ETSI TS 101 504 V7.1.1 (2000-08)

Technical Specification

Digital cellular telecommunications system (Phase 2+);
Inband Tandem Free Operation (TFO) of Speech Codecs;
Service Description;
Stage 3
(GSM 08.62 version 7.1.1 Release 1998)

GLOBAL SYSTEM FOR
MOBILE COMMUNICATIONS

Reference

RTS/TSGS-040862Q7R1

Keywords

TFO

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - NAF 742 C
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° 7803/88

Contents

Intellectual Property Rights

IPRs essential or potentially essential to the present document may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for **ETSI members and non-members**, and can be found in SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (http://www.etsi.org/ipr).

Pursuant to the ETSI IPR Policy, no investigation, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Foreword

This Technical Specification (TS) has been produced by the Special Mobile Group (SMG).

The present document details the Inband Signalling Protocol between Transcoder/Rate Adaptor Units for speech traffic channels for the Tandem Free Operation (TFO) of Speech Codecs within the digital cellular telecommunications system.

The contents of the present document is subject to continuing work within SMG and may change following formal SMG approval. Should SMG modify the contents of the present document it will be re-released with an identifying change of release date and an increase in version number as follows:

Version 7.x.y

where:

7 indicates Release 1998 of GSM Phase 2+

x the second digit is incremented for all changes of substance, i.e. technical enhancements, corrections, updates, etc.

y the third digit is incremented when editorial only changes have been incorporated in the specification.

1 Scope

This service description document details the Inband Signalling Protocol between Transcoder/Rate Adaptor Units for speech traffic channels for the Tandem Free Operation (TFO) of Speech Codecs.

This service description should be considered together with GSM 08.60 (Inband control of remote transcoders and rate adaptors for Enhanced Full Rate and Full Rate traffic channels) and GSM 08.61 (Inband control of remote transcoders and rate adaptors for Half Rate traffic channels).

Annex A is mandatory and describes the general Inband Signalling (IS) Principle.

Annex B is informative and gives the rules for In Path Equipment (IPE).

Annex C is the formal SDL description of the TFO Protocol as given in clause 10. Clause 10 has precedence in case of ambiguities. A part of Annex C is in electronic format. Annex C is informative. It supports the formal verification of the TFO Protocol.

2 References

The following documents contain provisions which, through reference in this text, constitute provisions of the present document.

- References are either specific (identified by date of publication, edition number, version number, etc.) or non-specific.
- For a specific reference, subsequent revisions do not apply.
- For a non-specific reference, the latest version applies.
- A non-specific reference to an ETS shall also be taken to refer to later versions published as an EN with the same number.
- For this Release 1998 document, references to GSM documents are for Release 1998 versions (version 7.x.y).

[1] GSM 01.04: "Digital cellular telecommunication system (Phase 2+); Abbreviations and acronyms".

[2] GSM 02.53: "Digital cellular telecommunication system (Phase 2+); Tandem Free Operation (TFO) of Speech Codecs; Service Description; Stage 1".

[3] GSM 03.05: "Digital cellular telecommunication system (Phase 2+); Technical performance objectives".

[4] GSM 03.53: "Digital cellular telecommunication system (Phase 2+); Tandem Free Operation (TFO) of Speech Codecs; Service Description; Stage 2".

[5] GSM 06.01: "Digital cellular telecommunications system (Phase 2+); Full rate speech processing functions".

[6] GSM 06.10: "Digital cellular telecommunications system (Phase 2+); Full rate speech transcoding".

[7] GSM 06.11: "Digital cellular telecommunications system (Phase 2+); Substitution and muting of lost frames for full rate speech channels".

[8] GSM 06.12: "Digital cellular telecommunications system (Phase 2+); Comfort noise aspect for full rate speech traffic channels".

[9] GSM 06.20: "Digital cellular telecommunications system (Phase 2+); Half rate speech transcoding".

[10] GSM 06.21: "Digital cellular telecommunication system (Phase 2+); Substitution and muting of lost frames for half rate speech traffic channels".

[11] GSM 06.22: "Digital cellular telecommunication system (Phase 2+); Comfort noise aspects for half rate speech traffic channels".

[12] GSM 06.31: "Digital cellular telecommunications system (Phase 2+); Discontinuous Transmission (DTX) for full rate speech traffic channel".

[13] GSM 06.32: "Digital cellular telecommunications system (Phase 2+); Voice Activity Detection (VAD)".

[14] GSM 06.41: "Digital cellular telecommunication system (Phase 2+); Discontinuous Transmission (DTX) for half rate speech traffic channel".

[15] GSM 06.42: "Digital cellular telecommunication system (Phase 2+); Voice Activity Detection (VAD) for half rate speech traffic channels".

[16] GSM 06.51: "Digital cellular telecommunications system (Phase 2+); Enhanced Full Rate (EFR) speech processing functions; General description".

[17] GSM 06.60: "Digital cellular telecommunications system (Phase 2+); Enhanced Full Rate (EFR) speech transcoding".

[18] GSM 06.61: "Digital cellular telecommunications system (Phase 2+); Substitution and muting of lost frames for Enhanced Full Rate (EFR) speech channels".

[19] GSM 06.62: "Digital cellular telecommunications system (Phase 2+); Comfort noise aspect for Enhanced Full Rate (EFR) speech traffic channels".

[20] GSM 06.81: "Digital cellular telecommunications system (Phase 2+); Discontinuous Transmission (DTX) for Enhanced Full Rate (EFR) speech traffic channels".

[21] GSM 06.82: "Digital cellular telecommunications system (Phase 2+); Voice Activity Detection (VAD) for Enhanced Full Rate (EFR) speech traffic channels".

[22] GSM 08.08: "Digital cellular telecommunication system (Phase 2+); Mobile-services Switching Centre - Base Station System (MSC-BSS) interface Layer 3 specification".

[23] GSM 08.54: "Digital cellular telecommunication system (Phase 2+); Base Station Controller - Base Transceiver Station (BSC - BTS) interface Layer 1 structure of physical circuits".

[24] GSM 08.60: "Digital cellular telecommunication system (Phase 2+); Inband control of remote transcoders and rate adaptors for Enhanced Full Rate (EFR) and full rate traffic channels".

[25] GSM 08.61: "Digital cellular telecommunication system (Phase 2+); Inband Control of Remote Transcoders and Rate Adaptors for half rate traffic channels".

[26] ITU-T Recommendation I.130: "Method for the characterization of telecommunication services supported by an ISDN and network capabilities of an ISDN".

3 Definitions and Abbreviations

3.1 Definitions

For the purposes of the present document, the following terms and definitions apply.

TRAU Frame: is used equivalent to "TRAU Speech Frame".

TFO Frame: is used equivalent to "TFO Speech Frame".

Abis/Ater: indicates that either the **Abis** or the **Ater** interface is used, depending on the location of the TRAU equipment.

3.2 Abbreviations

For the purposes of the present document, the following abbreviations apply.

BSC	Base Station Controller
BSS	Base Station Sub-system
BTS	Base Transceiver Station
EFR	Enhanced Full Rate
FR	Full Rate
GCME	GSM Circuit Multiplication Equipment
HR	Half Rate
IPE	In Path Equipment
MS	Mobile Station
MSC	Mobile Switching Centre
PCM	Pulse_Coded_Modulation
PCM sample	8-bit value representing the A_Law or µ_Law coded sample of a speech or audio signal; sometimes used to indicate the time interval between two PCM samples (125µs).
PCM_Silence	either PCM_Alaw_Silence, or PCM_µLaw_Silence, dependent on application
PCM_Alaw_Silence	PCM sample with value 0xD5.
PCM_µLaw_Silence	PCM sample with value 0xFF.
PCM_Idle	either PCM_Alaw_Idle, or PCM_µLaw_Idle, dependent on application
PCM_Alaw_Idle	PCM sample with value 0x54.
PCM_µLaw_Idle	PCM sample with value 0x00.
TFO_ACK	TFO Acknowledgement Message
T_Bits	Time Alignment Bits
TFO_FILL	TFO Fill Message
TFO_TRANS	TFO Transparent Mode Message
TFO_NORMAL	TFO Normal Mode Message
TFO_DUP	TFO (Half) Duplex Mode Message
TFO_REQ	TFO Request Message
TFO_SYL	TFO Sync Lost Message
TFO	Tandem Free Operation
TRAU	Transcoder and Rate Adaptor Unit

Other abbreviations used in this TS are listed in GSM 01.04.

