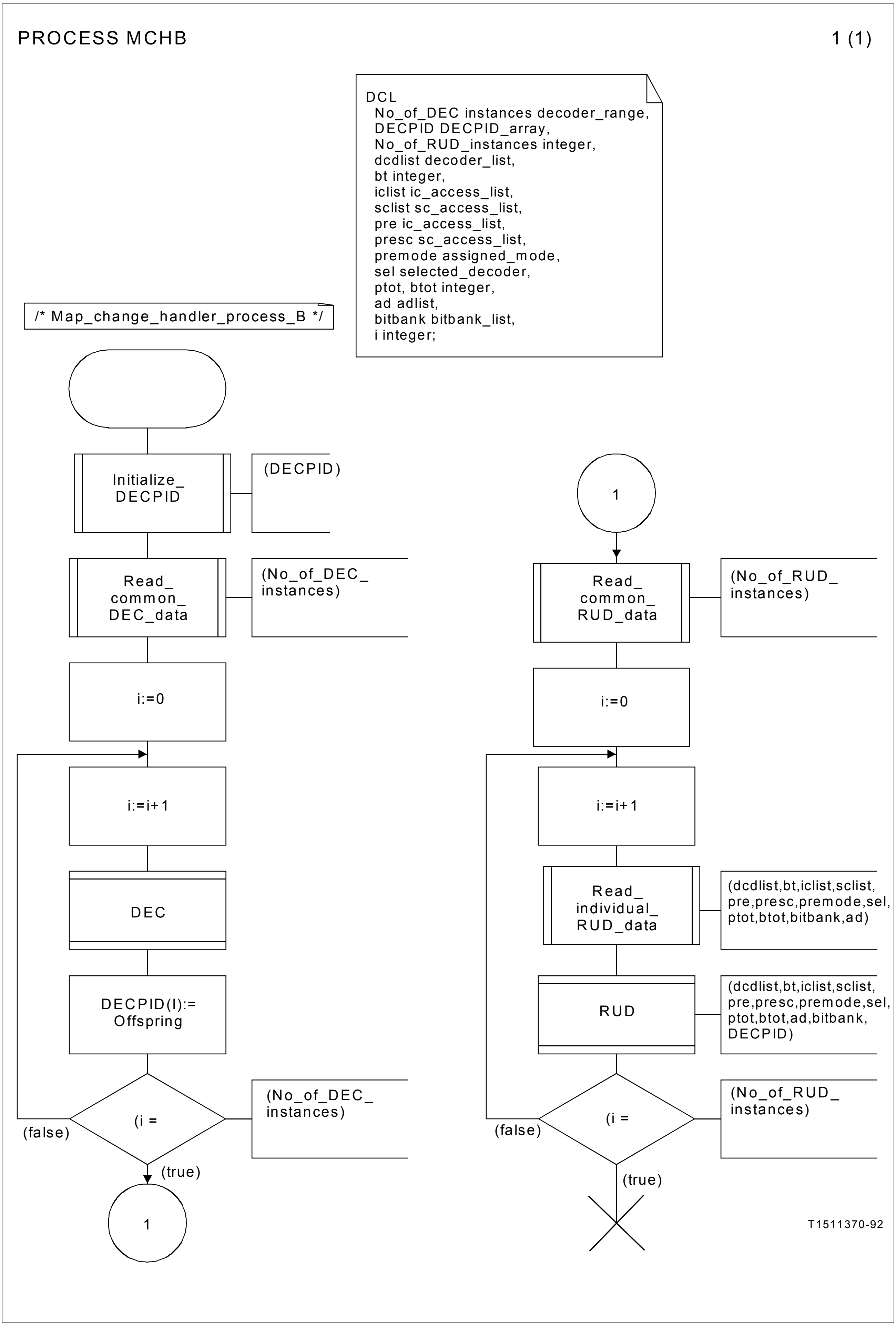



PROCESS BMI
1 (1)
SIGNALSET
SC_bitmap, Trigger,
Process_reset;
/* Bit_map_implementation_process (BMI) */
DCL
bit_map bit_mode_matrix,
bmi_queue queue;
Init_
queue
Wait
SC_
bitmap
(bit_map)
Trigger
Process_
reset
Store
(bit_map)
Retrieve
(bit_map)
Addressmap_
for_SCs
(bit_map)
Wait
Wait
/* Delay of the bit map signal */
T1511330-92


PROCESS DEC
1 (3)
SIGNALSET
Trigger, Seize, Seizev,
Release, Mode_map,
Process_reset;
/* Decoder_control_process (DEC) */
DCL
ic, nr, fic, cic integer,
mode, fmode, amd, cmode, md bit_mode,
dec_queue queue,
dec_reset, change boolean;
Init_
queue
/* Initialization of ic and mode queue */
ic:=0,
mode:=0
dec_reset:=
False,
Change:=False
Wait_for_
signal
Trigger
Seize
(nr,md)
Seizev
(nr)
Release
Mode_map
(amd)
Process_
reset
1
2
3
4
5
/* Seize of a
decoder not
for voice */
/* Seize of
a decoder
for voice */
/* Release of
a decoder */
/* The mode for
this DEC from
the RUD process */
/* Termination
of the process */
T1511340-92


/* See last page for explanations */
2
fic:=nr,
fmode:=md
Change:=True
Wait_for_
signal
3
fic:=nr
Change:=True
Wait_for_
signal
4
fic:=0,
fmode:=0
Change:=True
Wait_for_
signal
5
fmode:=amd
Change:=True
Wait_for_
signal
T1511350-92


1
(true)
(Change)
(false)
fic:=ic,
fmode:=mode
/* Storage and retrieval of
decoder connection */
Store(fic,
fmode)
Retrieve
(cic,cmode)
/* Check if the decoder is
to be reset */
(cic = ic)
else
dec_reset:=
True,
ic:=cic,
Change:=False
(true)
(cmode =
mode)
else
mode:=cmode,
Change:=False
(true)
Setcod(ic,
mode,
dec_reset)
dec_reset
:=False
Wait_for_
signal
T1511360-92


PROCESS MCHB
1 (1)
DCL
No_of_DEC instances decoder_range,
DECPID DECPID_array,
No_of_RUD_instances integer,
dcdlist decoder_list,
bt integer,
iclist ic_access_list,
sclist sc_access_list,
pre ic_access_list,
presc sc_access_list,
premode assigned_mode,
sel selected_decoder,
ptot, btot integer,
ad adlist,
bitbank bitbank_list,
i integer;
/* Map_change_handler_process_B */
Initialize_ DECPID
(DECPID)
Read_ common_ DEC_data
(No_of_DEC_ instances)
i:=0
i:=i+1
DEC
DECPID(I):= Offspring
(i =
(No_of_DEC_ instances)
(false)
(true)
1
1
Read_ common_ RUD_data
(No_of_RUD_ instances)
i:=0
i:=i+1
Read_ individual_ RUD_data
(dcdlist,bt,iclist,sclist, pre,presc,premode,sel, ptot,btot,bitbank,ad)
RUD
(dcdlist,bt,iclist,sclist, pre,presc,premode,sel, ptot,btot,ad,bitbank, DECPID)
(i =
(No_of_RUD_ instances)
(false)
(true)
T1511370-92

A.3.3 *Logic diagrams for on-demand transparent circuit handling*

The logic diagrams in this section of Annex A.3 are supplemental to the description of the on-demand transparent circuit handling given in section 8 of the Recommendation and includes a user optional TCH/DLC interaction override facility. The on-demand transparent circuit handling procedure is contained within one block:

TCH: transparent-channel-handling-block.

A.3.3.1 *The TCH block*

The TCH block contains one process and the following signals:

L5: Transpreq, Transprel

L30: S64, R64

L31: S64Ack, S64Nack, R64Ack Out-of-Service, Back-in-Service

L32: AD64, DD64

L33: Process-reset

L34: Override, No-Override

L35: Man-Reset

L52: Rxtranspreq, Rxtransprel

The states used by the process are the following:

(TCH) transparent circuit handling process (0,)

Not-64, Blocked, Circuit Out-of-Service, Connect-called-64, Connect-calling-64, Establish-forward-64, Disestablish-forward-64, Disestablish-backward-64, Autorecovery-64, Spurious-recovery.

The signals have the following meaning:

L5 – See previous explanation in Appendix I.

L30 – The S64 and R64 signals arrive from the SIU where they have been translated from the format used by the real originator, namely the ISC exchange. They imply that a transparent call shall be established or terminated.

L31 – The S64Ack, S64Nack and R64Ack signals are the responses that the TCH process generate upon reception of the L30 signals. The Out-of-Service and Back-in-Service signals are used to indicate to and from unavailability.

L32 – The AD64 and DD64 signals arrive from the DLC block in the DCME and indicate that the unit should stop accepting any new transparent requests from the ISC or that the DCME should start accepting requests for transparent connections from the ISC, respectively.

L33 – The Process-reset signal arrives from the MCH and causes a termination of the process instance that receives it.

L34 – The Override and No-Override signals are generated manually by the operator and indicate that the TCH/DLC interaction is disabled or enabled, respectively (see Note).

L35 – The Man-Reset signal is used by the operator to place an Out-of-Service circuit to Back-in-Service.

L52 – See previous explanation in Appendix II.

The TCH process is assumed to be created by the MCH at system start-up and after a map change. There is one instance of the process for every local IC handled by the DCME, according to its configuration data. The process uses the following variables:

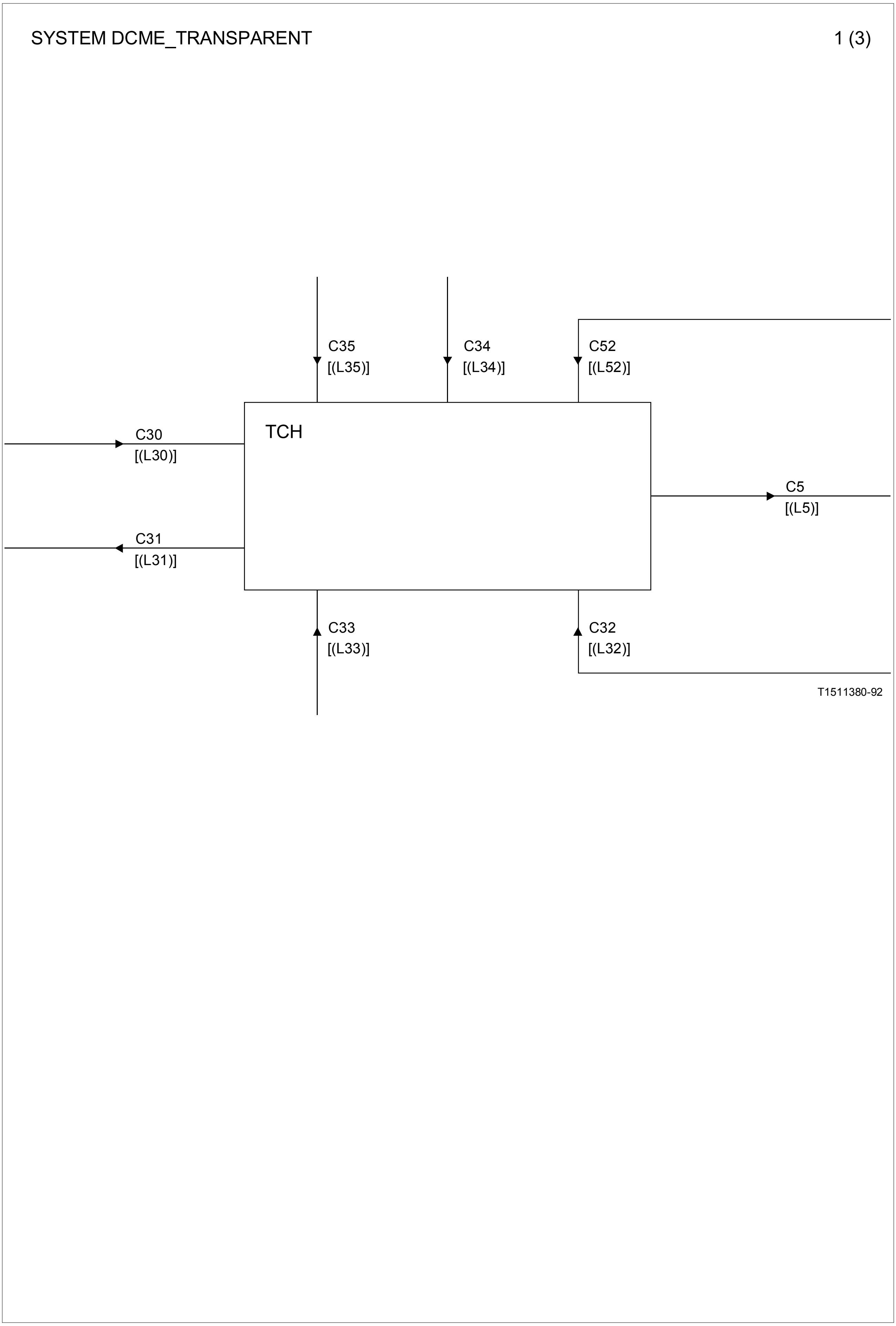
disabled – This Boolean variable is TRUE when the TCH/DLC interaction is disabled, otherwise FALSE. It is set and reset by arrival of the signals Override and No-Override, respectively (see Note).

dlcon – This variable stores the current DLC condition for the IC handled by the process. If DLC is "ON" the variable is True, if "OFF" False.

T1, T2, T3, T4 – Different timer values used by the TCH process instance.

ti – A timer variable.

Note – If the manual override is not implemented these signals do not exist. If the manual override is not implemented the variable is always FALSE.



SYSTEM DCME_TRANSPARENT
1 (3)
C35
[(L35)]
C34
[(L34)]
C52
[(L52)]
C30
[(L30)]
TCH
C5
[(L5)]
C31
[(L31)]
C33
[(L33)]
C32
[(L32)]
T1511380-92

SYSTEM DCME_TRANSPARENT DATADEFINITION (3)

```
SYNONYM number_of_ICs Integer = 216;

SYNTYPE ic_range = Natural
  CONSTANTS 1:number_of_ICs
ENDSYNTYPE ic_range;

NEWTYPE ic_access_list
  Array (ic_range, ic_range)
ENDNEWTYPE ic_access_list;
```

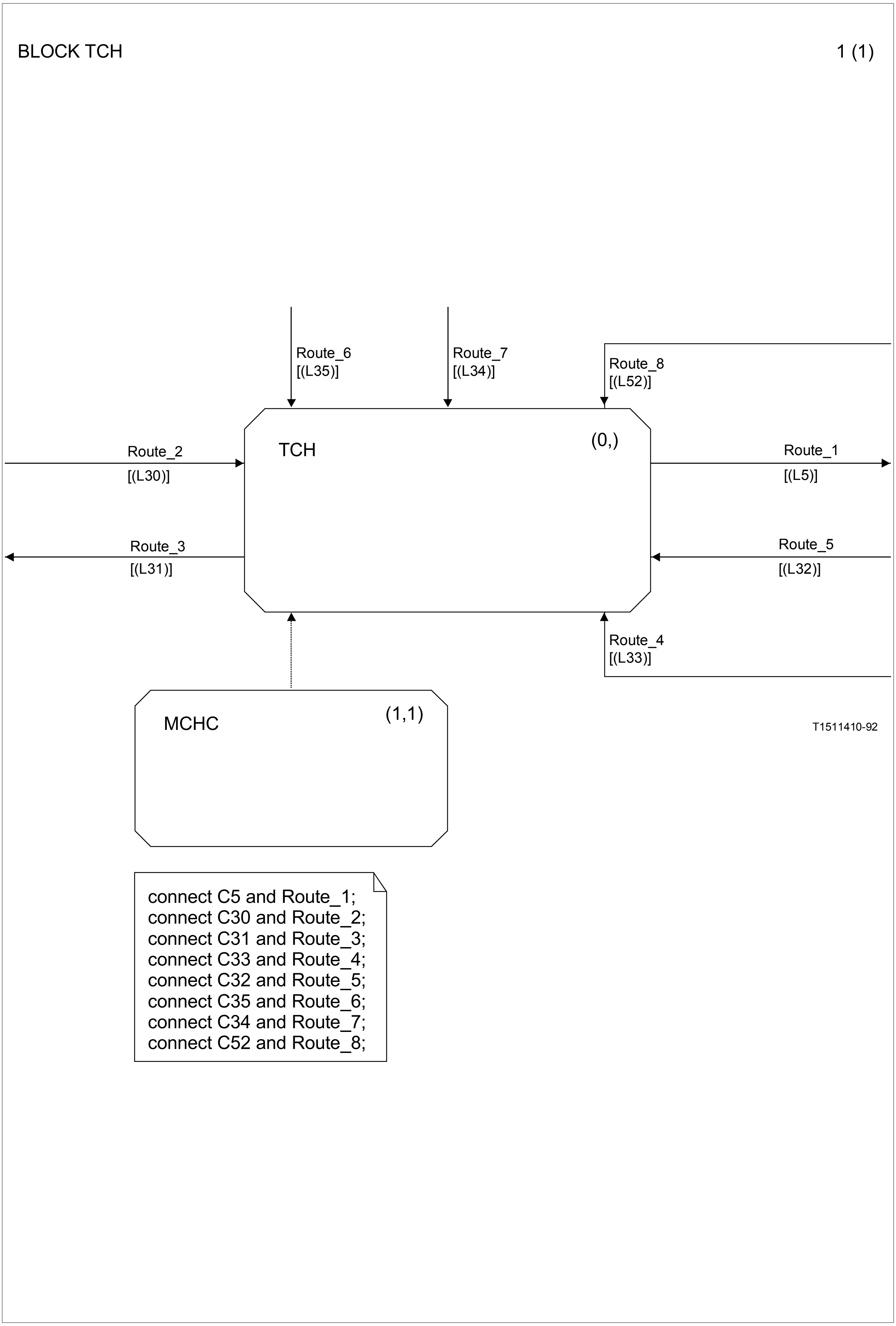
T1511390-92

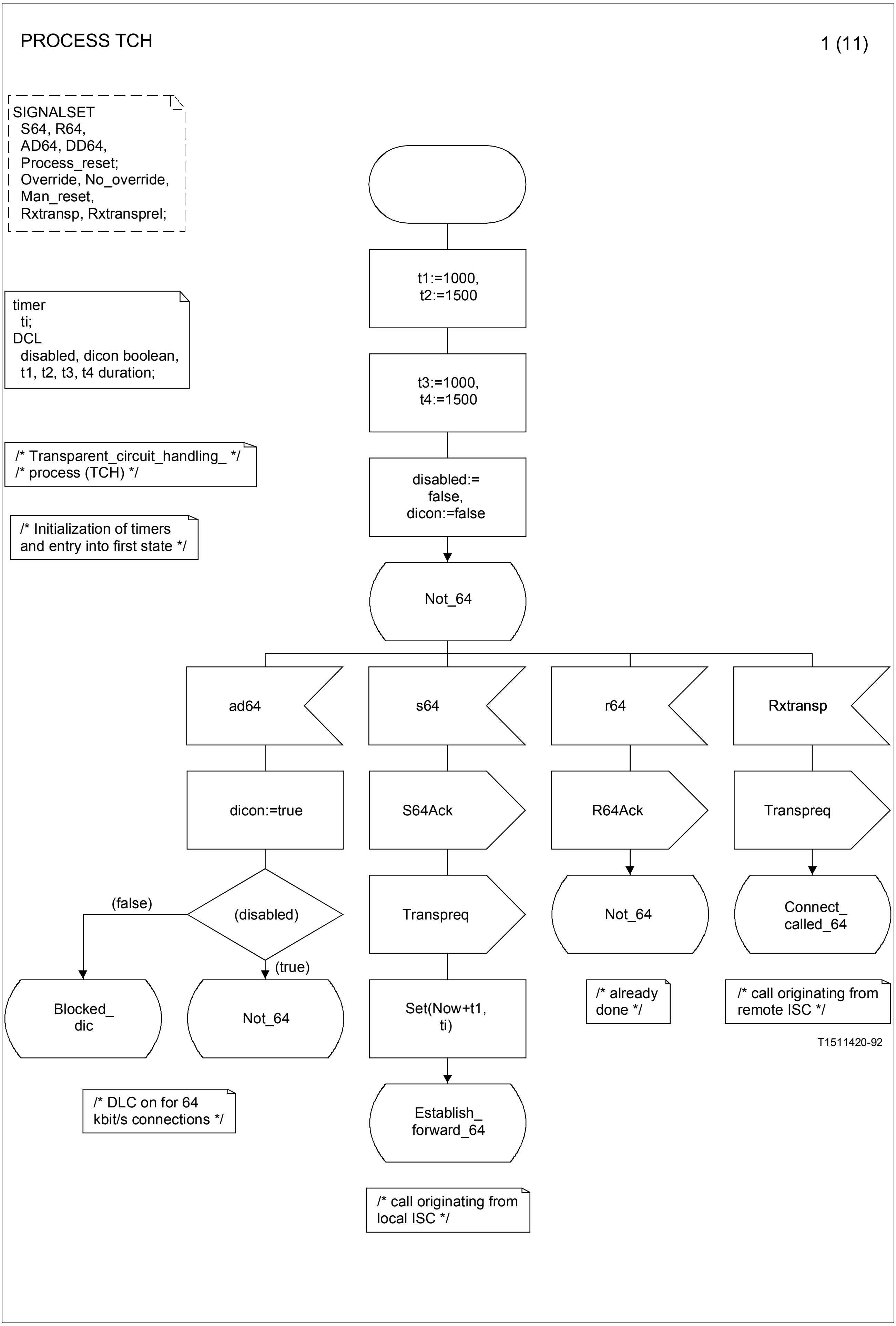
SYSTEM DCME_TRANSPARENT

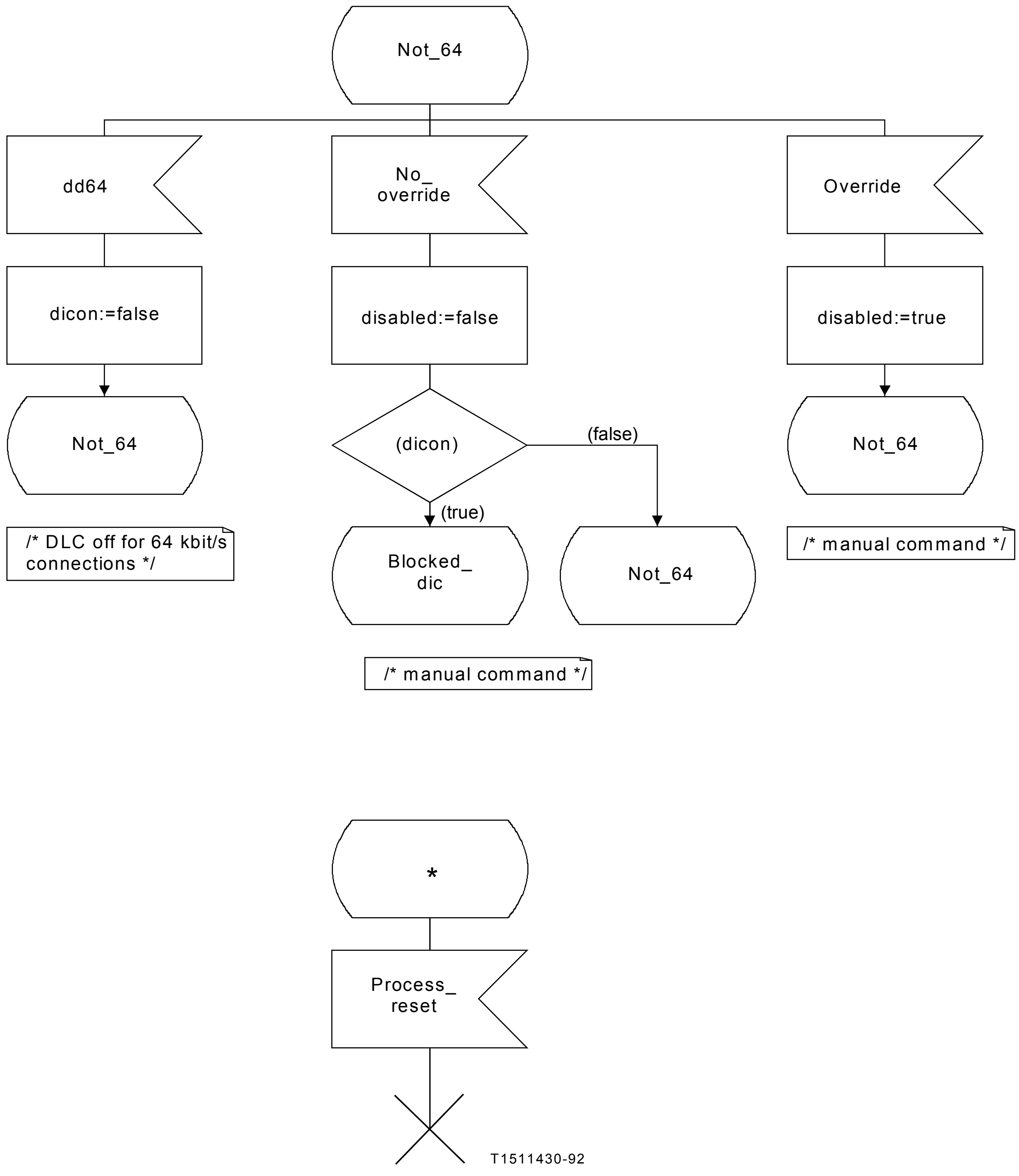
SIGNALDEFINITION (3)

```
/* Signal definitions */
SIGNAL
   Transpreq, Transprel,
   S64, R64,
   S64Ack, S64Nack, R64Ack,
   Out_of_service, Back_in_service,
   AD64, DD64,
   Process_reset,
   Override, No_override,
   Man_reset,
   Rxtransp, Rxtransprel;

/* Signallist definitions */
SIGNALLIST L5 = Transpreq, Transprel;
SIGNALLIST L30 = S64, R64;
SIGNALLIST L31 = S64Ack, S64Nack, R64Ack,
            Out_of_service, Back_in_service;
SIGNALLIST L32 = AD64, DD64;
SIGNALLIST L33 = Process_reset;
SIGNALLIST L34 = Override, No_override;
SIGNALLIST L35 = Man_reset;
SIGNALLIST L52 = Rxtransp, Rxtransprel;
```

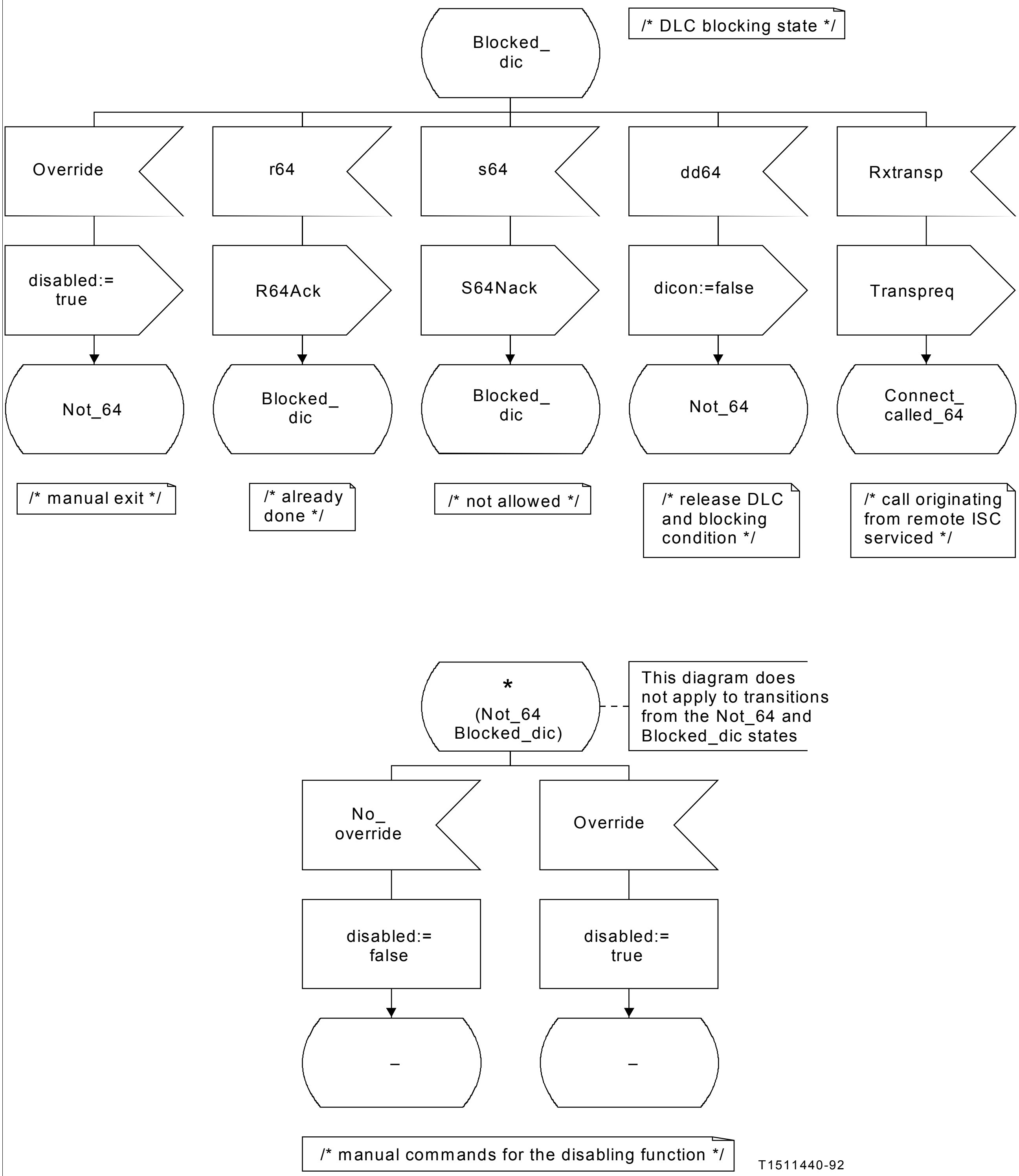
T1511400-92

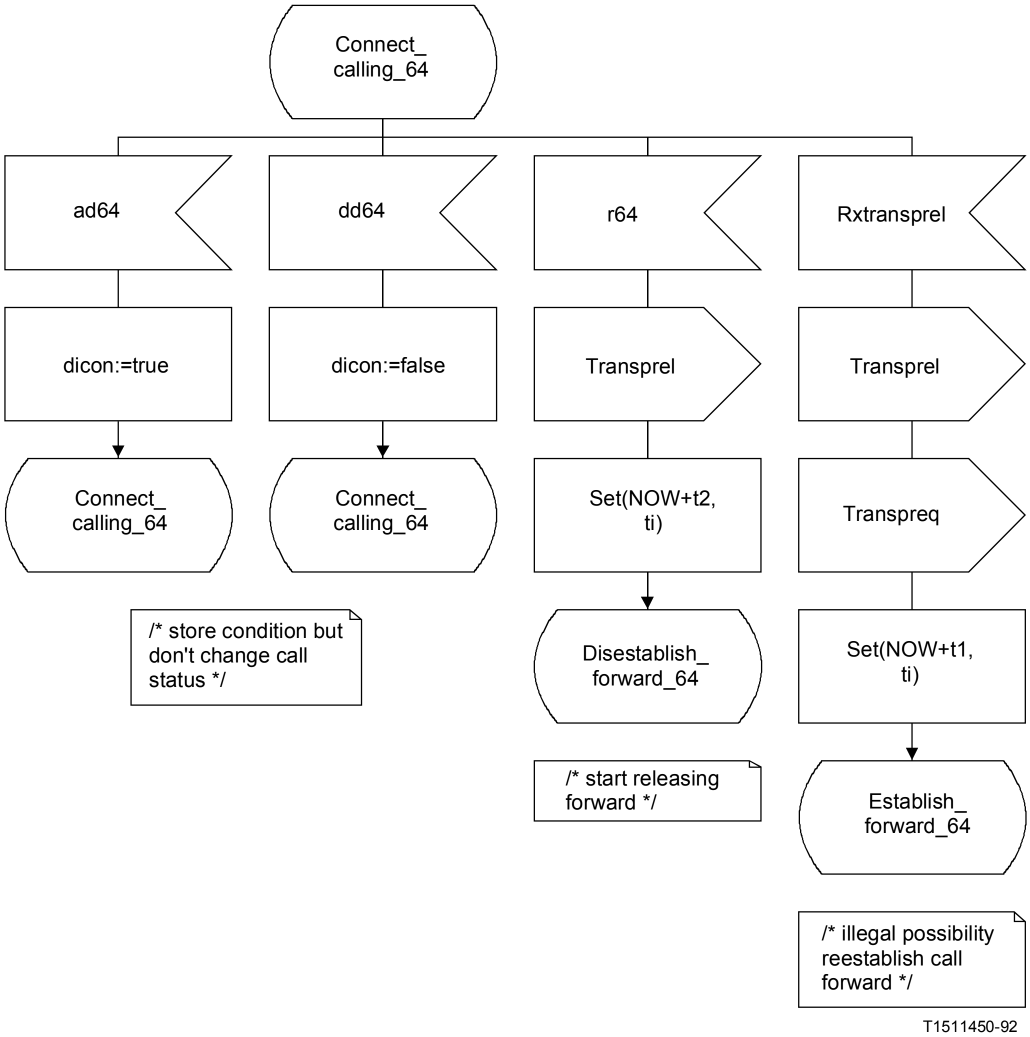

BLOCK TCH
1 (1)
Route_6
[(L35)]
Route_7
[(L34)]
Route_8
[(L52)]
TCH
(0,)
Route_2
[(L30)]
Route_1
[(L5)]
Route_3
[(L31)]
Route_5
[(L32)]
Route_4
[(L33)]
MCHC
(1,1)
T1511410-92
connect C5 and Route_1;
connect C30 and Route_2;
connect C31 and Route_3;
connect C33 and Route_4;
connect C32 and Route_5;
connect C35 and Route_6;
connect C34 and Route_7;
connect C52 and Route_8;

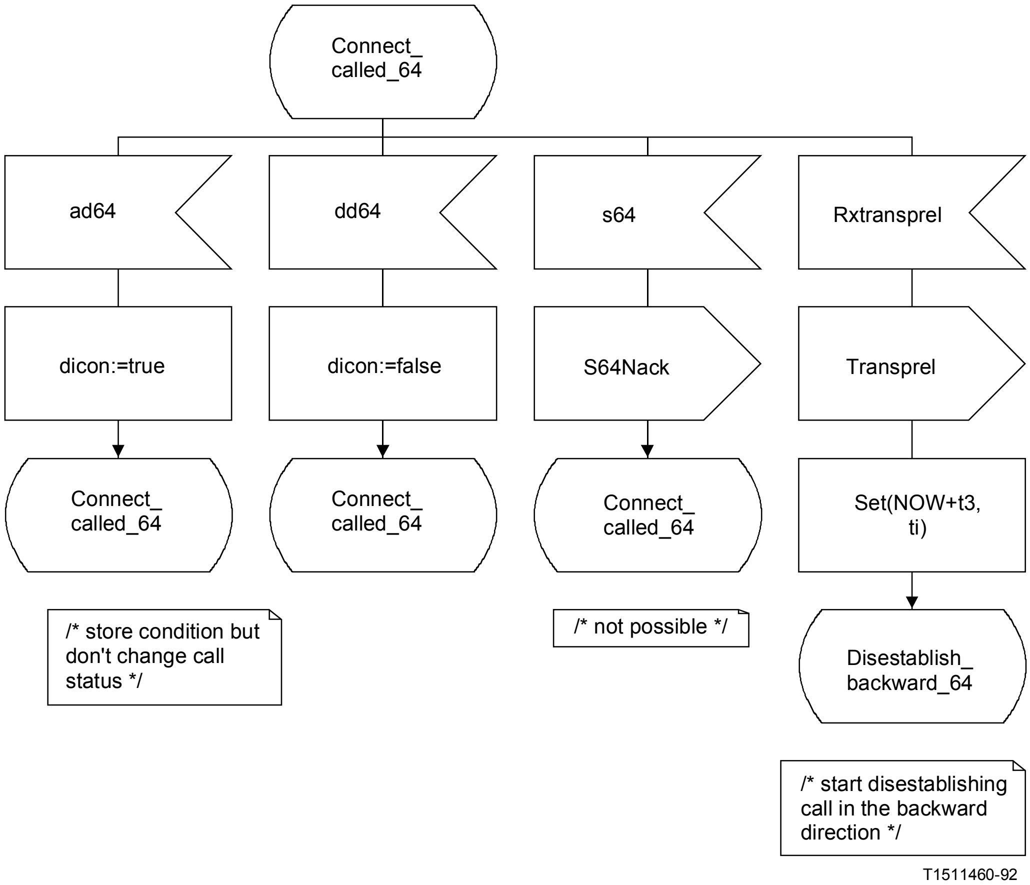

PROCESS TCH
1 (11)
SIGNALSET
S64, R64,
AD64, DD64,
Process_reset;
Override, No_override,
Man_reset,
Rxtransp, Rxtransprel;
timer
ti;
DCL
disabled, dicon boolean,
t1, t2, t3, t4 duration;
/* Transparent_circuit_handling_ */
/* process (TCH) */
/* Initialization of timers
and entry into first state */
t1:=1000,
t2:=1500
t3:=1000,
t4:=1500
disabled:=
false,
dicon:=false
Not_64
ad64
s64
r64
Rxtransp
dicon:=true
S64Ack
R64Ack
Transpreq
(false)
(disabled)
(true)
Transpreq
Not_64
Connect_
called_64
Blocked_
dic
Not_64
Set(Now+t1,
ti)
/* already
done */
/* call originating from
remote ISC */
T1511420-92
/* DLC on for 64
kbit/s connections */
Establish_
forward_64
/* call originating from
local ISC */