4 General Approach

4.1 Background

In case of mobile-to-mobile calls (MS-MS calls) in GSM networks without TFO, the speech signal is encoded within the first mobile station for transmission on the air interface, and decoded within the associated first "transcoder and rate adaptor unit" (TRAU). The PCM samples are then transported within the fixed part of the network to the second TRAU using 64 kBit/s traffic links. This second TRAU encodes the speech signal a second time for the transmission on the second air interface, and the associated mobile station decodes is again. The two Codecs (Encoder-Decoder pair) of the connection are in "Tandem Operation".

This **Tandem Operation** has several **disadvantages**:

- The two consecutive encoding/decoding processes degrade the speech quality more than necessary;
- The links between the TRAUs within the fixed network need 64 kBit/s, where 16 or 8 kBit/s would be sufficient;
- The unnecessary encoding/decoding within the TRAUs allocates DSP power.

Tandem Free Operation requires two (back and forth) "transparent" digital channels or paths between the TRAUs. Devices within these paths need to be transparent or to be switched off for the **TFO Messages** and the **TFO Frames**. To guarantee this digital transparency with **out_of_band signalling** is not trivial. Out_of_band signalling is especially not fast enough for fall back to normal operation in case of sudden interruption of the transparency of the links.

This TFO recommendation defines therefore an **inband signalling protocol** which **tests,** if:

- an MS-MS call is given;
- the paths between the TRAUs are digitally transparent;
- both TRAUs support TFO;
- the speech Codecs on both radio legs are identical.

establishes the TFO connection by:

- commanding the paths to go transparent;
- bypassing the decoder/encoder functions within the TRAUs.

guarantees a fast fall back procedure for sudden TFO interruption and **supports**:

- the resolution of Codec mismatch situations; and
- the cost efficient transmission within the fixed part of the network.

The Tandem Free Operation is **fully compatible** with existing GSM equipment. In its basic operation **it affects only TRAUs**. The additional computational complexity is small compared to the encoding/decoding functions of the TRAUs. Mobile Station, BTS, MSC and other network elements are not at all affected in this basic operation.

In an optional mode, the TFO supports the resolution of Codec mismatch situations, i.e. the situation where the Speech-Codecs at both radio-legs are different. For this, an additional communication channel between TRAU and BSS is necessary and the BSS has to perform a normal local intra cell handover to change the Codec type. That communication between TRAU and BSS is considered as manufacturer proprietary and not handled within this recommendation.

Once TFO functionality is implemented in TFO compatible TRAU equipment, it can be employed also for TFO connections to other systems, like ISDN phones, speech servers, Internet connections or connections to other systems, like UMTS.

4.2 Principle of Tandem Free Operation

The TRAU shall be controlled by the BTS when it is positioned remote from the BTS. In this case, the speech/data information and TRAU control signals shall be transferred between the BTS and the TRAU in frames denoted "TRAU Frames" on the **Abis** respectively **Ater** interface.

In Tandem Free Operation similar frames, denoted "**TFO Frames**", are transferred between the two TRAUs on the **A-interface** by inband signalling, i.e. inserting them into the PCM sample bit stream.

In the case of Half Rate speech traffic, these TFO Frames shall be carried by 8 kBit/s traffic channels mapped onto the least significant bit (LSB) of the PCM samples.

In the case of Full Rate and Enhanced Full Rate speech traffic, these TFO Frames shall be carried by 16 kBit/s traffic channels mapped onto the two least significant bits of the PCM samples.

Like TRAU Frames the TFO Frames have a fixed size (and length) of:

- 160 bits (20 ms) when the Half Rate Codec is used (see 06.20);
- 320 bits (20 ms) when the Full Rate Codec is used (see 06.10);
- 320 bits (20 ms) when the Enhanced Full Rate Codec is used (see 06.60).

Prior and parallel to these TFO Frames also other ***TFO Messages*** are transferred on the A-interface. TFO Messages conform to the IS_Message Principles described in Annexes A and B.

The TFO protocol between the TRAUs is independent of the position of the TRAUs within the GSM networks.

A possible configuration of two TRAUs is shown in Figure 1, which is intended as a reference model.

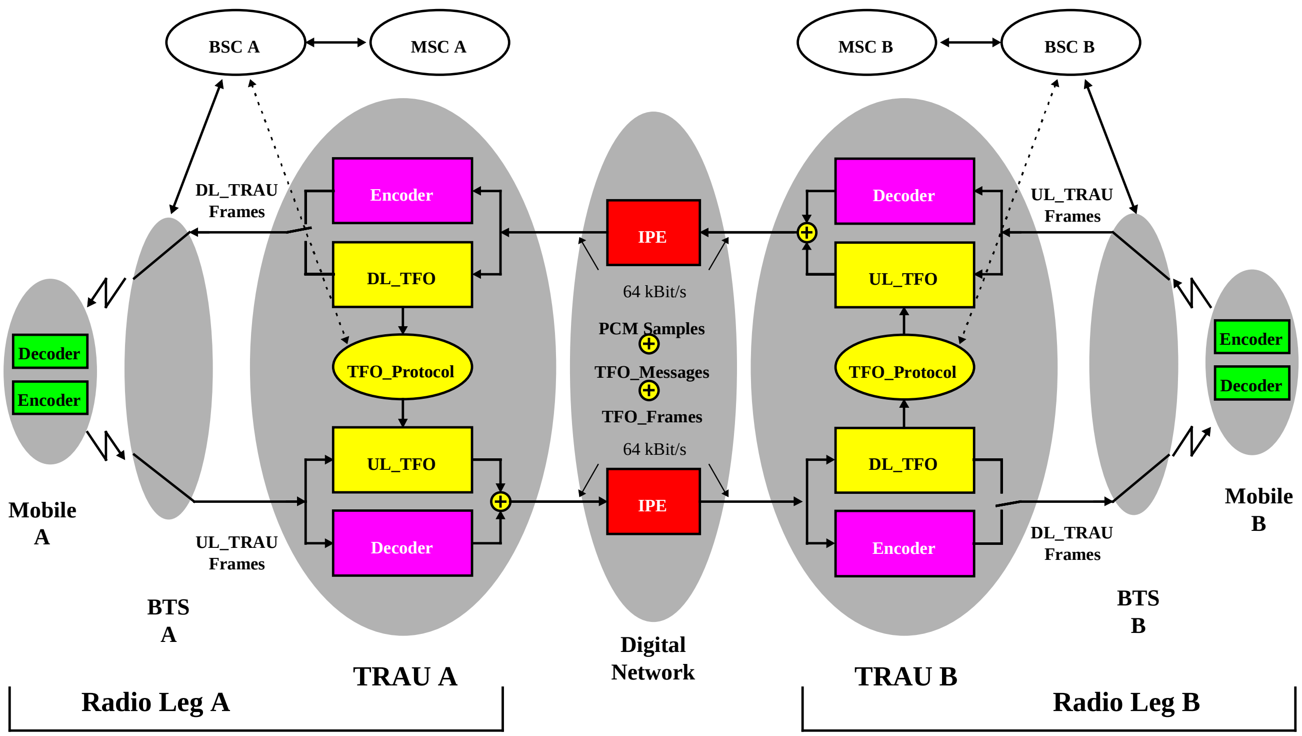

Figure 1: Functional Entities for Handling of Tandem Free Operation in MS-MS calls

TFO shall provide a virtually transparent digital channel from Encoder of Mobile A to Decoder of Mobile B and vice versa.