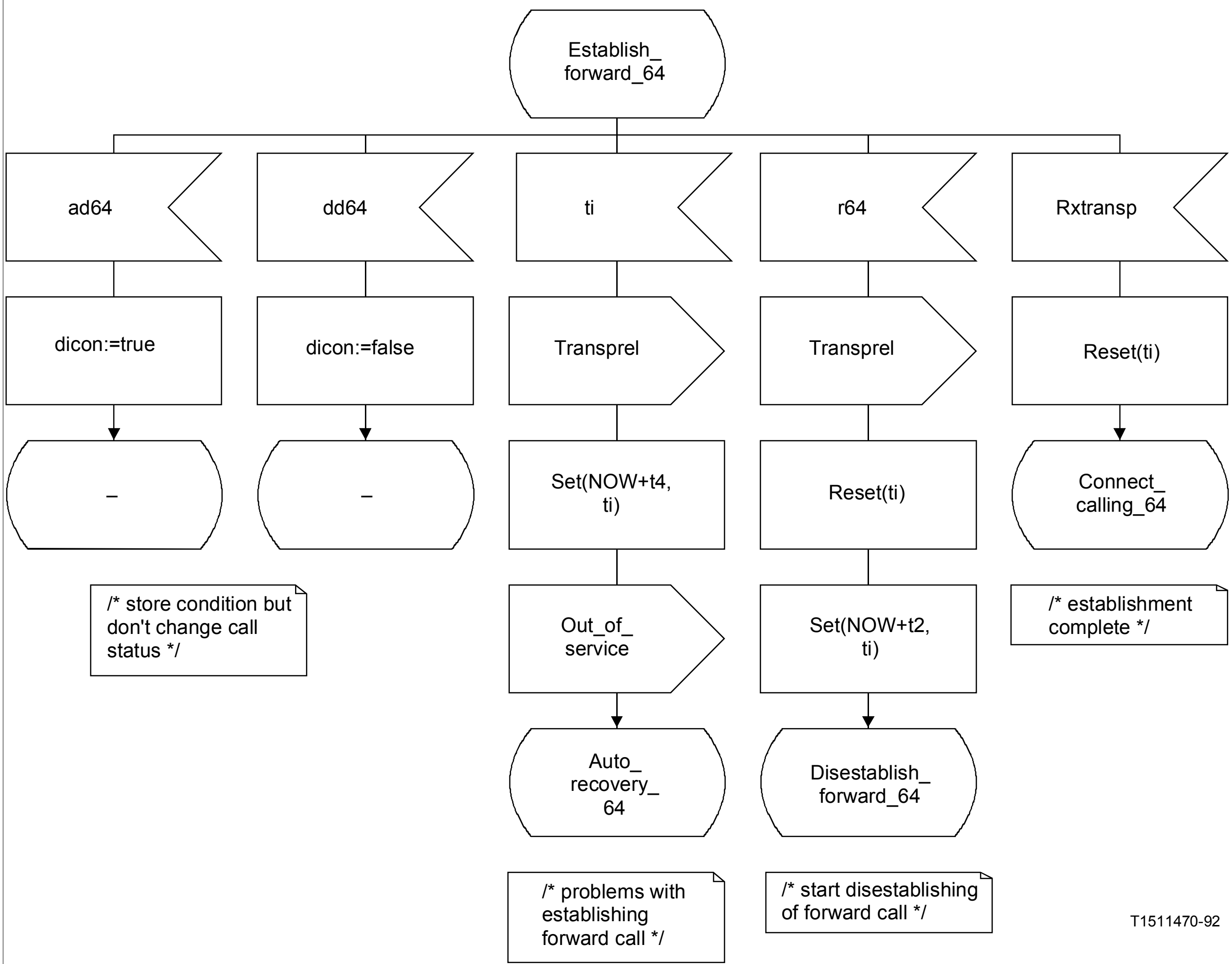


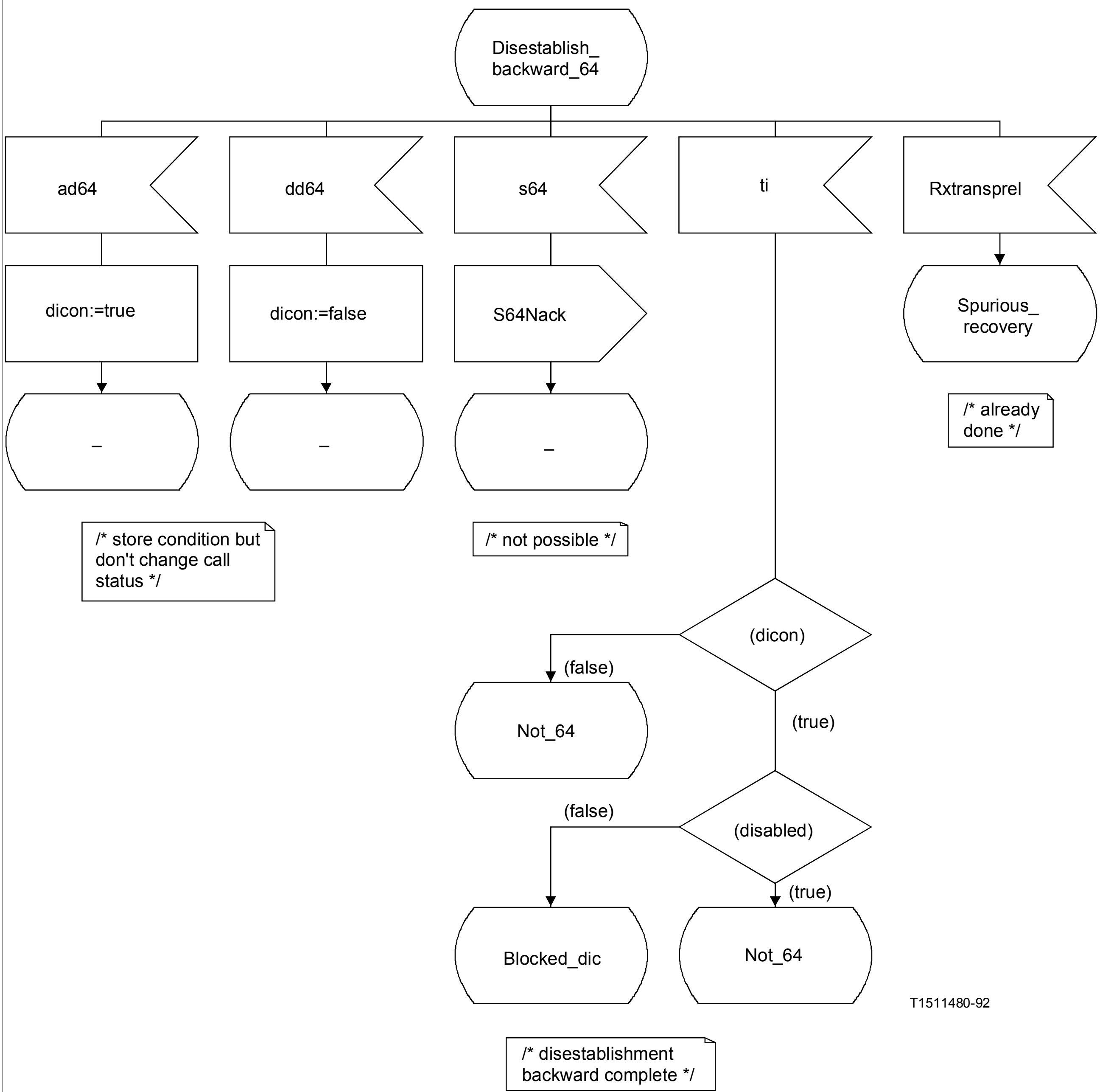
Not_64
dd64
No_
override
Override
dicon:=false
disabled:=false
disabled:=true
Not_64
(dicon)
(false)
Not_64
(true)
/* DLC off for 64 kbit/s
connections */
Blocked_
dic
Not_64
/* manual command */
/* manual command */
*
Process_
reset
T1511430-92

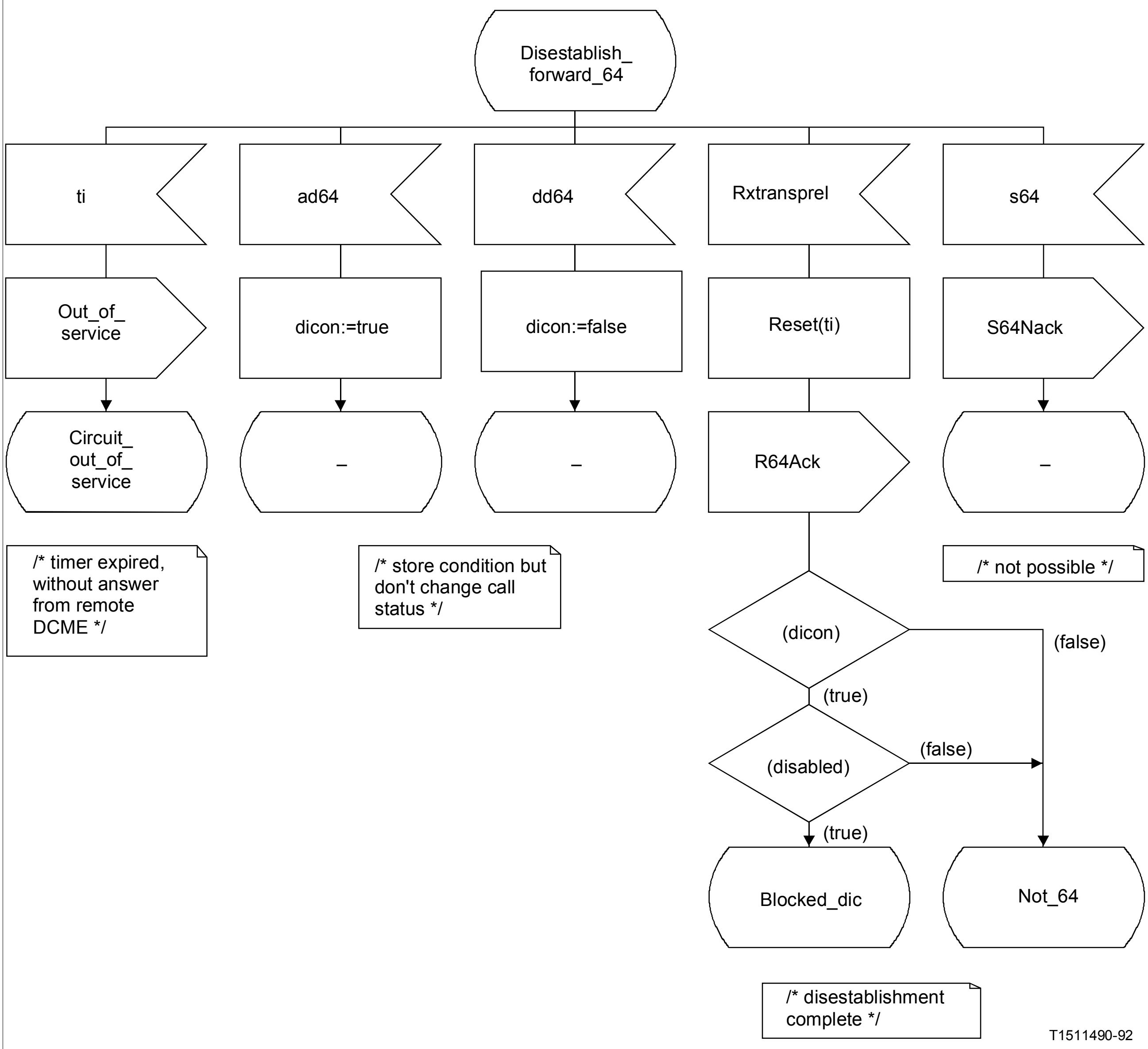

Blocked_
dic
/* DLC blocking state */
Override
r64
s64
dd64
Rxtransp
disabled:=
true
R64Ack
S64Nack
dicon:=false
Transpreq
Not_64
Blocked_
dic
Blocked_
dic
Not_64
Connect_
called_64
/* manual exit */
/* already
done */
/* not allowed */
/* release DLC
and blocking
condition */
/* call originating
from remote ISC
serviced */
*
(Not_64
Blocked_dic)
This diagram does
not apply to transitions
from the Not_64 and
Blocked_dic states
No_
override
Override
disabled:=
false
disabled:=
true
–
–
/* manual commands for the disabling function */
T1511440-92

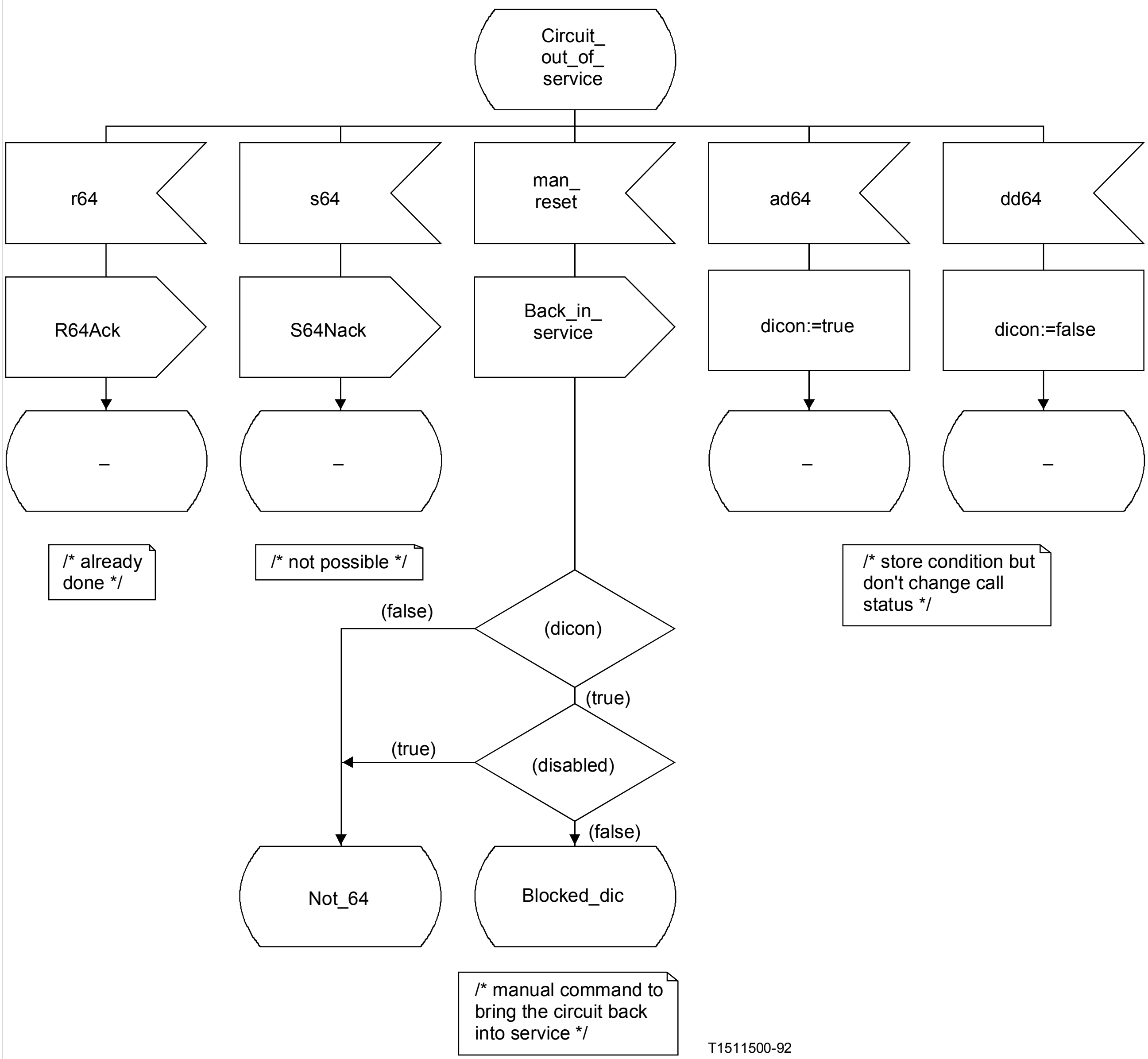

Connect_
calling_64
ad64
dd64
r64
Rxtransprel
dicon:=true
dicon:=false
Transprel
Transprel
Connect_
calling_64
Connect_
calling_64
Set(NOW+t2,
ti)
Transpreq
/* store condition but
don't change call
status */
Disestablish_
forward_64
Set(NOW+t1,
ti)
/* start releasing
forward */
Establish_
forward_64
/* illegal possibility
reestablish call
forward */
T1511450-92

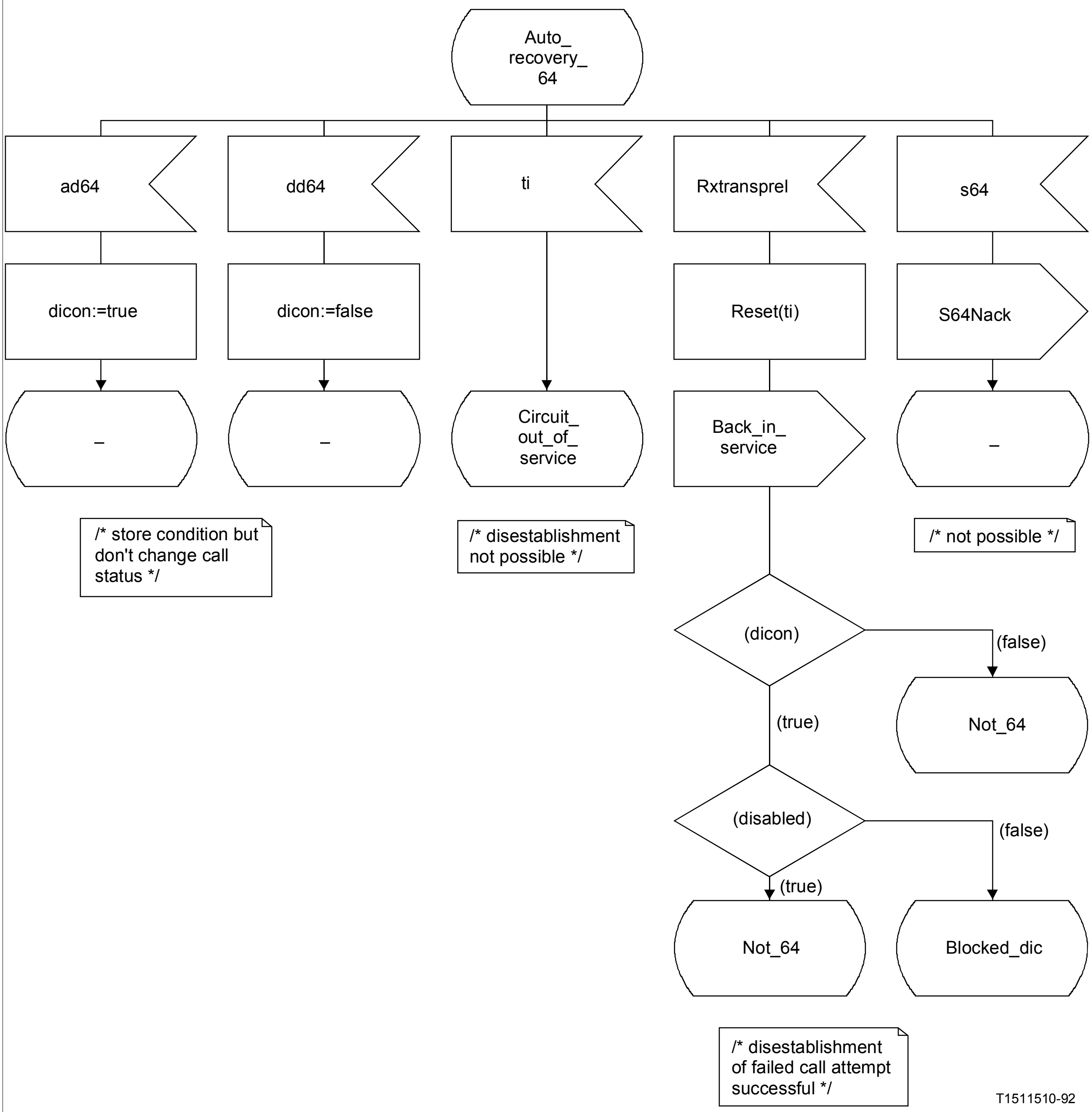

Connect_
called_64
ad64
dd64
s64
Rxtransprel
dicon:=true
dicon:=false
S64Nack
Transprel
Connect_
called_64
Connect_
called_64
Connect_
called_64
Set(NOW+t3,
ti)
/* store condition but
don't change call
status */
/* not possible */
Disestablish_
backward_64
/* start disestablishing
call in the backward
direction */
T1511460-92

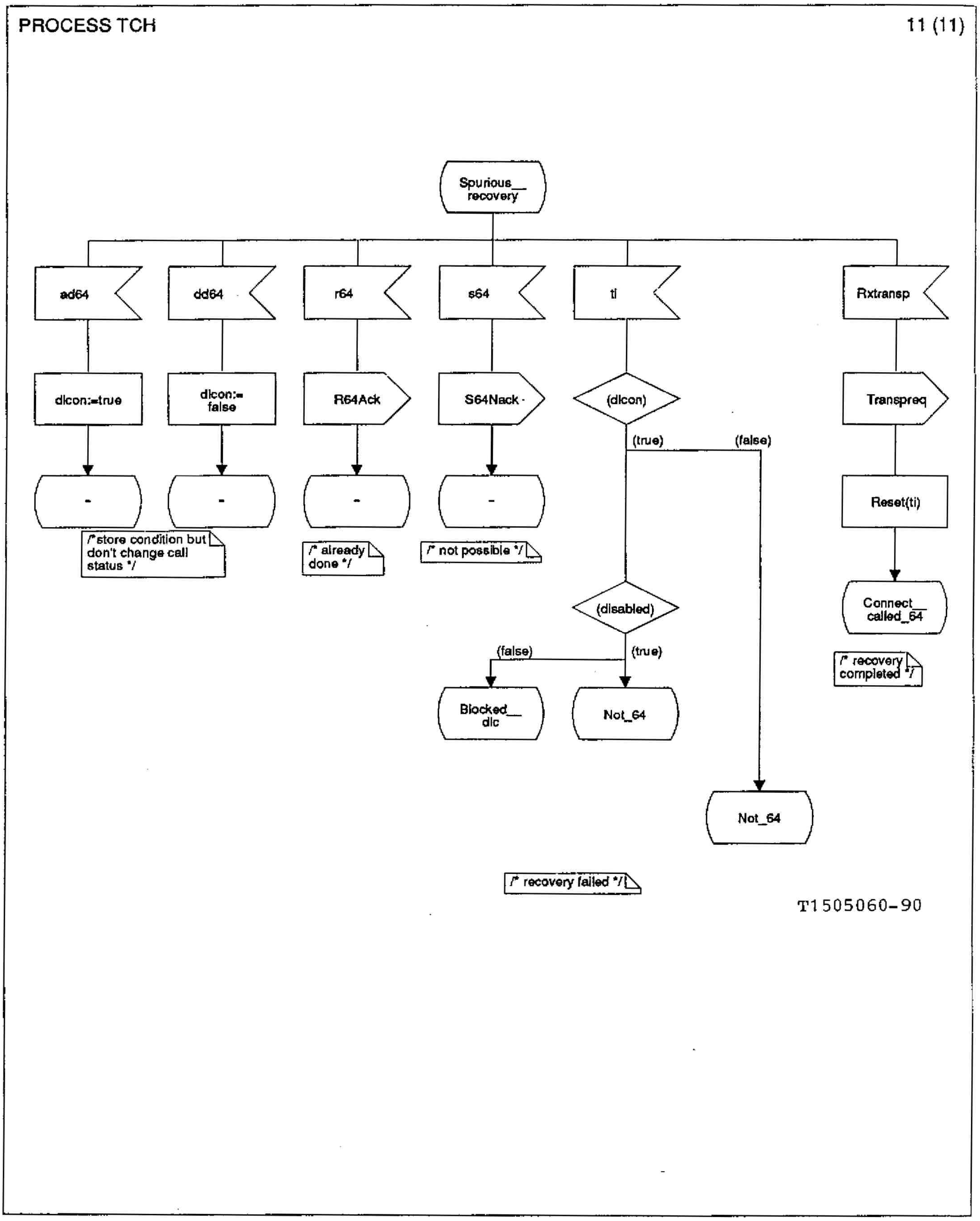

Establish_
forward_64
ad64
dd64
ti
r64
Rxtransp
dicon:=true
dicon:=false
Transprel
Transprel
Reset(ti)
–
–
Set(NOW+t4,
ti)
Reset(ti)
Connect_
calling_64
/* store condition but
don't change call
status */
Out_of_
service
Set(NOW+t2,
ti)
/* establishment
complete */
Auto_
recovery_
64
Disestablish_
forward_64
/* problems with
establishing
forward call */
/* start disestablishing
of forward call */
T1511470-92


Disestablish_
backward_64
ad64
dd64
s64
ti
Rxtransprel
dicon:=true
dicon:=false
S64Nack
Spurious_
recovery
/* already
done */
–
–
–
/* store condition but
don't change call
status */
/* not possible */
(dicon)
(false)
Not_64
(true)
(false)
(disabled)
(true)
Blocked_dic
Not_64
T1511480-92
/* disestablishment
backward complete */

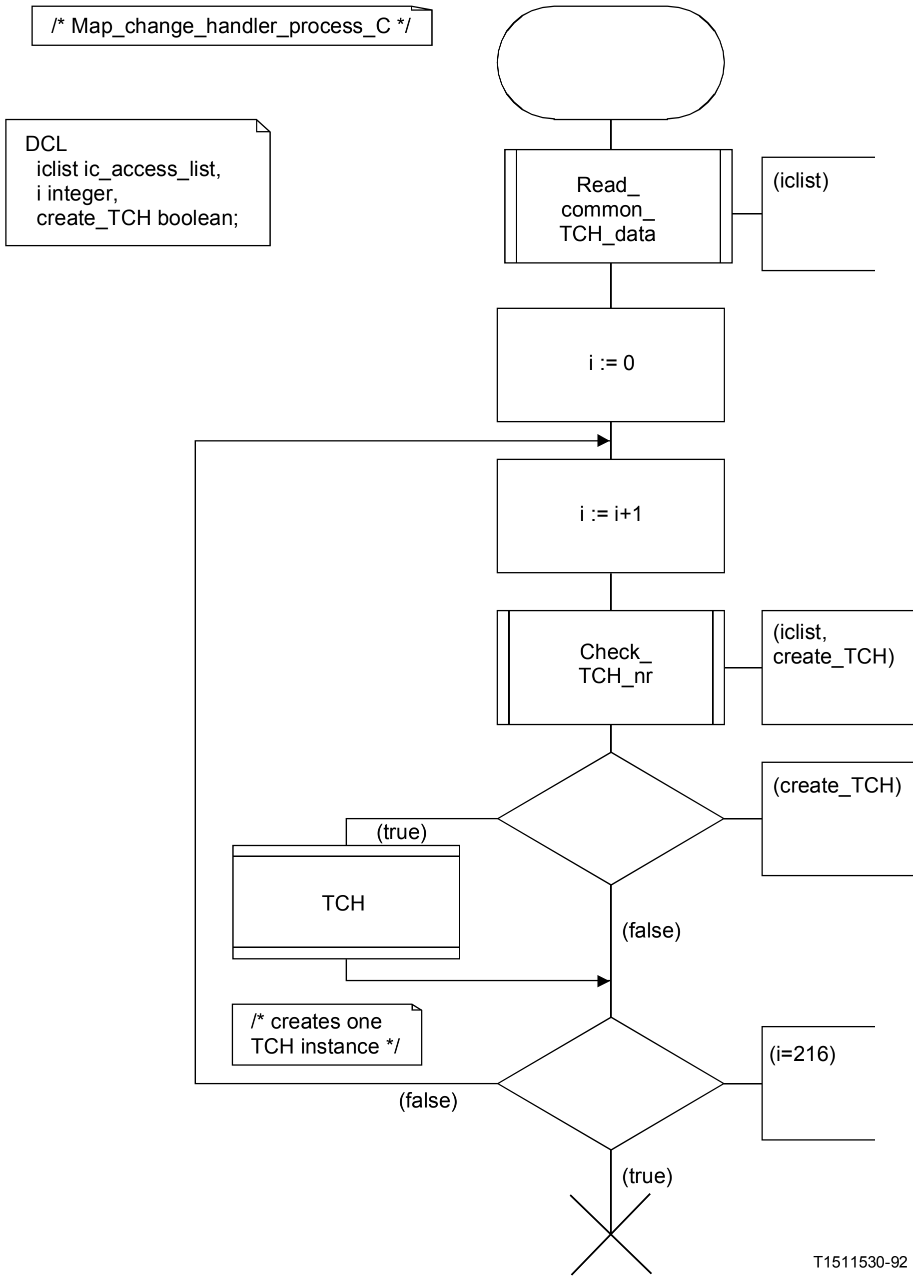

Disestablish_
forward_64
ti
ad64
dd64
Rxtransprel
s64
Out_of_
service
dicon:=true
dicon:=false
Reset(ti)
S64Nack
Circuit_
out_of_
service
–
–
R64Ack
–
/* timer expired,
without answer
from remote
DCME */
/* store condition but
don't change call
status */
/* not possible */
(dicon)
(false)
(true)
(disabled)
(false)
(true)
Blocked_dic
Not_64
/* disestablishment
complete */
T1511490-92


Circuit_
out_of_
service
r64
s64
man_
reset
ad64
dd64
R64Ack
S64Nack
Back_in_
service
dicon:=true
dicon:=false
–
–
–
–
/* already
done */
/* not possible */
/* store condition but
don't change call
status */
(false)
(dicon)
(true)
(true)
(disabled)
(false)
Not_64
Blocked_dic
/* manual command to
bring the circuit back
into service */
T1511500-92


Auto_
recovery_
64
ad64
dd64
ti
Rxtransprel
s64
dicon:=true
dicon:=false
Reset(ti)
S64Nack
–
–
Circuit_
out_of_
service
Back_in_
service
–
/* store condition but don't change call status */
/* disestablishment not possible */
/* not possible */
(dicon)
(false)
(true)
Not_64
(disabled)
(false)
(true)
Not_64
Blocked_dic
/* disestablishment of failed call attempt successful */
T1511510-92



PROCESS TCH
11 (11)
Spurious_ recovery
ad64
dd64
r64
s64
ti
Rxtransp
dlcon:=true
dlcon:= false
R64Ack
S64Nack
(dlcon)
Transpreq
(true)
(false)
-
-
-
-
Reset(ti)
/*store condition but don't change call status */
/* already done */
/* not possible */
(disabled)
Connect_ called_64
/* recovery completed */
(false)
(true)
Blocked_ dlc
Not_64
Not_64
/* recovery failed */
T1505060-90


/* Map_change_handler_process_C */
DCL
iclist ic_access_list,
i integer,
create_TCH boolean;
Read_
common_
TCH_data
(iclist)
i := 0
i := i+1
Check_
TCH_nr
(iclist,
create_TCH)
(create_TCH)
(true)
TCH
(false)
/* creates one
TCH instance */
(i=216)
(false)
(true)
T1511530-92

ANNEX B

(to Recommendation G.763)

B.1 *An example of a DLC double averaging technique*

The average number of encoding bits per sample is obtained using a double average process.

a) The first stage averaging is computed at discrete time instances once every n DCME frames, where n is operator selectable (n = 4, 16, 32, 64 or 128). The result of the computation is the ensemble average <Se> taken over the ensemble of BCs which are carrying voice traffic and will result in one of the following possible outcomes:

– <Se> = 4 for N ≤ M

– <Se> = 4M/N for 3N/4 ≤ M < N

with M = total number of 4 bit bearer time slots in the pool which are not used for voice band data, bit banks and 64 kbit/s on-demand traffic counted in the measurement frame.

N = total number of connected active voice ITs, counted in the measurement frame.

Two ensemble averages should be determined:

– <Sea> – which is the *actual* measured ensemble average of encoding bits/sample <Se> based on actual counts of M and N.

– <Sep> – which is the *predicted* ensemble average of encoding bits/sample <Se> based on the actual count of N and a reduced count of M–2.

b) The second stage averaging should be a moving discrete time averaging of <Sea> and <Sep>:

– Sta – which is the moving discrete time average of 100 consecutive values of <Sea>.

– Stp – which is the moving discrete time average of 100 consecutive values of <Sep>.

The value of Sta may be used as a measure of the average number of encoding bits/sample when determining the dynamic load control condition for voice and voice band data channels.

The value of Stp may be used as a measure of the average number of encoding bits/sample when determining the dynamic load control condition for on-demand 64 kbit/s channels.

B.2 *Transmit activity detector threshold and operate time characteristic*

A typical response to a sinusoidal stimulus signal in the band 300 to 3400 Hz will be as given below:

Average signal power (see Note)	*Operate time*
< –40 dBm0	OFF
≥ –40 dBm0, ≤ –30 dBm0	Figure B-1/G.763
> –30 dBm0	2 ms < t < 4 ms

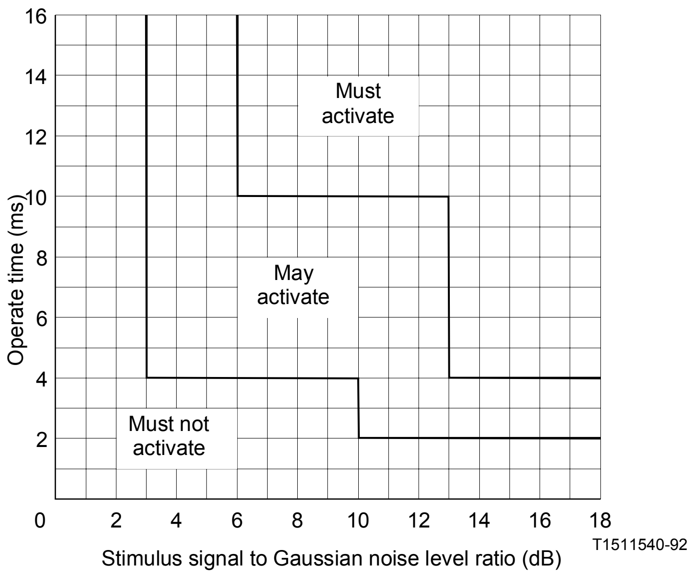
The operate time requirements will be satisfied while permitting tolerances on the average signal power of any stimulus signal in the frequency band at boundary conditions as follows:

–40 dBm0 ± 1.5 dB

–30 dBm0 ± 1.0 dB

A typical rate of change of the transmit activity detector adaptive threshold will be between 2.5 dB/s and 20.0 dB/s.

Note – The activity detector should not indicate activity for idle channel noise less than –40 dBm0.



Note – Mask applicable to stimulus signals ≥ –40 dBm0,
≤ –30 dBm0.

Stimulus signal to be 1020 Hz sinusoid.

FIGURE B-1/G.763

Transmit activity detection operate threshold mask

B.3 *Data/speech discriminator*

Functionally, the data/speech (D/S) discriminator determines whether the activity on each transmit IT is speech or data and provides a speech/data indication to the hangover control and signal classification process.

The implementation of the D/S discriminator may be performed by a combination of spectral analysis and 2100 Hz tone detection.

The following requirements should be met with the modem types and bit rates given in Table 7/G.763.

B.3.1 *Output conditions*

The D/S discriminator analyzes the activity on each transmit IT and provides the following output conditions:

Activity	*Output condition*
Speech	"Voice"
Tones except 2100 Hz	"Voice"
Data signal (see Note)	"Data"
2100 Hz	"Data"

Note – V.23 modem signals may be classified as either voice or data dependent upon the design of the data/speech discriminator.

The D/S discriminator provides a continuous output condition indicating the presence of either speech or data on the ITs. The current output condition should be maintained upon termination of activity on the IT or until the output condition of a subsequent activity is determined. Upon system start-up or map change, the D/S discriminator should be reset to "Voice".

B.3.2 *Accuracy*

The missed detection probability of data as speech or speech as data should be less than 0.5%.

B.3.3 *Response time*

The D/S discriminator should update its output condition within 200 ms after any of the following changes in the signal characteristics on the IT:

– Inactive-to-speech

– Inactive-to-data

– Speech-to-data

– Data-to-speech

B.3.4 *2100 Hz tone detection*

The 2100 Hz tone detector should meet the following requirements:

– Frequency range of tone: 2100 ± 21 Hz

– Minimum amplitude of tone: –25 dBm0

– Response time: < 100 ms (for further study)

B.4 *2400 Hz tone detector*

The 2400 Hz tone detector should meet the following requirements:

– Frequency of tone: 2400 Hz ± 15 Hz

– Minimum amplitude of tone: –25 dBm0

– Response time: < 50 ms

– Missed detection probability: < 0.5%.

B.5 *Speech detector/echo control device interactions*

Consideration must be given to minimizing excessive channel loading which may exist as the result of network interactions between the DCME speech detector and an echo control device (see Figure B-2/G.763).

If the DCME utilizes an adaptive threshold speech detector, interaction between the speech detector threshold adjustment and the echo control operation may generate excessive activity in the channel. The echo control device modulates the terrestrial circuit noise accumulated between the telephone and the send-input port of the echo control device. The adaptive threshold speech detector may falsely classify this change in terrestrial circuit noise as speech and increase the load on the DCME. This will increase the occurrence of overload channels and/or freeze-out, thereby degrading the performance in the baseband channel. This interaction occurs as follows:

a) Receive speech arrives at the receive input (Rin) port of the echo control unit.

b) The echo suppression switch or canceller centre clipper activates, stopping the echo or residual echo and attenuating the near-end-generated analogue terrestrial noise (N1) present at the send input (Sin) port.

c) If very little noise is generated between the echo control send output (Sout) port and the DCME speech detector input, the speech detector threshold will adapt to its minimum level (typically −50 dBm0).

d) When the receive speech stops, after a suitable echo control unit handover time the echo suppression switch or canceller centre clipper will close and the near-end-generated terrestrial noise, as seen by the DCME speech detector will reappear as a step change in noise level.

e) This step change in noise level may exceed the speech detector threshold, causing the DCME to transmit a noise spurt as if it were speech. The noise spurt duration will be a function of the adaptation speed of the speech detector and the near-end-generated terrestrial noise level.

This sequence will be repeated for every speech spurt and will produce a very annoying speech-correlated noise spurt heard by the far-end talkers every time they stop speaking.

This interaction is not limited to single echo control device network configurations. A typical network configuration with multiple echo control devices interacting with a DCME speech detector is shown in Figure B-3/G.763. In this configuration, the DCME speech detector may respond to unit step increases in noise power which result from echo suppressor switch or echo canceller centre clipper activations in the send paths of echo control devices 1 and 3. The DCME speech detector will first experience a unit step increase in noise power from echo control device 3 switch activation, followed by a second step increase from echo control device 1 switch activation. The extent to which the DCME speech detector incorrectly responds to these step increases in noise power will be a function of the noise power levels N_1, N_2, N_3, and N_4, and the specific DCME speech detector threshold adaptation algorithm. For example, the dual step increases in noise presented to the DCME speech detector which result from switch or centre clipper activation at locations 1 and 3 will be masked if the power level of N_4 is excessively high. Likewise, high noise power levels at N_2 or N_3 may mask step increases in noise power caused by echo control unit 1.

There are several methods for dealing with the interactions between the echo control devices and the DCME speech detector. In one approach, the echo control device could be modified to monitor the terrestrial-generated noise at the send-input port. When the send transmission path is broken, noise at the proper level is injected into the send-output toward the DCME, keeping the noise seen by the speech detector at a constant level and avoiding speech detector activation. This approach is unacceptable due to the number of different echo control devices in use and the uniqueness of the application. In a second approach, the speech detector adaptive threshold would be frozen in the presence of speech on the corresponding receive channel. A third approach would be to specify an adaptive speech detector with a fast adaptation feature which would track step changes in noise level and minimize the noise spurts.

The transmit activity detector threshold should not adapt to Gaussian noise level variations which are due to the action of echo suppressors or echo cancellers. This may be accomplished by any means which is functionally equivalent to providing the transmit activity detector with a threshold inhibit signal from a receive activity detector when activity is present on the receive channel (see § 12.4).

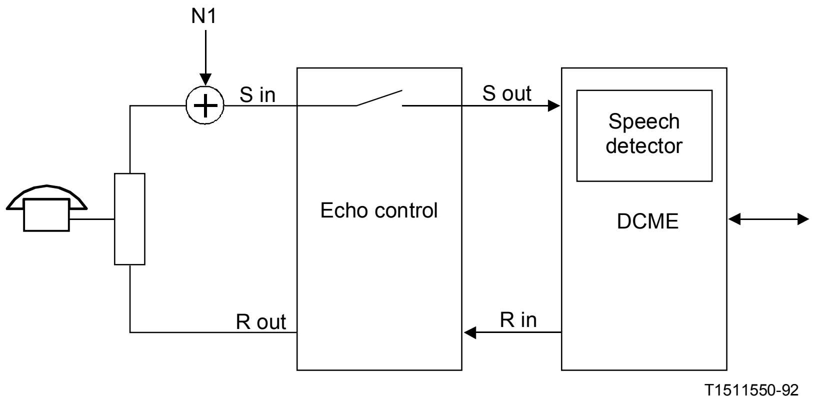


FIGURE B-2/G.763

Speech detector/echo control device interactions

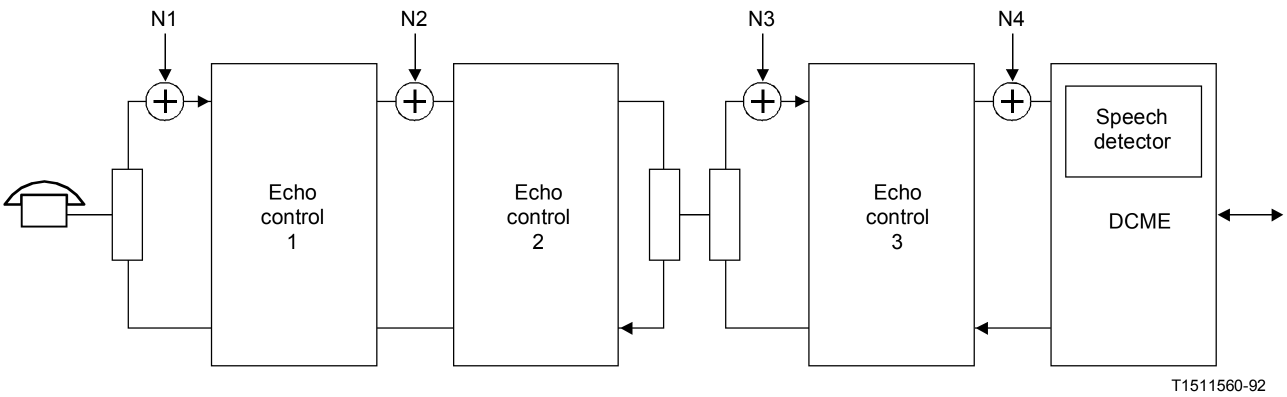


FIGURE B-3/G.763

Multiple echo control device network configuration

B.6 *Timing synchronization*

The following figures provide a number of examples of Doppler and plesiochronous slip buffer placements for a variety of network synchronization schemes. In the figures it is assumed that all buffers will derive their write clocks from the input bit stream.