5 TFO Frame Structure

5.1 TFO Frames for Full Rate and Enhanced Full Rate

Full Rate (Enhanced Full Rate) TFO Frames are structured similar to **uplink** Full Rate (Enhanced Full Rate) TRAU Frames.

Table 1: The coding of the Control Bits (C1 .. C21) for Full Rate TFO Frames

Control Bit	Description	Comment
C1 - C4	**Frame Type** FR EFR	C1 C2 C3 C4 0 . 0 . 0 . 1 1 . 1 . 0 . 1 All other code words are reserved.
C5	**EMBED**	Indicates the presence of an embedded TFO Message
C6 - C11	spare	(is Time Alignment in TRAU frame)
C12	**BFI**	Copied from uplink TRAU frame
C13 - C14	**SID**	Copied from uplink TRAU frame
C15	**TAF**	Copied from uplink TRAU frame
C16	spare	
C17	**DTXd**	Copied from uplink TRAU frame
C18 - C21	spare	

Any spare control bits should be coded binary "1". They are reserved for future use and may change.

The **Synchronisation Pattern** is similar to the Synchronisation Pattern in 08.60, with some exceptions:

EMBED: C5 equal "0": the Synchronisation Pattern is exactly as described in 08.60;

C5 equal ”1”: the Synchronization Pattern is changed by embedding a TFO Message.

For the coding of the **Data Bits** see GSM 08.60.

For the coding of the **Time Alignment Bits** (T_Bits, T1.. T4) see GSM 08.60.

The T_Bits correspond normally to the T_Bits received in the up-link TRAU Frame.

For the purpose of this description the 320 bits of one TFO Frame are arranged in 40 rows (0..39),

with 8 bit (1..8: one octet) each (see GSM 08.60).

The bits of a Full Rate (Enhanced Full Rate) TFO Frame are transmitted in the following order:

Bit m of octet n, shall be transmitted in the **Least** Significant Bit of the

PCM sample k = n*4 + (m+1)/2 for m = (1, 3, 5, 7) and n = (0..39).

Bit m of octet n shall be transmitted in the **second Least** Significant Bit of the

PCM sample k = n*4 + m/2 for m = (2, 4, 6, 8) and n = (0..39).

PCM sample (k=1) is the first PCM sample of the corresponding decoded speech frame (k=(1..160)).

5.2 TFO Frame for Half Rate

Half Rate TFO Frames are always structured similar to **uplink** Half Rate TRAU Frames for **8 kBit/s** submultiplexing, see GSM 08.61 subclauses 5.2.1 and 5.2.4.1.

If Half Rate TRAU Frames with 16 kBit/s submultiplexing are used on the Abis/Ater interface, then the Control and Extended Control Bits for the 8 kBit/s TFO Frame need to be generated on basis of the received Control Bits from the TRAU Frame.

Table 2: Void

The coding of the **Control Bits** (C1 .. C9) is according to the following table 3:

Table 3: The coding of the Control Bits **(C1 .. C9)**

Control Bit	Description	Comment
C1 - C4	**Frame Type** HR	C1 . C2 . C3 . C4 0 . 0 . 0 . 1 All other code words are reserved.
C5	**EMBED**	Indicates the presence of an embedded TFO Message
C7 - C8	spare	
C9	**DTXd**	Copied from uplink TRAU frame

Any spare control bits should be coded binary "1". They are reserved for future use and may change.

The **Synchronisation Pattern** is similar to the Synchronisation Pattern in 08.61, with some exceptions:

EMBED: C5 equal "0": the Synchronisation Pattern is exactly as described in 08.61;

C5 equal ”1”: the Synchronization Pattern is changed by embedding a TFO Message.

The coding of the **Extended Control Bits (XC1.. XC6)**:

XC1 is copied from the uplink TRAU Frame.

XC2 .. XC6: These bits are normally copied from the 8 kBit/s TRAU frame corresponding to this TFO Frame.

All other codes are reserved.

For the coding of the **Data Bits** see GSM 08.61.

For the coding of the **Time Alignment Bits** see GSM 08.61.

The T_Bits correspond normally to the T_Bits received in the up-link TRAU Frame.

For the purpose of this description the 160 bits of one frame are arranged in 20 rows (1..20), with 8 bit (1..8: one octet) each (see GSM 08.61).

The bits of a Half Rate TFO Frame are transmitted in the following order:

Bit m of octet n shall be transmitted in the **Least** Significant Bit of the

PCM sample k = (n-1)*8+m; with m = (1..8) and n = (1..20).

PCM sample (k=1) is the first PCM sample of the corresponding decoded speech frame (k=(1..160)).

6 TFO Message Structure

Several TFO Messages are defined, based on the general IS_Message principle, as defined in Annex A.
Definition for ***Sender*** side:

TFO_REQ (): Identifies the source of the message as a TFO capable device, using a defined speech Codec_Type.
TFO_REQ contains the following parameters ():

- the System_Identification of the sender;
- the specific Local_Signature of the sender (e.g. TRAU or TCME);
- the Local_Used_Codec_Type at sender side;
- possibly additional attributes for the Local_Used_Codec_Type.

TFO_ACK (): Is the response to a TFO_REQ Message.
TFO_ACK contains the corresponding parameters to TFO_REQ, but the Local_Signature is replaced by the Reflected_Signature, copied from the received TFO_REQ Message.

TFO_REQ_L (): Is sent in case of Codec Mismatch or for sporadic updates of information.
TFO_REQ_L contains the following parameters ():

- the System_Identification of the sender;
- the specific Local_Signature of the sender (e.g. TRAU or TCME);
- the Local_Used_Codec_Type at sender side;
- the Local_Codec_List of alternative Codec_Types;
- possibly additional attributes for the alternative Codec_Types.

TFO_ACK_L (): Is the response to a TFO_REQ_L Message.
TFO_ACK_L contains the corresponding parameters to TFO_REQ_L, but the Local_Signature is replaced by the Reflected_Signature, copied from the received TFO_REQ_L Message.

TFO_REQ_P (): Is used to indicate during ongoing TFO that an other Codec_Type would be preferred.
TFO_REQ_P contains the following parameters ():

- the System_Identification of the sender;
- the specific Local_Signature of the sender;
- the Preferred_Codec_Type at sender side (only used by TCME);
- possibly additional attributes for the Preferred_Codec_Type.

TFO_TRANS (): Commands possible IPEs to let the TFO Frames pass transparently within the LSB (8 kBit/s) or the two LSBs (16 kBit/s). TFO_TRANS contains the following parameter ():

- the Local_Channel_Type (8 kBit/s or 16 kBit/s).

TFO_NORMAL: Commands possible IPEs to revert to normal operation.
TFO_NORMAL has no parameters.

TFO_DUP: Informs the distant partner that TFO Frames are received, while still transmitting PCM samples.
TFO_DUP has no parameters.

TFO_SYL: Informs the distant partner (if still possible) that TFO Frames are no longer received.
TFO_SYL has no parameters.

TFO_FILL: Message without specific meaning, used to pre-synchronise IPEs or to bridge over gaps in TFO protocols. TFO_FILL has no parameters.