B.6.1 *Point-to-point operation*

B.6.1.1 *Terrestrial operation within a national network*

Figures B-4/G.763 and B-5/G.763 show methods of DCME terminal synchronization for operation within a national network.

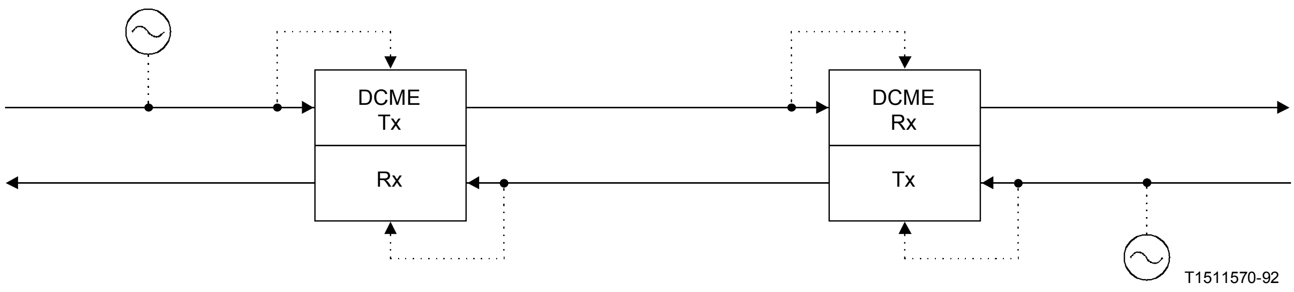


FIGURE B-4/G.763

DCME synchronous (independent) operation (in a single asynchronous network)

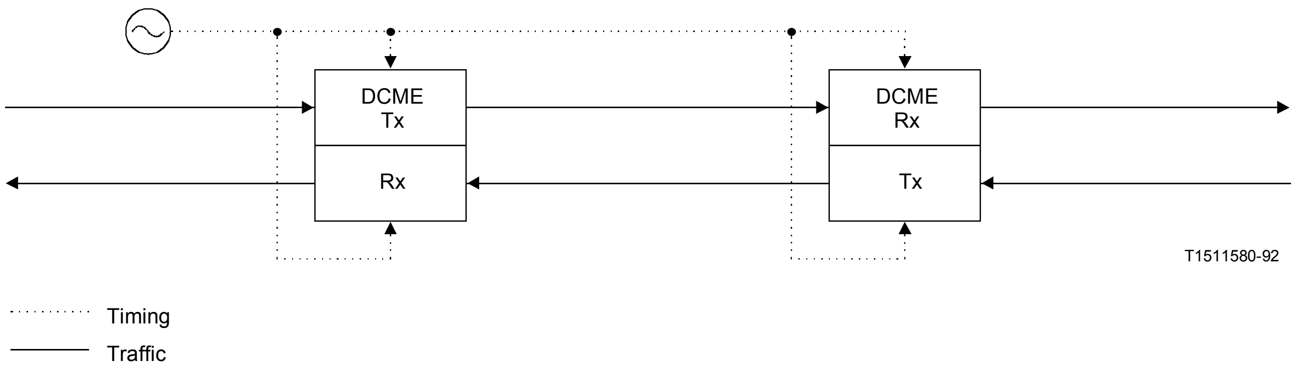


FIGURE B-5/G.763

DCME synchronous operation (in a single asynchronous network)

B.6.1.2 *Terrestrial operation between national networks*

Figures B-6/G.763, B-7/G.763 and B-8/G.763 show methods of terminal synchronization for operation between national networks via terrestrial networks. Plesiochronous buffers are required for networks as shown in Figures B-6/G.763 and B-7/G.763. Figure B-8/G.763 utilizes loop timing and therefore does not require plesiochronous buffering.

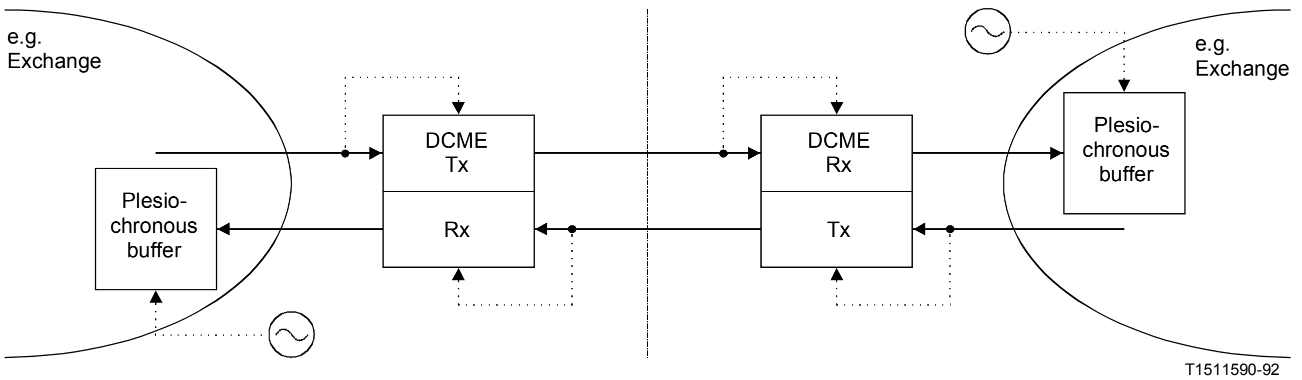


FIGURE B-6/G.763

DCME synchronous (independent) operation (between two plesiochronous networks)

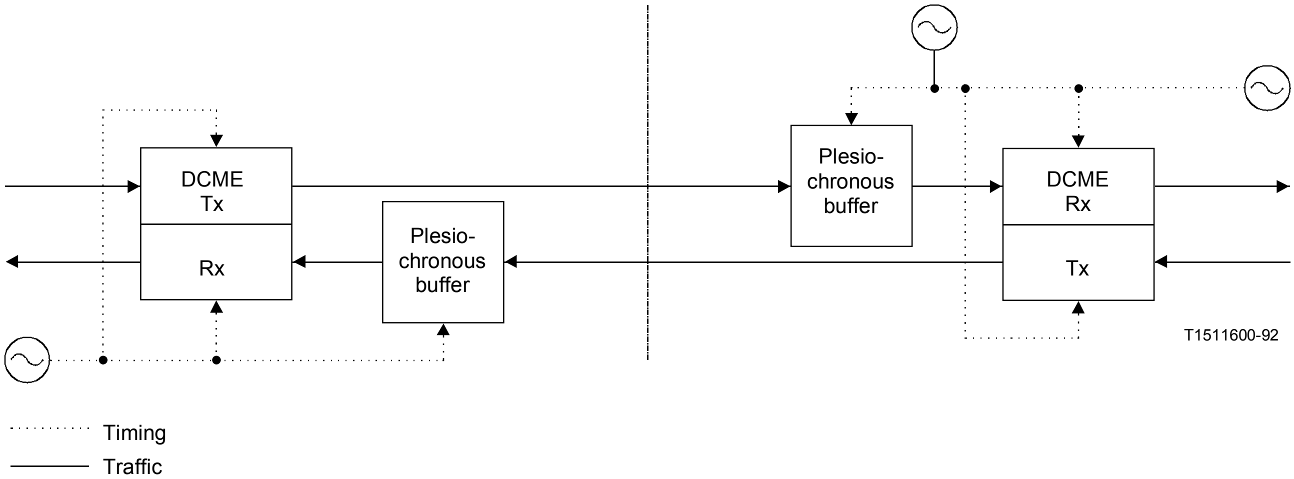


FIGURE B-7/G.763

DCME buffered plesiochronous operation
(between two plesiochronous networks)

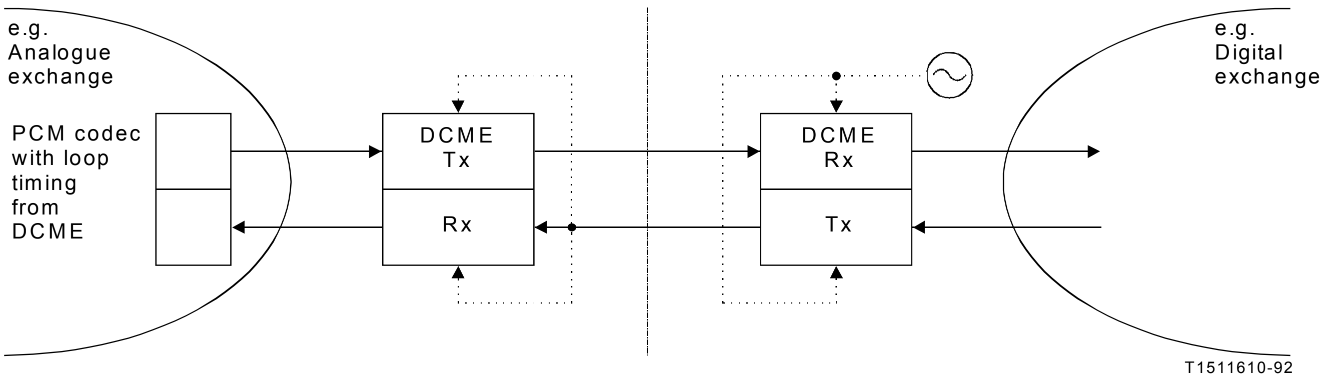


FIGURE B-8/G.763

DCME synchronous loop operation
(between analogue and digital networks)

B.6.1.3 *Satellite operation between national networks based upon continuous digital carrier type services*

Figures B-9/G.763, B-10/G.763, B-11/G.763 and B-12/G.763 show methods of terminal synchronization for operation between national networks over a satellite link based upon asynchronous continuous digital carrier type services. Figure B-9/G.763 introduces controlled slips between the DCMEs which are limited to 1 in 70 days if G.811 clocks are available in both networks. The configuration shown in Figures B-10/G.763, B-11/G.763 and B-12/G.763 permit slip free operation between the DCMEs.

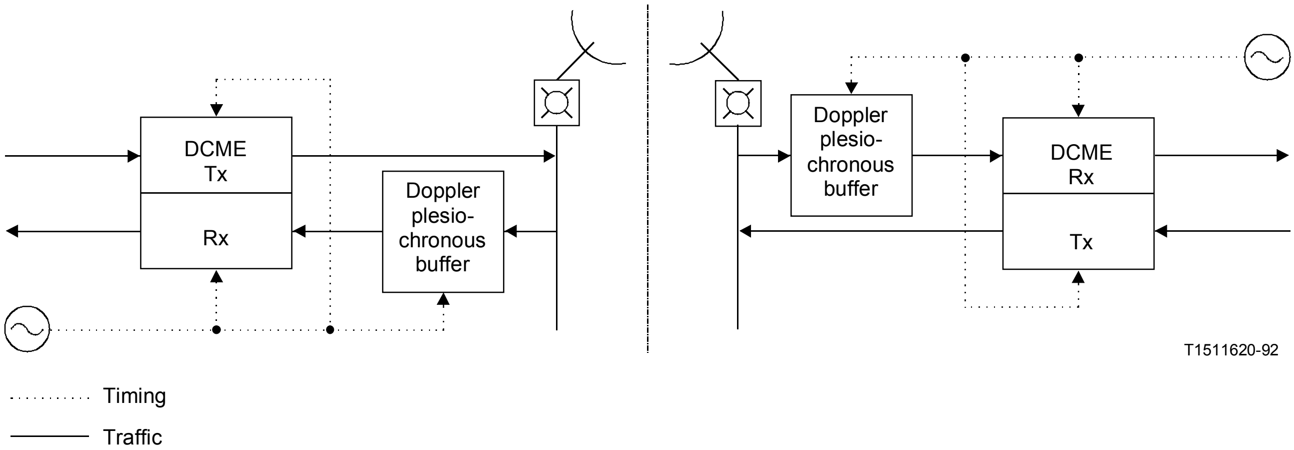


FIGURE B-9/G.763

DCME buffered plesiochronous operation
(between two plesiochronous networks)

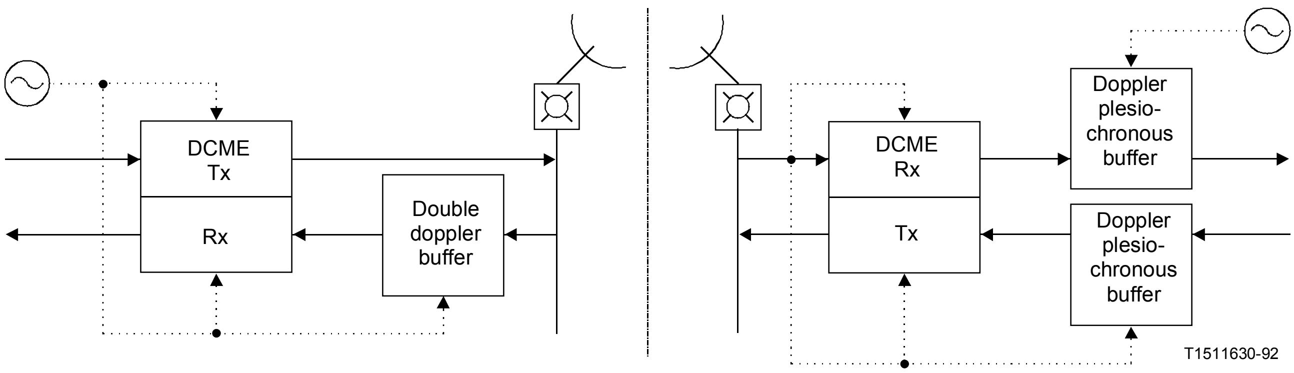


FIGURE B-10/G.763

DCME synchronous loop operation
(between two plesiochronous networks)

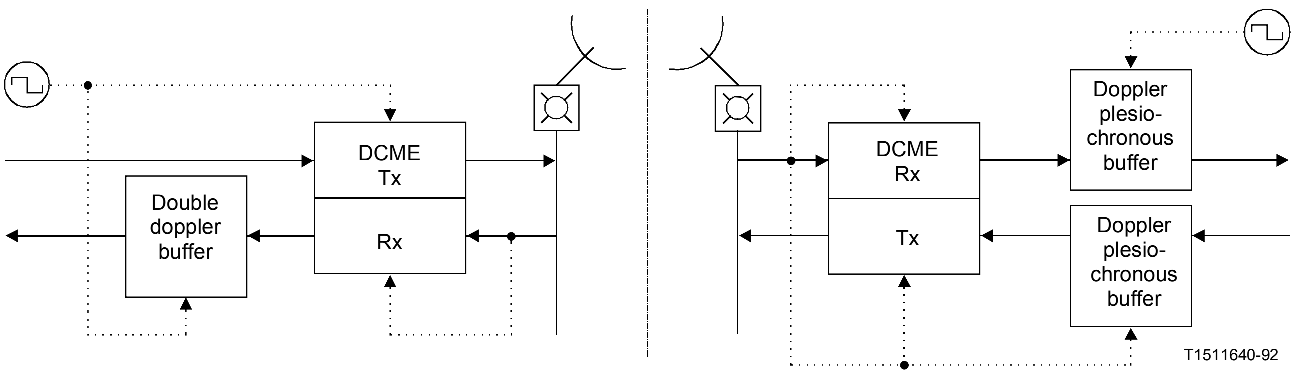


FIGURE B-11/G.763

DCME synchronous loop operation
(between two plesiochronous networks)

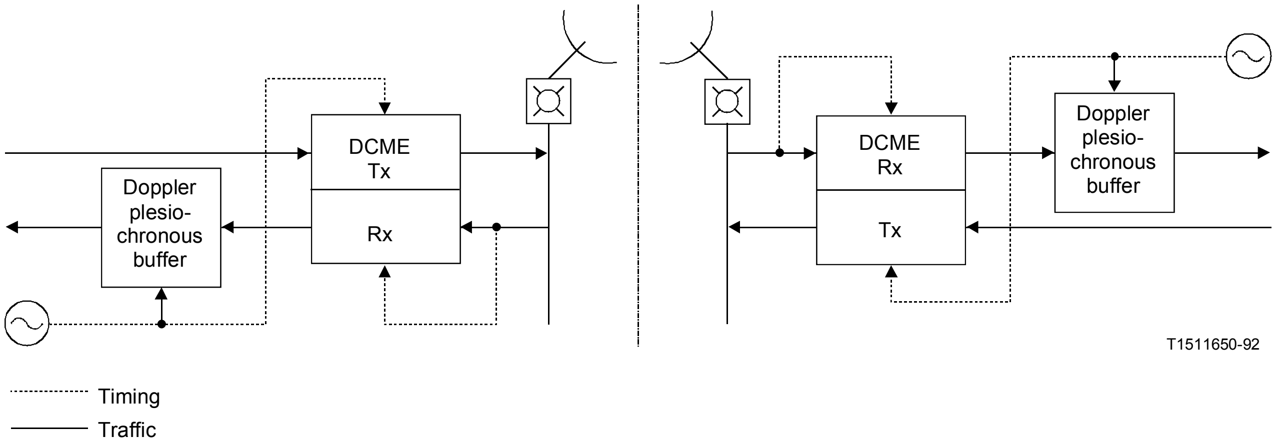


FIGURE B-12/G.763

DCME synchronous (independent) operation (between two plesiochronous networks)

B.6.1.4 *Satellite operation between national networks based upon TDMA type services*

Figures B-13/G.763 and B-14/G.763 show a method of DCME terminal synchronization for operation between national networks over a satellite link based on TDMA-type services. An appropriate interface is provided in the TDMA terminal to permit interfacing the DCME with and without multi-clique over a primary multiplex port. The configuration shown in Figure B-13/G.763 permits slip free operation between the DCMEs.

B.6.2 *Multi-clique operation*

B.6.2.1 *Terrestrial operation within a national network*

Figure B-15/G.763 shows a method of DCME terminal synchronization for operation within a national network. The cross connect function provides a means of assembling the received multi-clique pools on to a single primary multiplex.

B.6.2.2 *Terrestrial operation between national networks*

Figure B-16/G.763 shows a method of DCME terminal synchronization for operation between national networks via terrestrial facilities. Plesiochronous buffers are required to resolve timing differences between the various plesiochronous networks. Due to the multiple source nature of the multi-clique configuration, the plesiochronous buffers must be placed before the cross connect function.

B.6.2.3 *Satellite operation between national networks based upon continuous carrier type services*

Figure B-17/G.763 shows a method of DCME terminal synchronization for operation between national networks based on continuous digital satellite carriers. Plesiochronous/doppler buffers are required to resolve timing differences between the various plesiochronous networks and to remove satellite induced doppler shifts on the received data streams. Due to the multiple source nature of the multi-clique configuration, the plesiochronous/doppler buffers must be placed before the cross connect function.

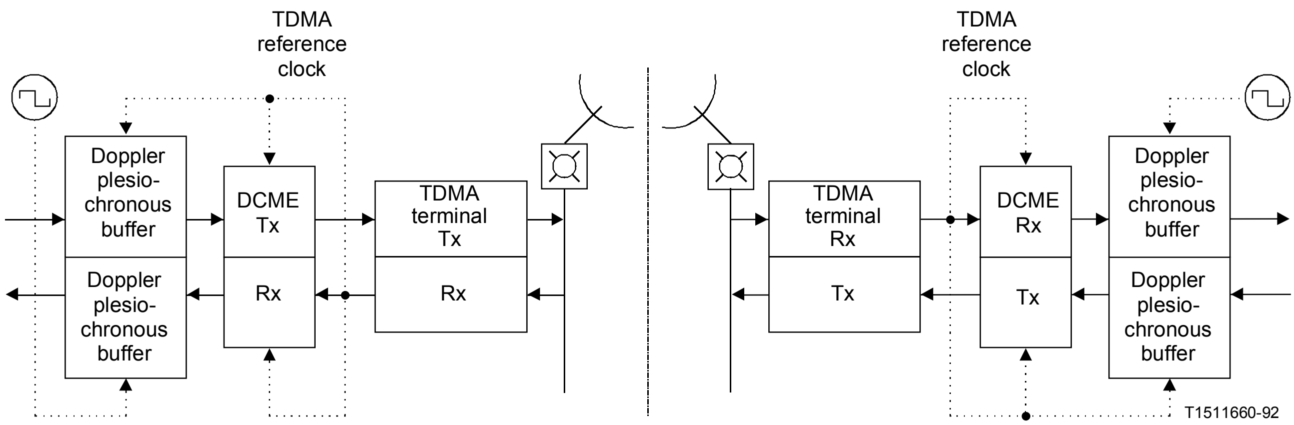


FIGURE B-13/G.763

DCME synchronous operation
(between two plesiochronous networks)

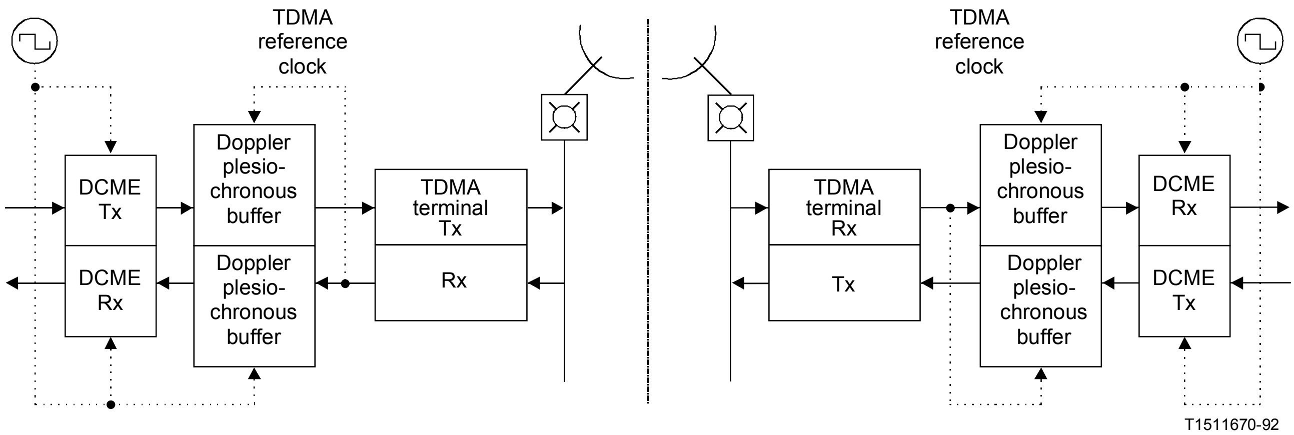


FIGURE B-14/G.763

DCME buffered plesiochronous operation
(between two plesiochronous networks)

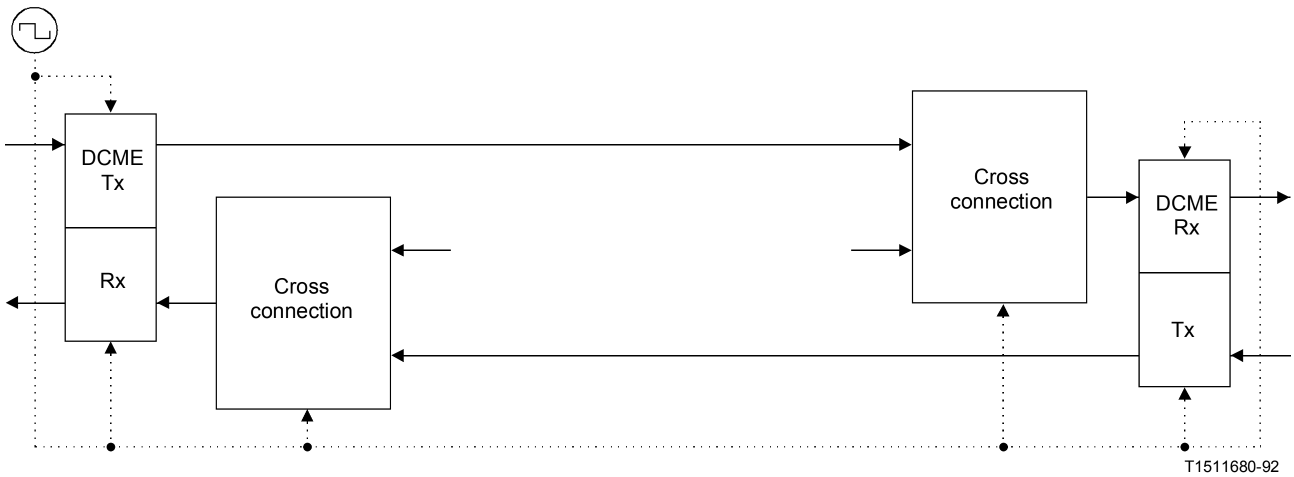


FIGURE B-15/G.763

DCME synchronous operation
(in a single synchronous network)

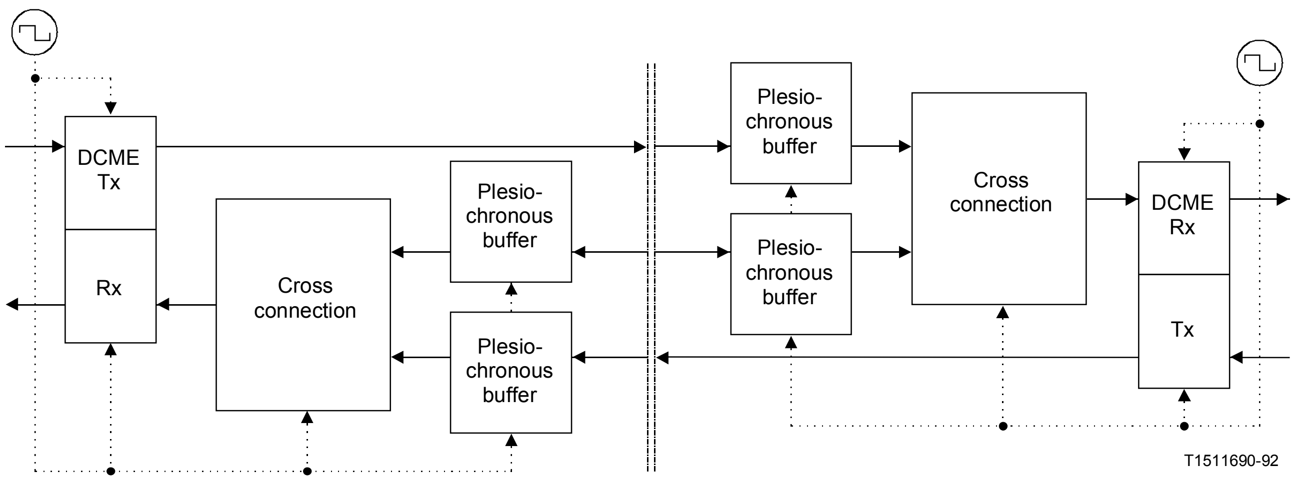


FIGURE B-16/G.763

DCME buffered plesiochronous operation
(between two plesiochronous networks)

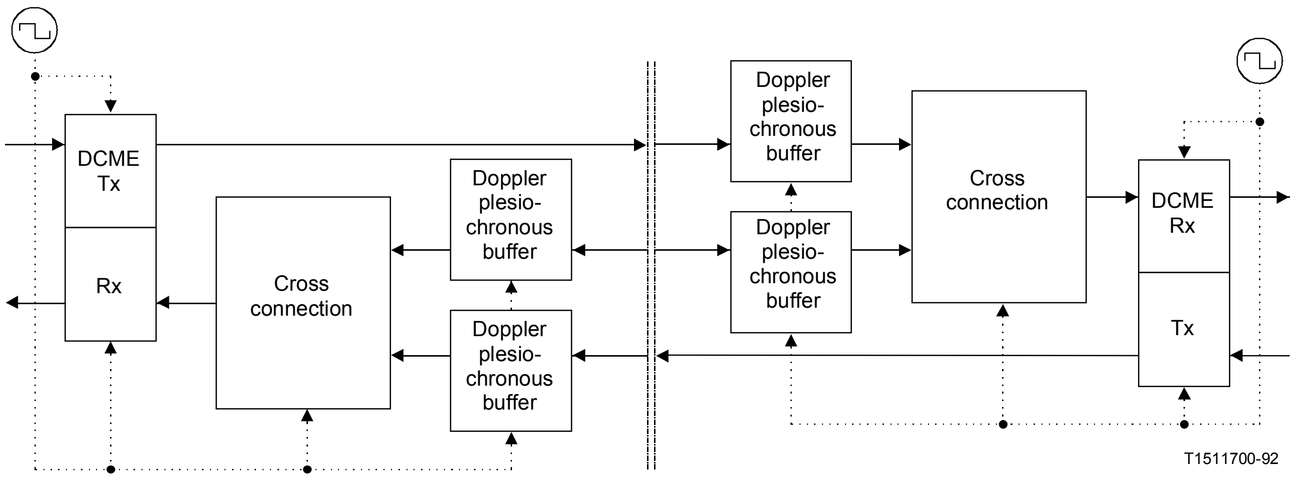


FIGURE B-17/G.763

DCME buffered plesiochronous operation
(between two plesiochronous networks)

B.6.3 *Multi-destination operation*

B.6.3.1 *Terrestrial operation within a national network*

Figure B-18/G.763 shows a method of DCME terminal synchronization for operation within a national network. The received data streams are assumed to originate from mutually synchronized sources.

B.6.3.2 *Terrestrial operation between national networks*

Figure B-19/G.763 shows a method of DCME terminal synchronization for operation between national networks via terrestrial facilities. Plesiochronous buffers are required to resolve timing differences between the various plesiochronous networks. Due to the multiple source nature of the multi- destination configuration, the plesiochronous buffers must be placed before the DCME receive function.

B.6.3.3 *Satellite operation between national networks based upon continuous carrier type services*

Figure B-20/G.763 shows a method of DCME terminal synchronization for operation between national networks based on continuous digital satellite carriers. Plesiochronous/doppler buffers are required to resolve timing differences between the plesiochronous networks and to remove satellite induced doppler shifts on the received data streams. Due to the multiple source nature of the receive signals in the multi-destination configuration, the plesiochronous/doppler buffers must be placed before the DCME receiver.

B.6.3.4 *Satellite operation between national networks based upon TDMA-type services*

Figures B-21/G.763 and B-22/G.763 show a method of DCME terminal synchronization for operation between national networks over a satellite link based on TDMA-type services. An appropriate interface is provided in the TDMA terminal to permit interfacing the DCME over a primary multiplex port. The configuration shown in Figure B-21/G.763 permits slip free operation between the DCMEs.

B.7 *Performance*

B.7.1 *Speech performance* (provisional)

Recommendation P.84 describes a subjective test method for comparing the performance of DCME systems against suitable reference conditions for carefully defined input signals. Recommendation P.84 consists of listening tests and is the recommended source of information about subjective testing of DCME. These tests are a first step and do not preclude the need for conversational tests.

It is recommended that a fixed delay be inserted in the transmit speech path to reduce the probability of front end clipping. This delay compensates for activity detection time and DCME assignment message connection delay. The delay should be such as to assure that the main speech spurt clipping is less than 5 ms.

B.7.2 *Voice band data performance*

Extensive testing has demonstrated satisfactory voice band data performance for the 40 kbit/s algorithm specified in Recommendation G.726 for voice band data rates up to 9600 bit/s.

Voice band data at rates up to 12 000 bit/s can be accommodated by 40 kbit/s ADPCM. The performance of V.33 modems operating at 14 400 bit/s over 40 kbit/s ADPCM is for further study. Selection of a 64 kbit/s unrestricted channel through a DCME is also possible and may be used for V.33 modems operating at 14 400 bits.

FIGURE B-18/G.763

DCME synchronous operation
(in a single synchronous network)

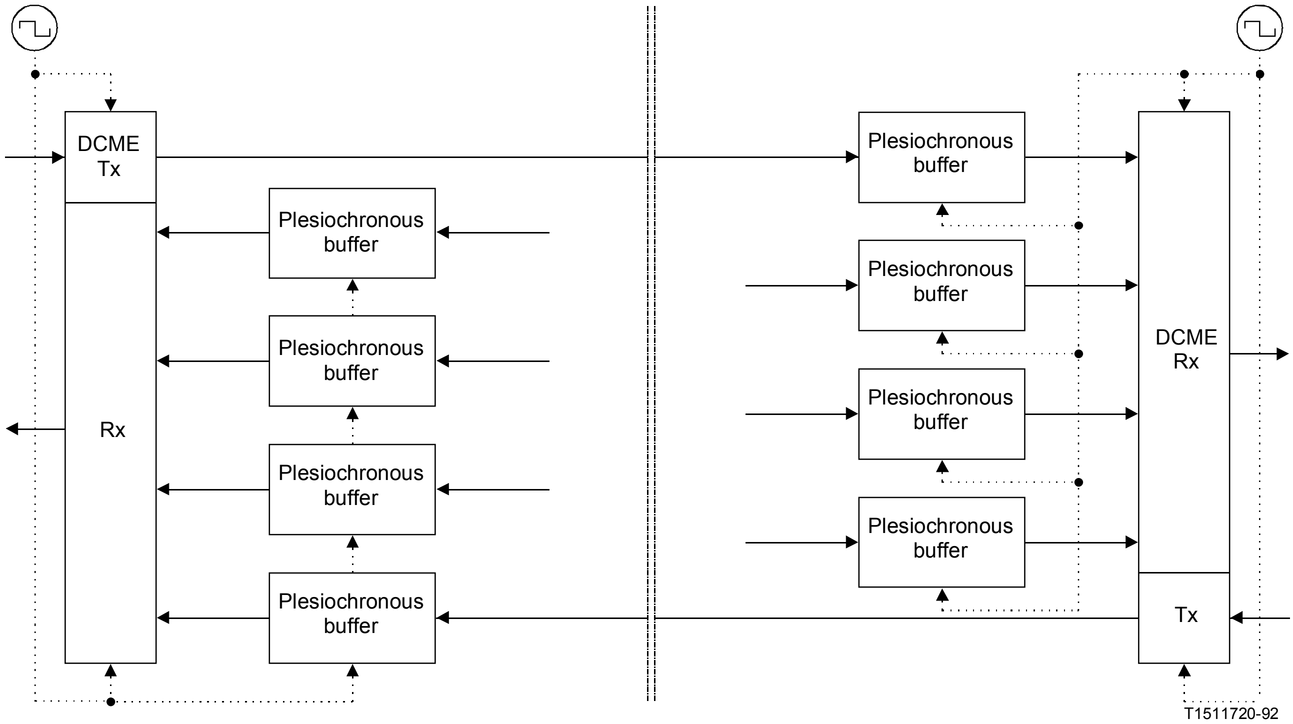


FIGURE B-19/G.763

DCME buffered plesiochronous operation
(between two plesiochronous networks)

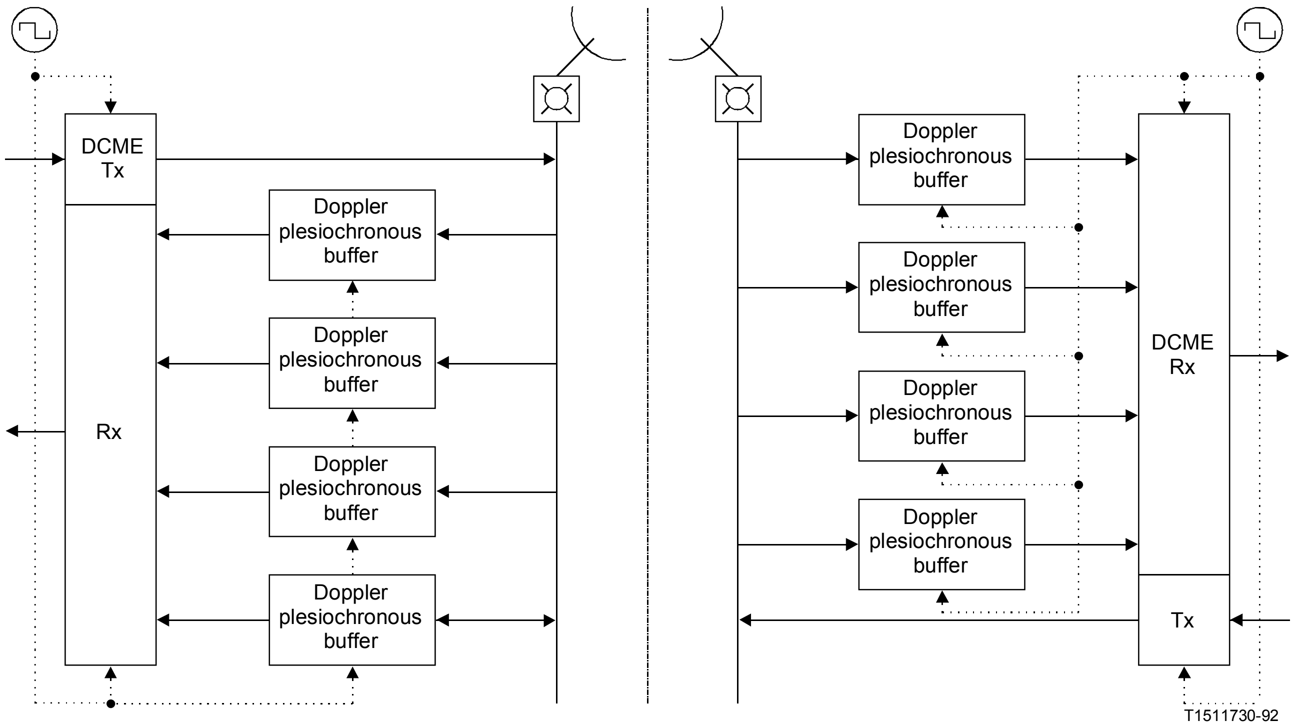


FIGURE B-20/G.763

DCME buffered plesiochronous operation
(between two plesiochronous networks)

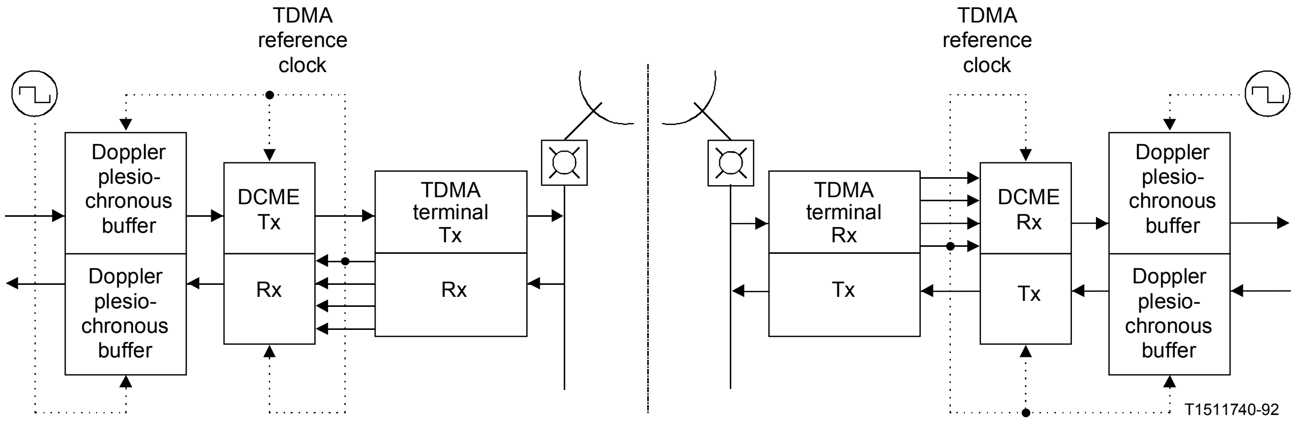


FIGURE B-21/G.763

DCME synchronous operation
(between two plesiochronous networks)

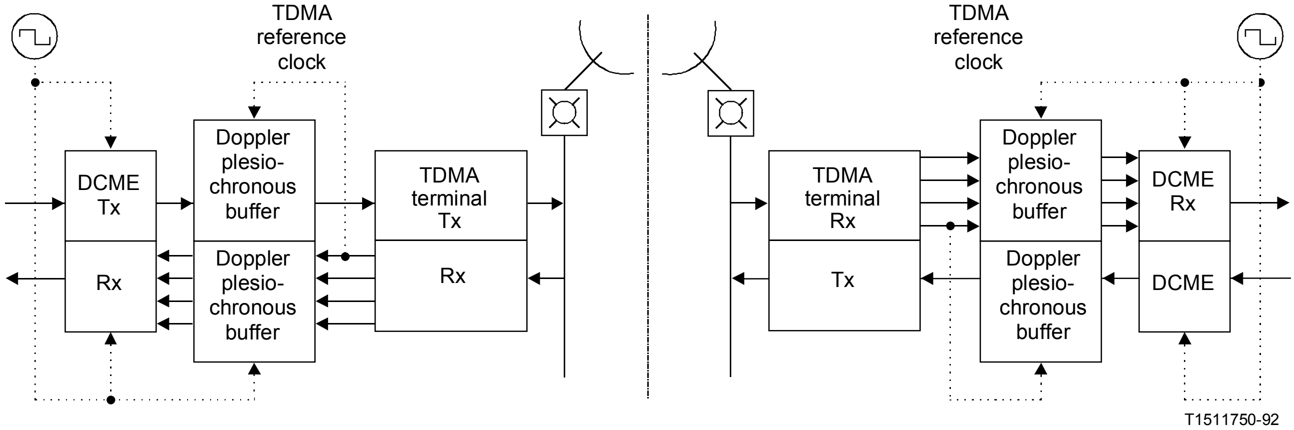


FIGURE B-22/G.763

DCME buffered plesiochronous operation
(between two plesiochronous networks)

Supplement No. 1

DCME TUTORIAL

(to Recommendation G.763)

1 Use of digital circuit multiplication system (DCMS)

DCMS provide the means to reduce the cost of transmission (e.g. long distance transmission) by making use of the combination of digital speech interpolation (DSI) and low rate encoding (LRE) techniques.