6.1 Extendibility

A mechanism for future extensions is defined in a way that existing implementations in the field shall be able to ignore future, for them unknown Codec_Types and their potential attributes. The existing implementations shall be able to decode the reminder of the messages (which is known to them) uncompromised. This mechanism allows to extent:

- the number of Local_Used_Codec_Types from 15 (short form) up to 255 (long form) for one System_Identification;
- the Codec_List;
- the Codec_Attributes (if needed).

In case of the TFO_REQ or TFO_ACK messages the attributes of the Local_Used_Codec_Type shall be sent in the Codec specific way, without a preceding Codec_Attribute_Head Extension_Block. Existing equipment, that do not know a future Codec_Type and therefore do not know if and how many attribute Extension_Blocks do follow, shall skip these Extension_Blocks, until they find a TFO Message Header again.

In case of the TFO_REQ_L or TFO_ACK_L Messages the simple Codec_List shall be sent immediately after the SIG_LUC and possible Codec_x Extension_Blocks. Then the attributes of all alternative Codec_Types shall follow. Each set of Codec attributes shall be preceded by the Codec_Attribute_Head Extension_Block (with Codec_Type Identifier and Length Indicater) followed by the Codec specific attributes.

TFO_REQ_P shall not contain the list of alternative Codecs, i.e. it shall be based on TFO_REQ and not on TFO_REQ_L.

6.2 Regular and Embedded TFO Messages

A TFO Message is called **"regular"**, if it is sent inserted into the PCM sample stream. A TFO Message is called "**embedded**", if it is sent together with (embedded into) TFO Frames, see also subclause 7.2. The bit stealing scheme (see Annex A) is identical for regular and embedded TFO Messages. Control bit C5 marks redundantly (in general) all TFO Frames that are affected by embedding a TFO Message. Due to the specific construction of the TFO Messages, they replace some of the synchronisation bits of the TFO Frames. TFO Frame synchronisation is in case of embedded TFO Messages therefore different, however, not endangered. Data and other control bits of the TFO Frames are not affected by embedded TFO Messages.

6.3 Cyclic Redundancy Check

The Extension_Blocks, defined in the following sub-clauses, shall be protected by three CRC parity bits. These shall be generated as define in GSM 08.60 for the Enhanced Full Rate. For simplicity this specification is reprinted here:

“These parity bits are added to the bits of the subset, according to a degenerate (shortened) cyclic code using the generator polynomial:

$$g(D) = D^3 + D + 1$$

The encoding of the cyclic code is performed in a systematic form which means that, in GF(2), the polynomial:

$$d(m)D^n + d(m+1)D^{n-1} + ...+ d(m + n-3)D^3 + p(0)D^2 + p(1)D + p(2)$$

where p(0), p(1), p(2) are the parity bits, when divided by g(D), yields a remainder equal to:

$$1 + D + D^2$$

For every CRC, the transmission order is p(0) first followed by p(1) and p(2) successively.”

In case of Extension_Blocks p(0)..p(2) are mapped to bits 16..18.

6.4 Definition of the TFO_REQ Messages

Symbolic Notation:
TFO_REQ (Sys_Id, LSig, Local_Used_Codec_Type[, Used_Codec_Attributes]).

TFO_REQ_L (Sys_Id, LSig, Local_Used_Codec_Type, Codec_List [, Alternative_Codec_Attributes]).

TFO_REQ_P (Sys_Id, LSig, Preferred_Codec_Type[, Preferred_Codec_Attributes]).

The TFO_REQ Messages conform to the IS_REQ Message, defined in the Annex A, with IS_System_Identification, followed by the SIG_LUC Extension_Block, optionally the Codec_x Extension_Block, the Codec_List Extension_Block(s) and the Codec_Attribute Extension_Blocks.

The shortest TFO_REQ takes 140 ms for transmission, see Figure 2a.
The shortest TFO_REQ_L takes 180 ms (Figure 2b).
The shortest TFO_REQ_P takes 180 ms for transmission (Figure 2c).

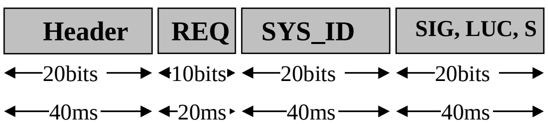

Figure 2a: Construction of the shortest possible TFO_REQ Message

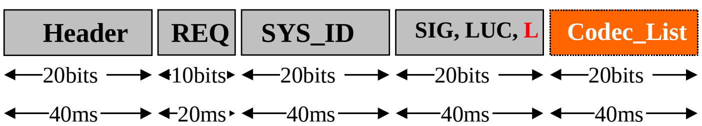

Figure 2b: Construction of the shortest possible TFO_REQ_L Message

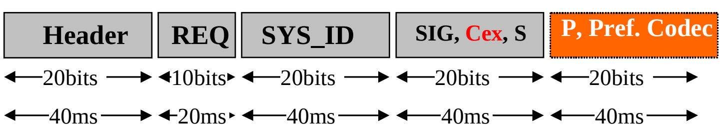

Figure 2c: Construction of the shortest possible TFO_REQ_P Message

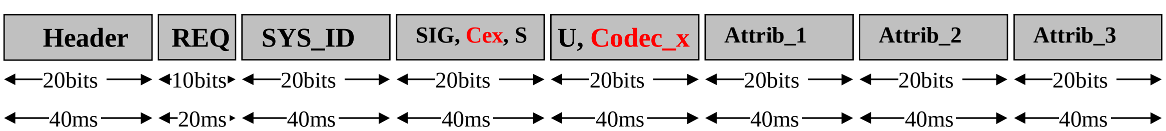

Figure 2d:Example of a TFO_REQ Message with a Codec with an index higher than 15 and with three Attribute Extension_Blocks (300 ms length)

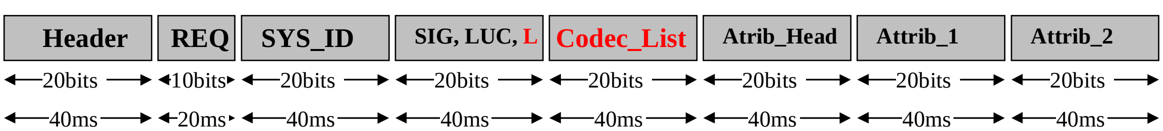

Figure 2e: Example of a TFO_REQ_L Message with Codec_List and one alternative Codec with two Attribute Extension_Blocks (300 ms length)

6.4.1 Definition of the SIG_LUC Extension_Block

The SIG_LUC Extension_Block consists of 20 bits, as defined in Table 4. It shall follow always immediately after the SYS_ID Extension_Block. It differentiates between TFO_REQ and TFO_REQ_L messages, respectively between TFO_ACK and TFO_ACK_L messages.

In case of a TFO_REQ_P message it shall be followed immediately by the Codec_x Extension_Block (Table 5).

The Codec_x Extension_Block shall also be used if the Local_Used_Codec_Type has a CoID higher than 14.

Table 4: SIG_LUC Extension_Block

Bit	**Description**	**Comment**
Bit 1	"**0**"	normal IS-Message Sync Bit, constant.
Bit 2	**List_Ind**	Indicates, whether the Codec_List is included in the TFO Message or not **0: S: TFO_REQ or TFO_REQ_P** or **TFO_ACK**: Codec_List is not included (short) **1: L: TFO_REQ_L** or **TFO_ACK_L**: Codec_List is included (long)
Bit 3..10	**Sig**	An 8-bit random number to facilitate the detection of circuit loop back conditions and to identify the message source
Bit 11	"**0**"	normal IS-Message Sync Bit, constant
Bit 12.. 15:	**Codec_Type** **CoID_s** (short form)	Identifies the Local_Used_Codec_Type, which is currently used by the sender **0000…1110: reserved for 15 Codec_Types** **1111: Codec_X** Extension_Block follows immediately, e.g. for "Preferred Codec" Type (Codec_Type, long form)
Bit 16..18:	**CRC**	3 CRC bits protecting Bits 2 to 10 and 12 to 15
Bit 19..20:	**EX** EX == "0.0" EX == "1.1"	The normal 2 bits for IS_Message Extension. No other extension block follows An other extension block follows

6.4.2 Definition of the Codec_x Extension_Block

The Codec_x Extension_Block consists of 20 bits, as defined in Tabel 5. It shall follow always immediately after the SIG_LUC Extension_Block, if the Codec_Type field is set to "1111".