DSI is used to concentrate a number of input channels (generally referred to as trunk channels) onto a smaller number of output channels (generally referred to as bearer channels). It does this by connecting a trunk channel to a bearer channel only for the period that the trunk channel is active, i.e. is carrying a burst of speech or voice-band data. Since in average conversations one direction of transmission is active only for 30% to 40% of the time, if the number of trunks is large the statistics of the speech and silence distributions will permit a significantly smaller number of bearer channels (bearer channel pool) to be used. Control information must also be passed between the terminals to make sure that bearer and trunk channel assignments at each end remain synchronized.

LRE uses digital filtering techniques to construct an estimate of the waveform at both the encoder and the decoder. Since the actual information rate of speech is much lower than the channel Nyquist rate the link used between the LRE encoder and the decoder can operate at a rate which is dependent mainly on the quality of the models and the permissible amount of transmission degradation. The CCITT has standardized in Recommendations G.726 and G.727 a type of LRE known as ADPCM, the performance of which has been extensively characterized. DCME uses the ADPCM defined in Recommendation G.726.

Facsimile compression uses recognition and decoding of some or all of the voice-band signals sent by the modem to enable the sub-multiplexing of the digital information from a number of trunk channels onto a reduced number of bearer channels with the object of enhancing both the quality and the efficiency of transmission as compared to rate reduction of the signals using ADPCM. This is under study.

The simplest way to use DCMS is in the single destination mode as shown in Figure 1/G.763. This mode of operation is most economic for the largest routes. For smaller routes there are two options:

– operation in multi-clique mode,

– operation in multi-destination mode.

Operation in multi-clique mode, see Figure 2/G.763, divides the bearer channels into a number of blocks or cliques, each associated with a different route. There is normally a fixed boundary between cliques, and trunk/bearer channel assignments are generally carried in a control channel within the clique to which they refer. This limits the dynamic processing of received channels to those which are contained in the wanted clique; selection of the wanted clique channels can be done using a simple static digital switch without reference to the assignment information. With a 2048 kbit/s bearer system in multi-clique DCMS the statistics of the DSI are unpromising with more than three routes. Recommendation G.763 provides for two cliques.

Operation in multi-destination mode, see Figure 3/G.763, permits any bearer channel to be associated with any trunk channel of any of a number of different routes. There is no segregation of routes on the bearer, and therefore at the receive terminal it is impossible to select the wanted channels without reference to the assignment information. Multi-destination mode is economic for very small routes via satellite, but practical difficulties limit the number of routes which it is desirable to have on a single DCMS.

2 Location

Location of DCME depends on its use. Equipment used in single destination mode or in multi-clique mode can in general be located at:

- ISC,
- earth station,
- cable head,

without significant restrictions. Equipment used in the multi-clique mode will typically be located at the ISC so that the advantages of DCMG can be extended over the national section. Equipment used in the multi-destination mode will typically be located at the earth station or cable head. The reason for this is that whereas in multi-clique mode the number of receive bearer channels at the DCME terminal is approximately equal to the number of transmit bearer channels, in multi-destination mode the number of receive bearer channels at the DCME terminal is the number of transmit bearer channels multiplied by the number of destinations. It therefore may be uneconomic to provide sufficient transmission capacity between earth station and ISC to permit location of multi-destination DCME at an ISC.

3 Transmission requirements

DCMS are usually required to carry any traffic which can be carried on ordinary General Switched Telephone Network (GSTN) connections. That includes voice-band data using V-Series Recommendation GSTN modems, facsimile calls following Recommendations T.4 and T.30 and using V.29 modems. In addition, in the ISDN 64 kbit/s unrestricted on-demand digital data and alternate speech/64 kbit/s unrestricted bearer services must be carried.

DCMS are primarily designed to maximize the efficiency of speech transmission. Use with voice-band data, especially at high rates, presents problems. These problems are mainly due to the difficulty for 32 kbit/s ADPCM of encoding voice-band data waveforms.

4 DCME gain (DCMG)

The gain of DCME is the input trunk channel transmission multiplication ratio, which is achieved through application of DCME, including LRE and DSI (for a specified speech quality at a certain level of bearer channel activity). The maximum available gain depends on:

- number of trunk channels;
- number of bearer channels;
- trunk channel occupancy;
- speech activity;
- voice-band data traffic;
- ratio of half duplex to full duplex voice-band data;
- type of signalling;
- 64 kbit/s traffic;
- minimum acceptable speech quality;
- dynamic load control threshold.

Of these the factor which has the greatest significance is the percentage of 64 kbit/s digital data traffic. This is because a trunk channel carrying 64 kbit/s traffic requires two 32 kbit/s bearer channels to be removed from the pool of channels available to the DSI process.

The peak percentage of voice-band data may vary between 5 and 30 per cent, depending on route. This is discussed in greater detail in Supplement No. 2.

The type of signalling system used on the route can significantly affect the gain. Continuously compelled signalling systems hold channels active for undesirably long periods. In the case of CCITT R2 digital signalling via a DCMS used on a satellite, the channel might be active for 5 to 14 seconds.

The measured speech activity depends on the characteristics of the activity detector. It is usual to assume 35 to 40 per cent. Channels with high ambient background noise can increase this activity factor. Outside of the route busy hour the occupancy of the trunk channels by traffic will be lower than in the route busy hour. The effect of this is to reduce the ensemble activity measured by the activity detector to about 27 per cent outside the route busy hour, whereas it will be close to the speech activity factor, i.e. about 40 per cent during the route busy hour.

The speech quality is governed by two main factors; the LRE encoding rate, and the amount of speech lost while a newly active trunk channel is awaiting connection to a bearer channel. If there are a great many newly active trunk channels in competition the beginning of a burst of speech is more likely to be clipped or frozen out than if relatively few trunk channels are active.

The speech quality of a DCME in a network with external echo control devices may be affected by clipping introduced by echo control devices and by a possible noise contrast effect. In particular when echo suppressors or echo cancellers are used on circuits where the near end generated noise is high with respect to the noise generated in the remainder of the link, suppression of the far end noise may be objectionable due to noise contrast. Possible means of eliminating this problem are use of echo control devices which insert idle line noise at the appropriate level during suppression periods, or insertion of idle line noise at the DCME during the relevant period when the echo control device is integrated in the DCME. Another approach is discussed in Annex B, § B.5 to Recommendation G.763.

When commissioning a new DCMS, observations should be made of the type and characteristics of the traffic which will use it. It is unwise to rely solely on customer complaints to indicate when a system is poorly dimensioned. This is because interactions between the DCMS and echo control (note) may obscure the true problem. Furthermore the consequence of trying to concentrate too many trunk channels onto too few bearers may be simply to increase the calling rate and to reduce the call holding time. This may result in greatly reduced quality, especially where continuously compelled signalling systems are used, and levels of trunk channel activity occur far above what was envisaged in the original system dimensioning.

Note – This highest speech quality is obtained when echo cancellers conforming to Recommendation G.165 (Red Book) are used for echo control. However echo suppressors conforming to Recommendations G.164 (Red Book) and G.161 (Yellow Book) may be used.

Two possible criteria for acceptable speech performance are an average of 3.7 bits per sample and less than 2.0% probability of clipping exceeding 50 ms, or alternatively that less than 0.5% of speech should be lost due to clipping.

Using the above criteria, approximations have been derived that relate the percentage of voice-band data and the number of trunk channels to the gain of a DCME. Approximations intended for use in initial system dimensioning are given in Supplement No. 2 to Recommendation G.763.

If a more accurate representation is required, then it will be necessary to do the first order Markov chain analyses referred to in the literature on DSI [1], [2], [3].

5 ISDN bearer services

DCMS are generally required to carry the full range of ISDN bearer services which can be provided on a 64 kbit/s channel as specified in Recommendation I.230 (Blue Book). These are:

– Circuit mode 64 kbit/s unrestricted, 8 kHz structured bearer service category.

  This may be used among other things for speech, multiple sub-rate information streams multiplexed by the user, or for transparent access to an X.25 public network.

– Circuit mode 64 kbit/s, 8 kHz structured bearer service category, usable for speech information transfer.

  This is broadly similar to the preceding category, but with different access protocols.

– Circuit mode 64 kbit/s, 8 kHz structured bearer service category, usable for 3.1 kHz audio information transfer.

  This bearer service provides the transfer of 3.1 kHz bandwidth audio information, such as for example voice-band data via modems, Group I, II and III facsimile information, and speech.

– Circuit mode alternate speech/64 kbit/s unrestricted 8 kHz structured bearer service category.

  This service is similar to both the unrestricted and speech 64 kbit/s circuit-mode bearer services, but provides for the alternate transfer of either voice or unrestricted digital information at 64 kbit/s within the same call.

6 Restoration of services

For most applications the loss of traffic under failure conditions would be such that it would be insufficient to install a single pair of terminals on a route without a means of rapid changeover to spare equipment in the event of failure. This means that DCME is often used in a cluster of N active DCMEs for one standby. Automatic changeover permits the standby to be loaded with the configuration and synchronization information of the failed terminal. Other automatic fallback modes may be considered.

Failure of the transmission system between DCME terminals can be handled by normal transmission system restoration procedures. Failure of the transmission systems entering the DCME terminals from the exchanges may result in a wide range of different alarm conditions being experienced particularly where a multi-destination DCME terminal serves more than one exchange and more than one route. It is desirable to limit the generation of alarm conditions to the channels which have actually failed.

7 Control of transmission overload

The reduction in the number of bearer channels available to the interpolation process, due to the high activities of voice band and 64 kbit/s data services or statistical variations in the ensemble input speech activity can occur when the number of instantaneously active trunk channels exceeds the number of available bearer channels. Either event requires action to be taken to safeguard speech quality. There are four possible solutions:

– The system can be dimensioned so that with the maximum anticipated short-term trunk channel activities there is negligible probability of violating the speech quality criteria. This employs the DCMS very inefficiently outside the busy hour.

– A multi-destination system can be made to carry routes with widely different busy hours, so that though the trunk channels might have relatively low non-busy hour occupancy, the bearer channels would always be well loaded.

– Signals can be sent from the DCME to the exchange to busy out part of the route when the quality criteria are violated. This is known as dynamic load control (DLC), and can be an effective control method, but it cannot be retrospective and it is slow to take effect. Furthermore care must be taken to ensure that when circuits are returned to service the increase in bearer channel activity is not sufficient to result in the immediate reapplication of DLC.

– The signal to quantization performance can be traded against the clipping of speech bursts. By using variable rate ADPCM algorithms it is possible to quantize to three or optionally two rather than four bits on individual speech channels on a pseudo-cyclic basis for a given number of samples. In this way the system can be given a gradual degradation characteristic, rather than suddenly overloading.

In a DCME conforming to Recommendation G.763 all of these techniques may be used.

8 Transmission link performance monitoring

Experience with DCMEs has shown the value of using cyclic redundancy check information in the detection and tracing of certain faults. In order to provide a comprehensive set of long-term and short-term indicators the DCME should provide the following means of monitoring the performance of any digital path(s) terminated upon it:

– cyclic redundancy check (CRC);

– frame alignment signal (FAS);

– other primary rate alarms;

– far end block error information of distant CRC (FEBE);

– DCME control channel FAS;

– violations of the Golay FEC of the control channel(s).

References

[1] KOU (K.Y.), O'NEAL (J.B.), NILSON (A.A.): Computations of DSI (TASI) overload as a function of the traffic offered, *IEEE Trans. on Communications*, Vol. COM-33, No. 2, February 1985.

[2] BRADY (P.T.): A model for generating on-off speech patterns in 2-way conversation, *Bell System Technical Journal*, page 2445 *et seq*, September 1969.

[3] Special issue on bit rate reduction and speech interpolation, Guest Editors M.R. Aaron and N.S. Tayant, *IEEE Trans. on communications*, Vol. COM-30, No. 4, April 1982.

Supplement No. 2

DCME DIMENSIONING METHODS FOR DIFFERENT ROUTE CHARACTERISTICS

(to Recommendation G.763)

1 Introduction

This supplement draws attention to the implications of the measurements of channel occupancy and voice-band data levels which have been done on particular routes for which the number of voice-band data calls is either large in absolute terms, or large compared to the total number of calls.

2 Route profiles

Figure 1 shows the kind of profile which has been obtained from measurements on an FDM route between the United Kingdom and a country for which the proportion of voice-band data calls was suspected to be high. It can be seen from this that there are two peaks which are of interest in DCME dimensioning – one (the voice peak) where voice is the dominant feature with a relatively small amount of voice-band data, and another (the data peak) where voice-band data dominates voice.

Note – The data profile is not symmetric in each direction of transmission.

Voice-band data requires more bearer capacity than voice in a DCME system incorporating digital speech interpolation (DSI) and low rate encoding (LRE) and therefore it is not immediately obvious which of these peaks is the limiting factor when calculating the achievable gain of a DCME on a particular route. Each route has to be examined carefully to determine the achievable gain. The limiting value of the gain does not necessarily occur at either of the peaks and in practice a scan across several days' profiles is necessary to determine the achievable gain.

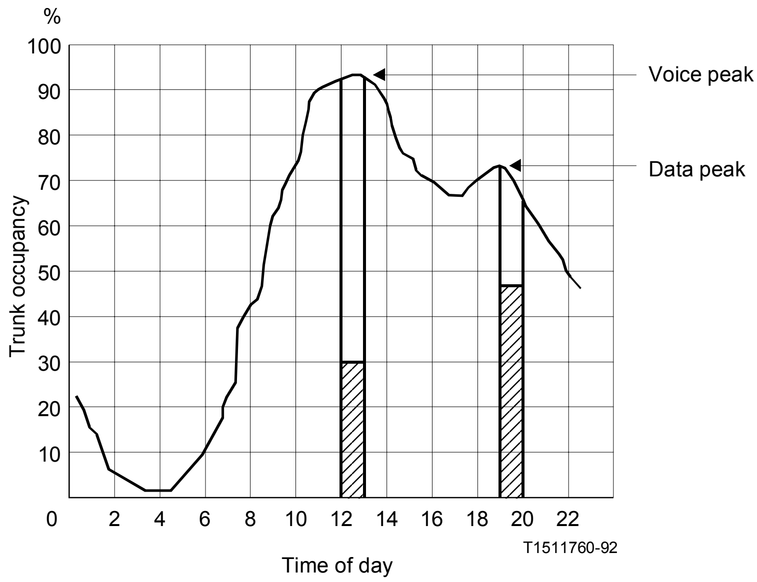


FIGURE 1

FDM profile

Figure 2 shows a typical profile obtained from the TDMA route for the same country. Due to different traffic origins and loading distributions the voice and data peaks are coincident, and the transmit and receive profiles are more nearly symmetrical in this case.

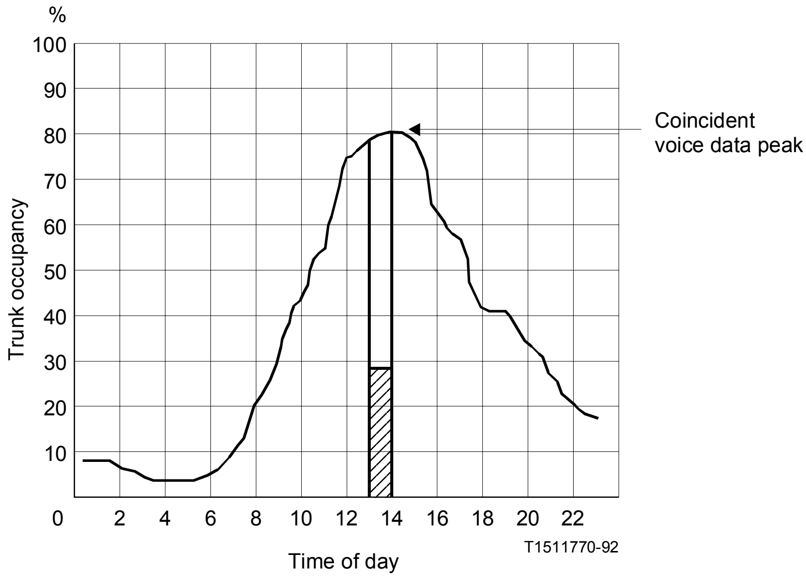


FIGURE 2

TDMA profile

3 DCME operation

Figure 3 shows a DCME consisting of a DSI stage and an LRE stage. Voice and voice-band data have to be treated separately in each of these stages when trying to access the achievable gain of a particular DCME faced with a particular route profile.

3.1 *DSI gain for voice*

This is dependent upon the number of input trunks carrying voice and it is *not a linear relationship*.

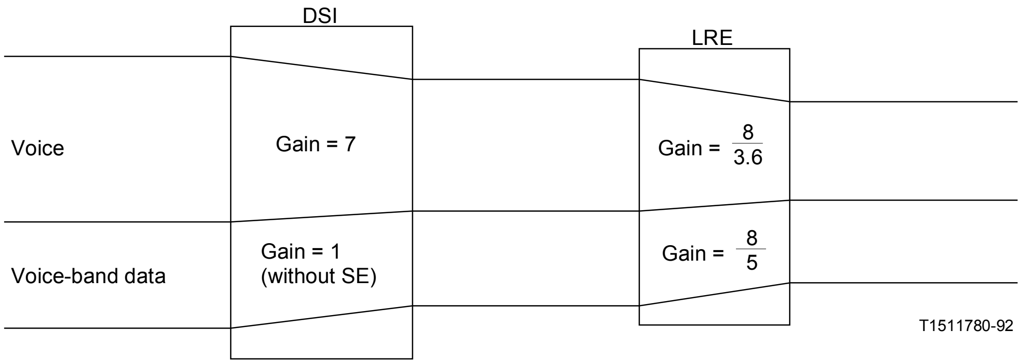


FIGURE 3

DCME gain

3.2 *DSI gain for data*

Facsimile is the dominant data service and can be considered as half duplex, i.e. on a particular call if data is flowing in one direction of transmission at a particular time, then the opposite direction is silent. If the total amount of facsimile traffic in one direction of transmission is balanced by an equivalent amount in the opposite direction of transmission then a technique known as silence elimination can be employed to free the opposite channel when data is flowing in one direction. This leads to a theoretical DSI gain of 2. However, if the total facsimile traffic on a route is not balanced in each direction of transmission, making silence elimination difficult to implement (or if silence elimination has not been built into a particular DCME) then the DSI gain for voice-band data is 1.

3.3 *LRE gain for voice*

Studies have indicated that the minimum acceptable average number of bits per sample is of the order of 3.6, which will be the threshold for operation of dynamic load control. Therefore the LRE gain for voice is unlikely to exceed 8/3.6.

3.4 *LRE gain for data*

The LRE gain for data depends on how many bits/sample a particular system allocates to a data call.

In this supplement all calculations assume the use of the 40 kbit/s encoding rate for voice-band data, in conformity with Recommendation G.763, therefore the LRE gain for data = 8/5.

Examples for facsimile compression are not presented.

4 Calculation of DCME gain

Table 1 gives some approximate non-analytical formulas for calculation of the voice part of the DCME gain. It should be noted that these approximations are strictly valid only for DCMEs conforming to Recommendation G.763 and having ideal speech detection (i.e. the activity indicated by the speech detector is the same as the actual speech activity).

TABLE 1

Formulas for voice interpolation gain (Gv)

No. of bits/sample	No. of trunks (N)	Formula	Activity factor (AF)		
			33%	35%	37%
3.6	N < 80	Gv = a + b × 1n(N)	a = 0.23 b = 0.61	a = 0.04 b = 0.60	a = 0.30 b = 0.51
	N > = 80	$Gv = \frac{1.1388 \times N}{N \times AF + \sqrt{N \times AF}}$	AF = 0.33	AF = 0.35	AF = 0.37
3.7	N < 80	Gv = a + b × 1n(N)	a = 0.23 b = 0.61	a = 0.04 b = 0.60	a = 0.27 b = 0.52
	N > = 80	$Gv = \frac{1.1081 \times N}{N \times AF + \sqrt{N \times AF}}$	AF = 0.33	AF = 0.35	AF = 0.37
3.8	N < 80	Gv = a + b × 1n(N)	a = 0.24 b = 0.59	a = 0.01 b = 0.61	a = 0.28 b = 0.51
	N > = 80	$Gv = \frac{1.0789 \times N}{N \times AF + \sqrt{N \times AF}}$	AF = 0.33	AF = 0.35	AF = 0.37

4.1 *Limitations*

Ideally the calculation of the DCME gain would be done by a comprehensive computer modelling of the system in the way which has already been demonstrated with great success by Swedish Telecom Radio. Given an intimate knowledge of the route, in terms of its hourly, daily and seasonal variations in voice and voice-band data traffic flow, signalling systems, call holding times and effective/ineffective ratios over a period of time it may be possible to model the route with a high degree of accuracy, at least retrospectively. However the major limitation is the quality of the information fed into the model. To overcome this limitation the digital channel occupancy analyser (DCOA) has been developed. If the DCOA is used on a group of circuits which previous sampling or other information has shown to be typical then very useful dimensioning information results. The limitation then is the total permissible measuring time. In most cases, for operational reasons, greater than two weeks is unlikely to be feasible. This represents a severe limitation on the attempt to create an accurate model, such that for dimensioning (as opposed to the verification of the operation of the equipment) Monte Carlo type simulations do not appear to be necessary.

4.2 *Example gain calculations using simplified techniques*

The following examples illustrate the concepts outlined in § 2, and demonstrate the use of a simplified technique for DCME dimensioning using DCOA route profiles.

4.2.1 *DCME dimensioning using the profile of a route without silence elimination*

Assumptions:

Number of trunk channels at service date = 240.

Figure 4 is the applicable DCOA route profile.

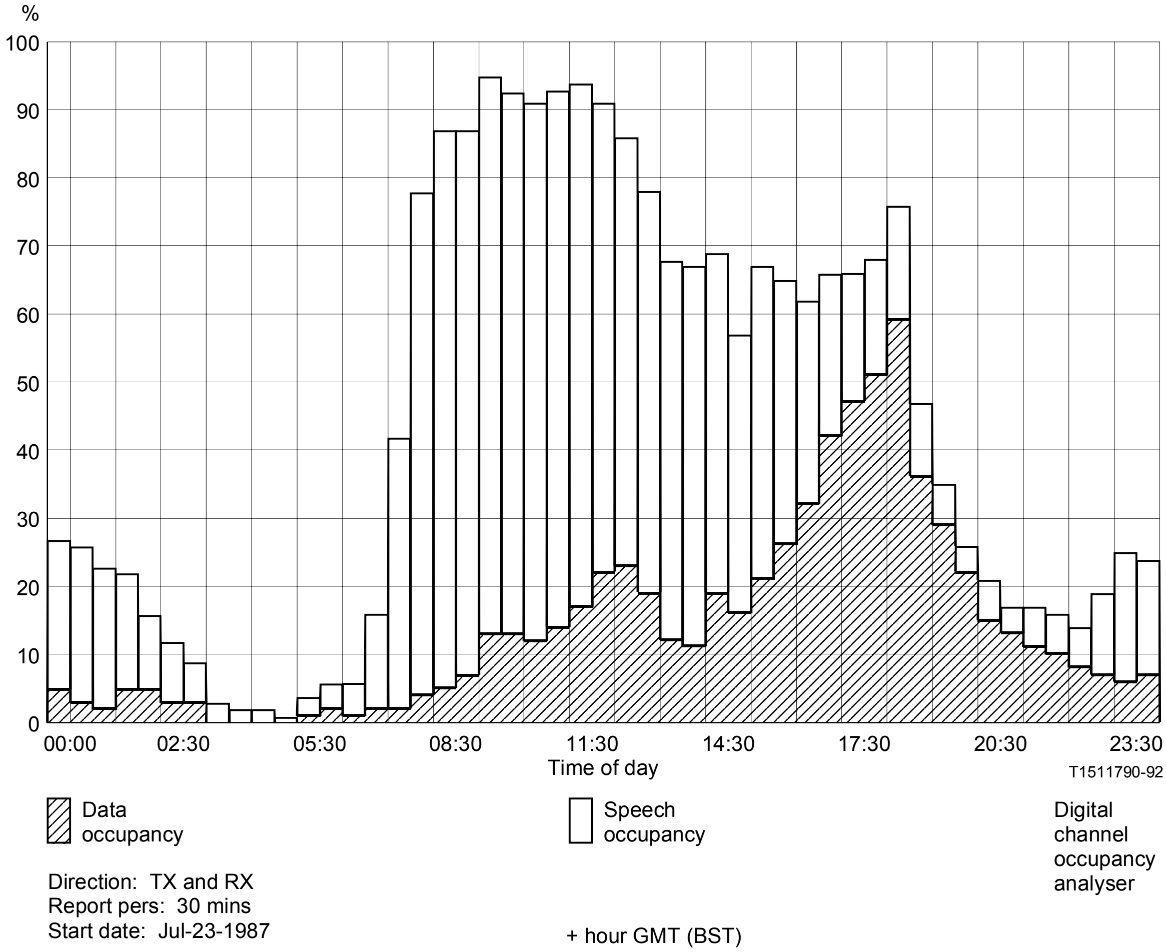


FIGURE 4

DCOA route profile for example 1

Remark:

From experience or from rough calculations it can be seen that for the given number of trunk channels and quantity of voice-band data traffic at least three DCMEs each using 30 bearer channels are likely to be required, but let us assume that four DCMEs are to be used on the route in order to calculate the gain for the voice traffic (this gain is dependent upon how many DCMEs the voice traffic is spread over). This is to ensure that the DCMEs are not overloaded and may also allow for growth on the route. In practice an iterative procedure would have to be used to determine the optimum number of DCMEs for each route.

From Figure 4 there are two peaks to be considered. One is dominated by the amount of data (data peak) and the other is dominated by the amount of voice (voice peak):

Data peak

59% data:

number of data trunks $= 240 \times 0.59$

$= 142$ trunks,

number of data trunks per DCME $= \frac{142}{4}$

$= 36$

DSI gain = 1 (no silence elimination advantage to be gained because almost all the data is in one direction of transmission)

LRE gain $= \frac{8}{5}$

17% voice:

number of voice trunks $= 240 \times 0.17$

$= 41$ trunks total

number of voice trunks per DCME = 10

DSI gain (for 10 trunks) = 1.25 (from tables)

LRE gain $= \frac{8}{3.6}$

Hence the 64 kbit/s bearer channel requirement is:

$$\frac{36 \times 5}{8} + \frac{10 \times 3.6}{1.25 \times 8}$$

= 23 (data) + 4 (voice)

= *27 bearer channels.*

The total bearer requirement is therefore:

27×4

= *108 bearer channels.*

Voice peak

13% data:

number of data trunks $= 240 \times 0.13$

$= 32$ trunks total,

number of data trunks per DCME $= \frac{32}{4}$

$= 8$

DSI gain = 1 (no silence elimination advantage to be gained because almost all t the data is in one direction of transmission),

LRE gain $= \frac{8}{5}$

83% voice:

number of voice trunks = 240×0.83

= 200 trunks total

number of voice trunks per DCME = 50

DSI gain (for 50 trunks) = 1.92 (from tables)

LRE gain = $\frac{8}{3.6}$

Hence the 64 kbit/s bearer channel requirement per DCME is:

$$\frac{8 \times 5}{8} + \frac{50 \times 3.6}{1.92 \times 8}$$

= 5 (data) + 12 (voice)

= *17 bearer channels.*

The total bearer requirement is therefore:

17×4

= *68 bearer channels.*

Inference:

It seems therefore that in this case the DCME dimensioning is determined by the number of trunk channels required to cope with the speech peak, and by the number of bearer channels required to handle the data peak. Since the number of channels shown as active by the DCOA is an average over the measurement interval, it is reasonable to assume that all 240 trunk channels, rather than only 132 were active for some brief duration. Assuming that only the wanted bearer channels are used, and neglecting the assignment channel, the achievable gain will be:

$$\frac{240}{108} = 2.22.$$

4.2.2 *DCME dimensioning using the profile of a route with silence elimination*

Assumptions:

Number of trunk channels at service date = 347.

Figure 5 is the applicable DCOA route profile.

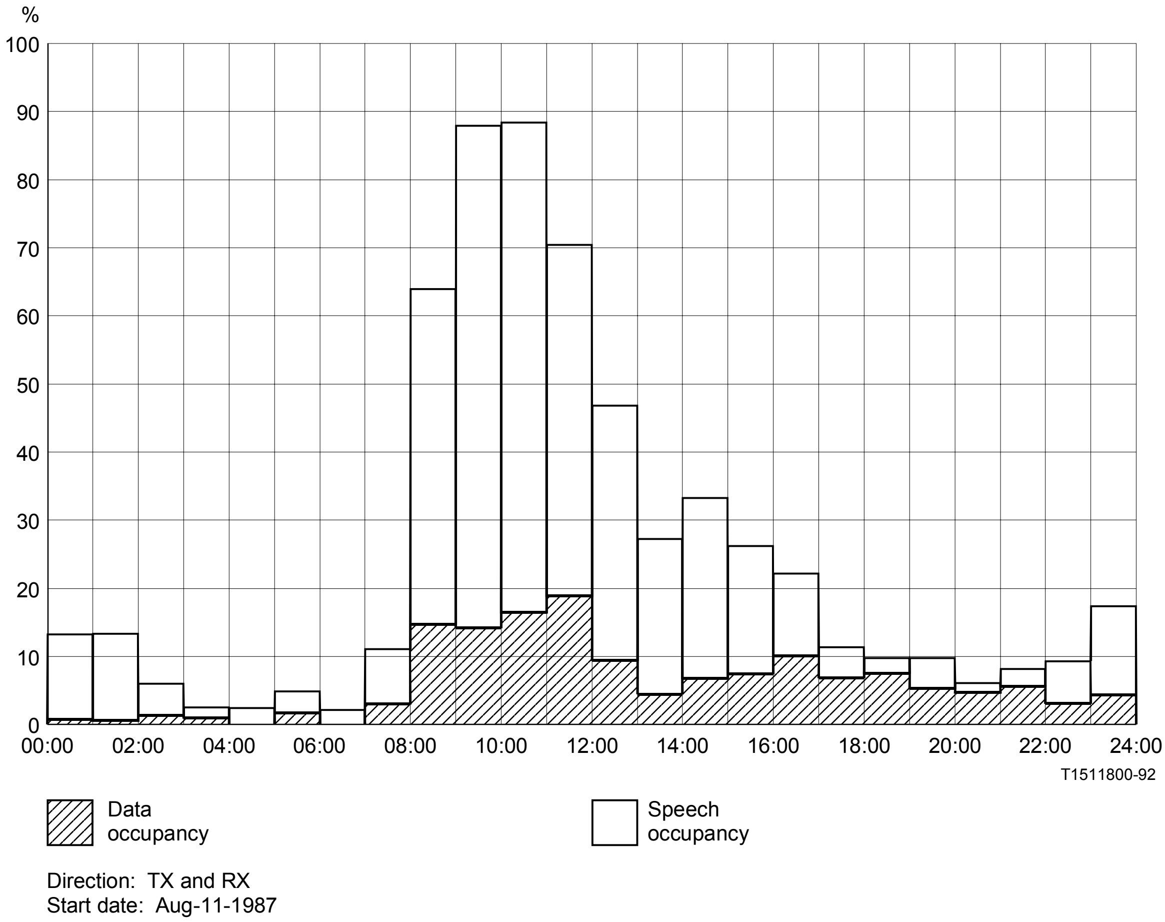


FIGURE 5

DCOA route profile for example 2

Remark:

On this route it appears that use of silence elimination will give some benefits. Other DCOA measurements have indicated that there is approximately twice as much voice-band data activity in the transmit direction as in the receive direction. Therefore the achievable DSI gain on voice-band data due to silence elimination is of the order of 1.5. This assumes that there are as many transmit as receive bearer channels on each DCME terminal. Rough calculations and experience indicate that because of the relatively low voice-band data percentage of this example three DCMEs will probably be sufficient.

From Figure 5 there is only one peak to be considered:

15% data:

number of data trunks $= 347 \times 0.15$

$= 52$ trunks

number of data trunks per DCME $= \dfrac{52}{3}$

$= 18$

DSI gain $= 1.5$ (due to silence elimination)

LRE gain $= \dfrac{8}{5}$

72% voice:

number of voice trunks $= 347 \times 0.72$

$= 250$ trunks total

number of voice trunks per DCME $= 83$

DSI gain (for 83 trunks) $= 2.08$ (from tables).

Hence the 64 kbit/s bearer channel requirement per DCME is:

$$\frac{18 \times 5}{1.5 \times 8} + \frac{83 \times 3.6}{2.08 \times 8}$$

$= 8$ (data) $+ 19$ (voice)

$=$ *27 bearer channels.*

The total bearer requirement is therefore:

27×3

$=$ *81 bearer channels.*

Inference:

In this case, assuming that only the wanted bearer channels are used, the DCME can achieve a gain of:

$$\frac{347}{81} = 4.28.$$

However, as was shown by the previous example, it would be very unwise to assume that a DCME gain as high as four will be achievable for all types of DCME, without careful consideration of the route conditions. A corollary to this is that when a DCME has been installed on a route its performance must be continually monitored to ensure that changes in the traffic distribution on the route do not impact seriously upon the transmission quality.

4.3 *Two pitfalls for the unwary*

Figure 6 shows a plausible example of a DCOA record, covering a typical two hour period. On the basis of the trunk occupancy percentage for the route it might be thought that the maximum bearer occupancy would be coincident with the peak in voice traffic, however this is not so. The actual maximum occurs immediately before, as Figure 7 shows, during period 2. The reason for this is that the voice-band data traffic peaks before the voice traffic. Administrations may wish to consider whether this is a likely state of affairs; whether for example the facsimile transmission of financial results at close of business on any particular day is likely to result in follow-up telephone conversations. The relevant information for each period is summarized in Table 2.

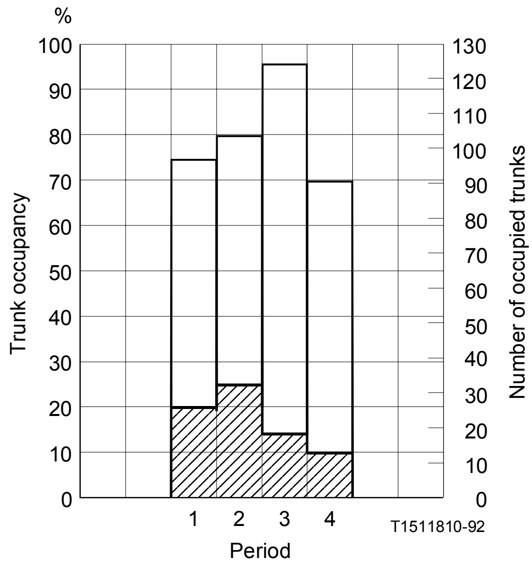


FIGURE 6

DCOA profile, trunk side

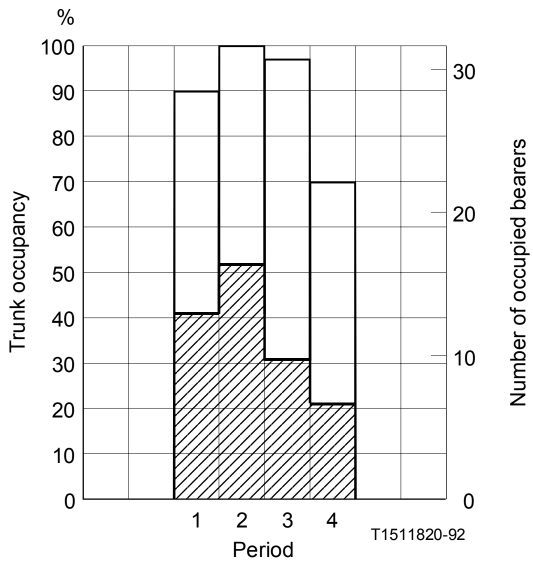


FIGURE 7

DCOA profile, bearer side

TABLE 2

Comparison of trunk and bearer occupancies

	Period							
	1		2		3		4	
	%	chs	%	chs	%	chs	%	chs
Data occupancy	20	26	25	32.5	15	19.5	10	13
Speech occupancy	55	71.5	55	71.5	80	104	60	78
Total occupancy	75	97.5	80	104	95	123.5	70	91
Data bearers		13		16.5		10		6.5
Speech bearers		15		15		21		16
Total bearers		28		31.5		31		22.5

Care must be taken when the short-term characteristics of a measured route are not known. This may be especially significant when the route is small, since the presentation of voice-band data traffic may not be very uniform. Over a five minute period 2:1 variations in the short-term voice-band data activity level are not unusual events. It might therefore be prudent to repeat any dimensioning exercises which use a DCOA profile, but doubling all the voice-band data occupancies, for comparison against the absolute maximum number of channels available when *all* voice activity is allocated 3 bits. If that comparison shows that clipping would be experienced under those conditions then a lower gain setting should be chosen, based on whichever is believed to be the limiting period.

5 Conclusion

An approach to dimensioning DCME systems has been demonstrated, which though not statistically rigorous, is nevertheless capable of giving reasonable estimates of system capabilities, given adequate input data. A number of potential dimensioning problems have been described, and the solutions outlined. These methods have been used successfully in the introduction of DCMEs on a number of routes.

Printed in Switzerland