Table 5: Codec_x Extension_Block

Bit	**Description**	**Comment**
Bit 1	"**0**"	normal IS-Message Sync Bit, constant.
Bit 2	**Codec_Sel**	Differentiates the Codec_x Extension_Block **0: U: Used_Codec_Type** is defined in Codec_Type_x field **1: P: Preferred_Codec_Type** is defined in Codec_Type_x field Note: The Preferred_Codec_Type is only defined in TFO_REQ Messages. It is reserved for future use in TFO_ACK messages.
Bit 3..10	**Codec_Type_x** **CoID** (long form)	Identifies the Local_Used_Codec_Type, which is currently used by the sender **0000.0000 … 1111.1111 reserved for 255 Codec_Types** 0000.1111 is undefined and shall not be used.
Bit 11	"**0**"	normal IS-Message Sync Bit, constant
Bit 12.. 15:	"1010"	reserved for future use, set to "1010" to minimise audible effects
Bit 16..18:	**CRC**	3 CRC bits protecting Bits 2 to 10 and 12 to 15
Bit 19..20:	**EX**	The normal 2 bits for IS_Message Extension. 00: No other extension block follows 11: An other extension block follows

6.4.3 Definition of the Codec_List_Extension_Block

The Codec_List Extension Block consists of 20 bits, as defined in Table 6. It identifies the Codec_Types that are supported by the sender, respectively the BSS subsystem, including the mobile station and the radio resource at sender side. The Codec_List must at least contain the Local_Used_Codec_Type. If a system supports more than 12 Codec_Types, then possibly other Codec_List Extension_Blocks (Table 7) may follow.

Table 6: Codec_List Extension Block

Bit	Description	Comment
Bit 1	"**0**"	normal IS-Message Sync Bit, constant.
Bit 2..10	**Codec_List_1**	First part of Codec_List. For each Codec_Type one bit is reserved. If the bit is set to "0" then the specific Codec_Type is not supported; if the bit is set to "1" then the specific Codec_Type could be used.
Bit 11	"**0**"	normal IS-Message Sync Bit, constant
Bit 12.. 14:	**Codec_List_2**	Second part of the Codec_List; All three bits are reserved for future Codec_Types (up to Codec_Type 12)
Bit 15	**Codec_List_x**	If set to "1" a further Codec_List Extension_Block follows; otherwise set to "0"
Bit 16..18:	**CRC**	3 CRC bits protecting Bits 2 to 10 and 12 to 15
Bit 19..20:	**EX**	The normal 2 bits for IS_Message Extension: 00: No other extension block follows 11: An other extension block follows

Table 7: Further Codec_List Extension Block(s)

Bit	Description	Comment
Bit 1	"**0**"	normal IS-Message Sync Bit, constant.
Bit 2..10	**Codec_List_1x**	First part of Codec_List. For each Codec_Type one bit is reserved. If the bit is set to "0" then the specific Codec_Type is not supported; if the bit is set to "1" then the specific Codec_Type could be used. Bit 2: Codec_Type 13 (+ x*12; x=1..2..3) Bit 4: Codec_Type 14 (+ x*12; x=1..2..3) and so on
Bit 11	"**0**"	normal IS-Message Sync Bit, constant
Bit 12.. 14:	**Codec_List_2x**	Second part of the Codec_List;All three bits are reserved for future Codec_Types (up to Codec_Type 24 (+x*12; x=1..2..3)
Bit 15	**Codec_List_xx**	If set to "1" a further Codec_List Extension_Block follows; otherwise set to "0"
Bit 16..18:	**CRC**	3 CRC bits protecting Bits 2 to 10 and 12 to 15
Bit 19..20:	**EX**	The normal 2 bits for IS_Message Extension: 00: No other extension block follows 11: An other extension block follows

6.4.4 Definition of the Codec_Attribute_Head Extension_Block

The Codec_Attribute_Head Extension_Block (Table 8) shall precede the Codec Attribute Extension_Blocks of a Codec_Type, if this Codec_Type needs additional attributes. Then this Codec_Attribute_Head identifies the Codec_Type and the number of additional Extension_Blocks for it.

Table 8: Codec_Attribute_Head Extension_Block

Bit	Description	Comment
Bit 1	"**0**"	normal IS-Message Sync Bit, constant.
Bit 2	**PAR_Sel**	Differentiates this Extension_Block **0:** Parameters included in **PAR** field: Simple Codec_List_Extension **1:** Length Indicator (**LI**) included: Parameters follow in subsequent Extension_Blocks
Bit 3..10	**CoID**	This field identifies the Codec_Type for which the subsequent attributes are valid. The same coding as in the Codec_x Extension_Block is used (long form)
Bit 11	"**0**"	normal IS-Message Sync Bit, constant
Bit 12.. 15:	**LI / PAR**	If Par_Sel==1: LI: Length Indicator: 0000: reserved; 0001: one other Extension_Block follows, etc. If Par_Sel==0: PAR: Codec specific definition of these four bits
Bit 16..18:	**CRC**	3 CRC bits protecting Bits 2 to 10 and 12 to 15
Bit 19..20:	**EX**	The normal 2 bits for IS_Message Extension: 00: No other extension block follows 11: An other extension block follows

Note : this Extension_Block shall be used for the codecs introduced in the future that need attributes. It shall precede the Attribute Extension_Blocks. This allows earlier versions to skip the blocks they do not understand. It shall not be used for the FR, HR and EFR Codec_Types.

6.5 Definition of the TFO_ACK Messages

Symbolic Notation:
TFO_ACK (Sys_Id, RSig, Local_Used_Codec_Type [, Used_Codec_Attributes])

TFO_ACK_L (Sys_Id, RSig, Local_Used_Codec_Type, Codec_List [, Alternative_Codec_Attributes]).

TFO_ACK_P: undefined, reserved for future use.

The TFO_ACK Messages conform to the IS_ACK Message, defined in the Annex A, with IS_System_Identification, followed by the SIG_LUC Extension_Block, optionally the Codec_x Extension_Block, the Codec_List Extension_Block(s) and the Codec_Attribute Extension_Blocks.

TFO_ACK and TFO_REQ Messages differ only in the ACK / REQ Command block and the construction of the Signature: Local_Signature in case of TFO_REQ, Reflected_Signature in case of TFO_ACK. All extension blocks defined for the TFO_REQ are valid as well for TFO_ACK.

The shortest TFO_ACK takes 140 ms for transmission.
The shortest TFO_ACK_L takes 180 ms.

6.6 Definition of the TFO_TRANS Messages

Symbolic Notation: TFO_TRANS (Channel_Type).

Two TFO_TRANS Messages are defined in conformity to the IS_TRANS Messages in Annex A.
For 8 kBit/s submultiplexing the "**TFO_TRANS (8k)**" is used and is identical to "IS_TRANS_1_u".
For 16 kBit/s submultiplexing the "**TFO_TRANS (16k)**" is used and is identical to "IS_TRANS_2_u".

TFO_TRANS() takes 100 ms for transmission.

In most cases the respective TFO_TRANS Message shall be sent twice: once as a regular TFO Message, exactly before any series of TFO Frames, and once embedded into the first TFO Frames, see clause 10.

6.7 Definition of the TFO_NORMAL Message

Symbolic Notation: TFO_NORMAL.

The TFO_NORMAL Message is identical to the IS_NORMAL Message defined in the Annex A.

It shall be sent at least once whenever an established tandem free operation need to be terminated in a controlled way.

TFO_NORMAL takes 100 ms for transmission.

6.8 Definition of the TFO_FILL Message

Symbolic Notation: TFO_FILL.

The TFO_FILL Message is identical to the IS_FILL Message, defined in the Annex A.

TFO_FILL may be used to pre-synchronise IPEs. Since IS_FILL is one of the shortest IS Messages, this is the fastest way to synchronize IPEs, without IPEs swallowing other protocol elements. By default three TFO_Messages shall be sent at the beginning; this number may be, however, configuration dependent.

One TFO_FILL takes 60 ms for transmission.

6.9 Definition of the TFO_DUP Message

Symbolic Notation: TFO_DUP

The TFO_DUP Message is identical to the IS_DUP Message, defined in Annex A.

TFO_DUP informs the distant TFO Partner, that TFO Frames have been received unexpected, e.g. during Establishment. This enables a fast re-establishment of TFO after a ***local*** handover.

TFO_DUP takes 60 ms for transmission.

6.10 Definition of the TFO_SYL Message

Symbolic Notation: TFO_SYL

The TFO_SYL Message is identical to the IS_SYL Message, defined in Annex A.

TFO_SYL informs the distant TFO Partner, that tandem free operation has existed, but suddenly no TFO Frame were received anymore. This enables a fast re-establishment of TFO after a ***distant*** handover.

TFO_SYL takes 60 ms for transmission.

6.11 Specification of the TFO Messages for GSM

6.11.1 The GSM Codec_Types

The GSM Codec_Types are defined in long form (Codec_Type_x, CoID) as
Bit 3…Bit10 (Codec_Type_x):

CoID	**Codec_Type**
0000.0000:	GSM Full Rate
0000.0001:	GSM Half Rate
0000.0010:	GSM Enhanced Full Rate
0000.0011:	reserved for GSM Adaptive Multi-Rate – FR (TCH/F)
0000.0100:	reserved for GSM Adaptive Multi-Rate – HR (TCH/H)
other codes:	reserved for future use.

The short form (CoID_s) exists for all Codec_Types with indices below 15 and consists of the last four bits of the long form (CoID).

6.11.2 The GSM Codec_List

For GSM the Codec_List is defined as:

	Codec_Type
Bit 2:	GSM Full Rate
Bit 3:	GSM Half Rate
Bit 4:	GSM Enhanced Full Rate
Bit 5:	reserved for GSM Adaptive Multi-Rate – FR (TCH/F)
Bit 6:	reserved for GSM Adaptive Multi-Rate – HR (TCH/H)

The remaining bits are reserved for future Codec_Types.

6.11.3 The GSM Codec_Type Attributes

The GSM Codec_Types Full Rate, Half Rate and Enhanced Full Rate do not need additional attibutes. They are fully defined by their System_Identification (see Annex A.5) and Codec_Type.

7 Time Alignment of TFO Frames and TFO Messages

The time alignment procedures for the downlink TRAU Frames, as specified in GSM 08.60 (full rate traffic) and GSM 08.61 (half rate traffic) on the Abis/Ater interface, are not affected by the TFO procedures on the A interface. The relative TRAU Frame phase positions of the two TRAUs using TFO across the A interface are arbitrary and depend on the local timing structure of the relevant BTSs. This is not changed by the TFO Protocol.

TFO Frames and embedded TFO Messages are always exactly aligned with each other and follow the uplink TRAU Frames with a small, neglegible, constant delay (Tultfo: some PCM samples).

7.1 Time Alignment of TFO Messages

At start up of the TFO Protocol the first regular TFO Message is aligned to an uplink TRAU Frame in the same way as a TFO Frame, respectively an embedded TFO Message would be aligned (see subclause 7.2). Then, after that, all regular TFO Messages follow contiguously, without any phase shift in time alignment, until the first TFO Frame needs to be sent (in general after the TFO_TRANS Message). Then the necessary number of T_Bits (if any) is inserted before the first TFO Frame, see subclause 7.2. Consequently all following, embedded TFO Messages are always aligned with the TFO Frames in a way, that the first bit of any TFO Messages is placed into the LSB of the first sample of a TFO Frame. Due to this definition, embedded TFO Messages only modify some of the synchronisation bits of the TFO Frames and control bit C5.

7.2 Time Alignment of TFO Frames

The contents of the Uplink TRAU Frame, received from the BTS via the Abis/Ater Interface, undergo the small, constant delay (Tultfo) required to perform the modifications of the C5 and Sync bits, before being forwarded to the other TRAU over the A Interface as TFO Frame. Since this delay is substantially smaller than the delay for the decoded speech signal, the TFO Frames preceed the corresponding speech samples. Figure 4 shows the relations. Please note that no exact delay value for Tultfo is defined or need to be defined.

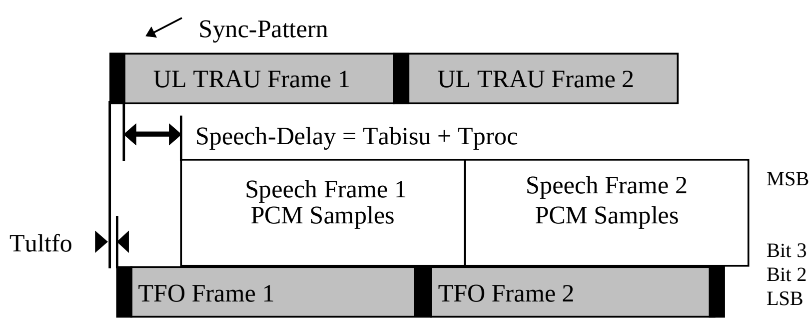

Figure 4: Uplink TFO Frame Time Alignment

On the transition between the sending of regular TFO Messages and the first TFO Frame on the A interface, a sufficient number (up to a maximum of 159) of Time Alignment Bits, also called "T_Bits", are inserted into the LSBs of the PCM samples to align the TFO Frame as described above.

This insertion of Time Alignment Bits (if necessary) is started exactly with the 16th PCM sample after the last bit of the last regular TFO Message (i.e. the TFO_TRANS Message).

Whenever, in a later stage, the phase of the uplink TRAU Frame changes, then again T_Bits need to be inserted between two consecutive TFO Frames or deleted from the tail of the last TFO Frame to ensure proper alignment.

The insertion of T_Bits as a result of timing changes shall occur ***between*** TFO Frames and not within TFO Frames.

If the time alignment is necessary while a TFO Message is embedded into a series of TFO Frames, then the TFO Message may be cut into two parts with the T_Bits in between. Therefore, whenever an adjustment of the phase of the TFO Frames is necessary, then one additional TFO Message shall be embedded into the next TFO Frames (after the possibly ongoing TFO Message). If nothing else is to be transmitted, then the TFO_FILL Message shall be used. One TFO_TRANS Message is ***always*** embedded into the first TFO Frames. See the following Figure 5:

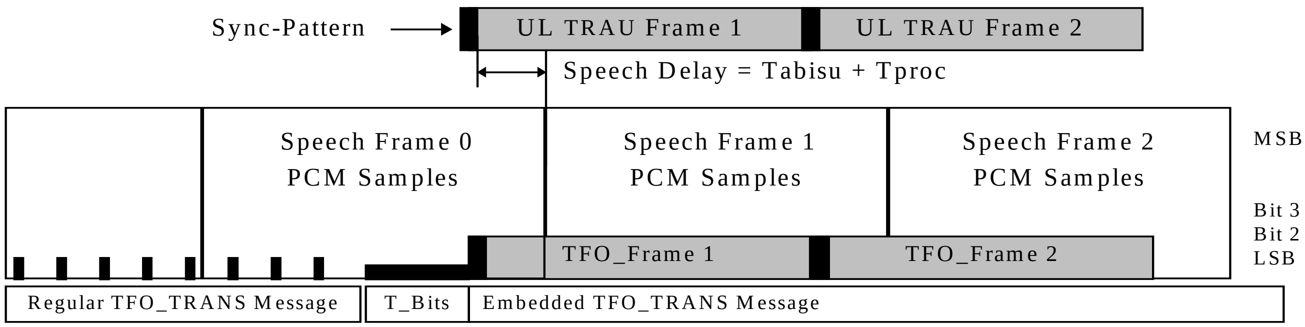

Figure 5: Time Alignment by inserting T_Bits and embedding one TFO_TRANS Message

7.3 Time Alignment of TFO Frames to Downlink TRAU Frames

The phase position of the downlink TRAU Frames is not affected by the TFO Protocol.

The phase difference between the received TFO Frames and the downlink TRAU Frames is in general constant, but arbitrary between 0 and 159 PCM samples. The time alignment of the TFO Frames to the downlink TRAU Frames must therefore be managed by buffering the TFO Frames within the receiving downlink TRAU. This can be done in one of two methods:

The received TFO Frame is buffered for a period between 0 to 159 PCM samples in addition to the processing delay *(Tbfh)* required *to perform a suitable Bad Frame Handling on parameter level.* Transmission of the downlink TRAU Frame may in this case begin ***prior*** to receipt of the complete TFO Frame.

NOTE 1: In this first method the overall one way signal delay will be between *30 ms and 10 ms lower than the delay* in normal tandem connections.

Alternatively the received TFO Frame is buffered for a period between 160 to 319 PCM samples in addition to the processing delay required *to perform a suitable Bad Frame Handling on parameter level (Tbfh).* Transmission of the downlink TRAU Frame will in this case always begin ***after*** the receipt of the complete TFO Frame.

NOTE 2: In this second method the overall one way signal delay will always *be up to 10ms lower or up to 10 ms higher* than the delay in normal tandem connections.

NOTE 3: The two methods differ in one way signal delay always by exactly 20 ms. Figure 6 highlights the relations for an arbitrarily selected relative phase difference between TFO and TRAU Frames of 80 samples (10 ms). *Tbfh is in the order of some PCM samples only, if error concealment is done "in advance" based on the parameters of the previous TFO Frame, before the actual TFO Frame is even received.*

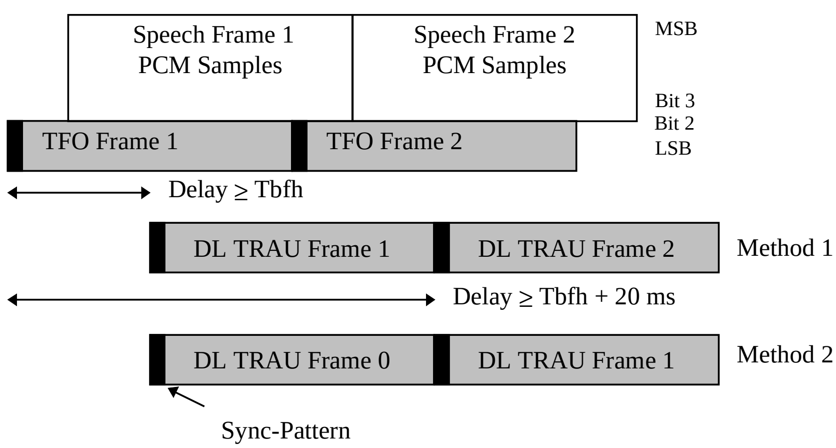

Figure 6: Downlink Time Alignment of TFO Frames

8 Processes for Tandem Free Operation

The following chapters describe the actions within the TRAU to establish and maintain Tandem Free Operation in terms of a State Machine, respectively TFO Processes, handling synchronisation and protocol. The description of the TFO Protocol does not reflect implementation details for the I/O Processes, but they may need to be considered for the exact understanding of the behaviour. Only the TFO_Protocol Process is detailed, which is responsible for the handling of the TFO Protocol.

The SDL-Simulation, as described in Annex C, however, takes the necessary details into account and can serve as example implementation for all processes, as far as the TFO Protocol is concerned.

The TFO_TRAU can be regarded as consisting of five processes, which are strongly coupled to each other, which run in parallel, but phase shifted. The TFO_Protocol Process communicates with the TFO I/O processes and, optionally, with its corresponding process within the BSS (BSC) to resolve Codec Mismatch, see figure 7.

Under normal circumstances (exceptions occur during time alignments or octet slips) all TFO I/O Processes are triggered every 160 samples or every speech frame of 20 ms. All events and actions are quantized in time into these smallest intervals.

It can be assumed that the processing times for the TFO Processes are very short and negligible.

However, it must be ensured that no timing ambiguity occurs between the Processes.

This means the processing and exchange of information between them do not overlap in time. Care must be taken especially when time alignment occurs, which may be completely independent in uplink and downlink.

During these time alignments the TFO Frames or TFO Messages may shift in time and consequently the triggering point for the related TFO Processes changes, too.

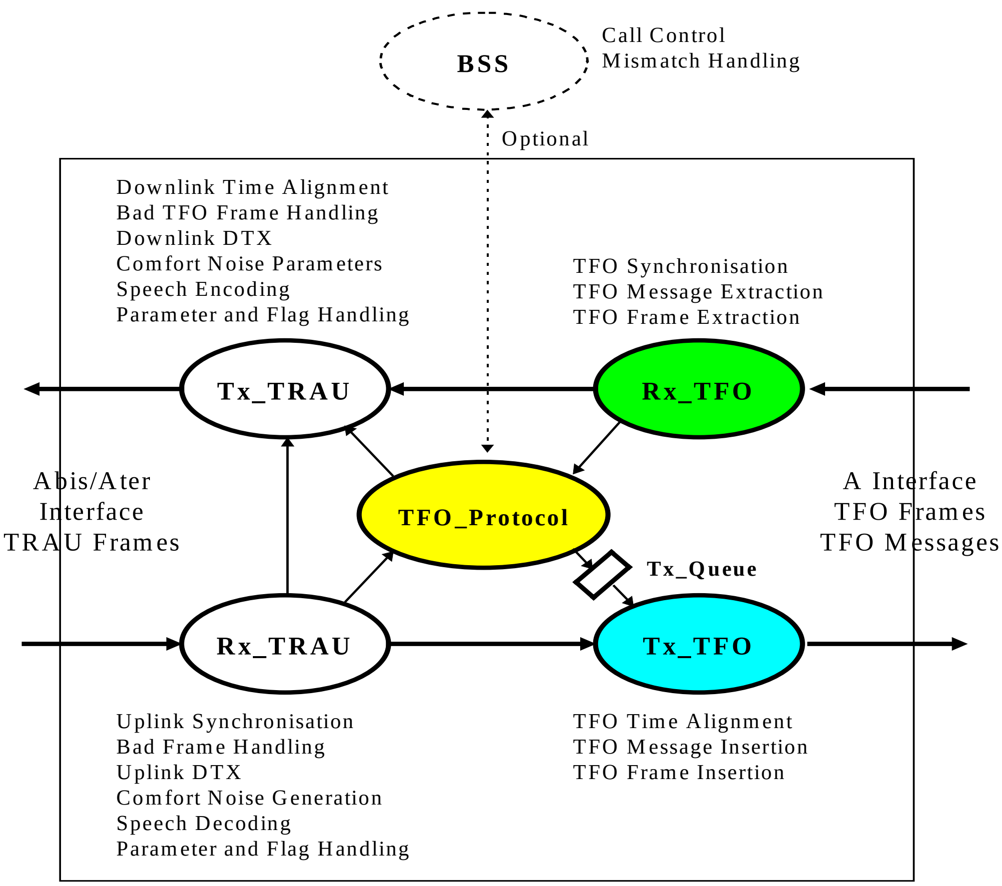

Figure 7: The TFO_TRAU consists of five Processes

8.1 Rx_TRAU Process

The Rx_TRAU Process receives Uplink TRAU Frames from the Abis/Ater Interface and synchronises to them, i.e. checks correct synchronisation and contents. It performs all actions of a conventional Uplink TRAU (see GSM 08.60 respectively GSM 08.61): It extracts the data bits and calls, if appropriate, the Bad Frame Handler, the Uplink DTX functions and Comfort Noise Generator and finally the Speech Decoder.

The resulting speech samples are handled to the Tx_TFO Process for output to the A interface. In addition Rx_TRAU passes the Uplink TRAU Frames directly and unaltered to Tx_TFO.

It further extracts the control bits, respectively commands, from the Uplink TRAU Frames and sends corresponding Rx_TRAU Messages to the Tx_TRAU Process (see GSM 08.60 respectively GSM 08.61) and the TFO_Protocol Process (see subclause 8.5).

8.2 Tx_TRAU Process

The Tx_TRAU Process builds autonomously the relevant Downlink TRAU Frames and sends them in the correct phase relation onto the Abis/Ater-Interface as commanded by the time alignment from the BTS.

Tx_TRAU has two major States: TFO == OFF (default at beginning) and TFO == ON, see Figure 8.

Toggling between these two States is commanded by TFO_Protocol with Accept_TFO, respectively Ignore_TFO.

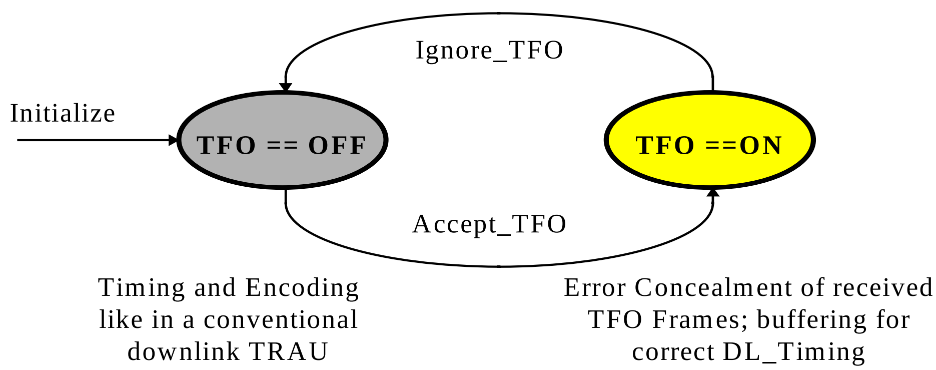

Figure 8: States of the Tx_TRAU Process

During TFO == OFF, Tx_TRAU performs all actions of a conventional downlink TRAU (see GSM 08.60 respectively GSM 08.61): On command from Rx_TRAU it performs necessary downlink time alignments and starts or stops sending of TRAU Frames. It samples one frame of speech samples in the correct phase position and calls the Speech Encoder. The resulting speech parameters are then transmitted downlink on the Abis/Ater interface.

During TFO == ON, Tx_TRAU performs Bad Frame Handling and Comfort Noise Parameter Handling on parameter level on the received TFO Frames, if necessary. The resulting speech parameters and control bits are buffered until they are passed as Downlink TRAU Frames in correct phase position to the BTS (see also subclause 7.3).

There are four possible cases regarding DTX in a Mobile-to-Mobile communication, as reflected in Table 7.

Table 7: DTX configurations in Mobile-To-Mobile communications

Case	Local TRAU: Downlink	Distant TRAU: Uplink
0	No-DTX	No-DTX
1	No-DTX	DTX
2	DTX	DTX
3	DTX	No-DTX

8.2.1 Downlink Speech Transmission if TFO is ON

During TFO == ON and if neither Uplink nor Downlink DTX is active (case 0 in Table 7), the Tx_TFO Process receives TFO Frames from the A Interface with SID == "0". It synchronises to them, i.e. checks correct synchronisation and contents. It extracts the data bits and calls, if appropriate (e.g. if BFI == "1" or if the TFO Frame is not-valid, see subclause 8.4.2), a Bad Frame Handler to derive suitable parameters for Downlink TRAU Frames. This Bad Frame Handler on parameter level is subject to manufacturer dependent future improvements and is not part of this recommendation.

During TFO == ON and if distant Uplink DTX is active, but not local Downlink DTX (case 1 in Table 7), then the Tx_TFO Process receives TFO Frames containing speech parameters (SID == "0": handling as in case 0, see above), but also TFO Frames containing SID parameters (SID == "1" or "2") and TFO Frames marked with BFI == "1" during speech inactivity. Tx_TFO then calls a Comfort Noise Generator to derive suitable "speech" parameters for Downlink TRAU Frames. The SP flag shall always be set to SP = "1". The Downlink TRAU Frames shall not contain the SID codeword, but parameters that allow a direct decoding. Also this Comfort Noise Generator on parameter level is subject to manufacturer dependent future improvements and is not part of this recommendation.

8.2.2 DTX Procedures in Downlink Direction if TFO is ON

During TFO == ON and if distant Uplink DTX and local downlink DTX are active (case 2 in Table 7), then the Tx_TFO Process receives TFO Frames containing either Speech parameters (SID == "0, handling see subclause 8.2.1) or SID parameters (SID == "1" or "2") or TFO Frames marked with BFI == "1" during speech inactivity due to transmission errors.

If a TFO Frame marked as a valid SID frame (SID == "2", BFI == "0") is received, it shall be stored in Tx_TRAU and its parameters shall be sent directly as Downlink TRAU SID Frame with correct timing. The DL_TRAU SID Frames produced from the valid stored frame are output repeatedly to the Abis/Ater interface whilst invalid SID frames (SID == "1") or frames marked as bad (BFI == "1") are received. These Downlink TRAU SID Frames shall be marked with the SP flag = "0" and shall all contain the SID codeword.

The stored SID Frame shall be considered as being valid for SID frame generation purposes until the receipt of the second instance of TAF == "1" (in a TFO Frame) following its initial storage. On expiry of the stored SID frame a suitable Bad Frame Handler for SID Frames shall be invoked to mute the Comfort Noise. Also this Bad Frame Handler for SID Frames on parameter level is subject to manufacturer dependent future improvements and is not part of this recommendation.

During TFO == ON and if distant Uplink DTX is not active, but local downlink DTX is on (case 3 in Table 7), i.e. only TFO Frames containing speech parameters are received , then one of the following alternative methods shall be used:

Downlink DTX need not to be used.
The speech parameters are extracted from the TFO Frames and are passed to the BTS. This is virtually identical to case 0 in Table 7, with no speech pauses detected, and handled like described above.

Alternatively, a voice activity detector makes the decision as to whether the frame contains speech or not based on the PCM samples received from the A interface. During periods decided as "Active Speech" the TFO Frame parameters are used as described above. During periods of "Speech Pause" Comfort Noise Parameters are calculated.

These operations are manufacturer dependent and not detailed here.

Alternatively, decoding of the speech parameters received in TFO Frames may be undertaken and these PCM samples may be used for normal downlink VAD and DTX functions.

8.2.3 Synchronisation and Bit Errors in Received TFO Frames

If Rx_TFO detects an error in the received TFO Frame synchronization or control bits or if a CRC error is detected, and the error is detected **prior** to beginning the output of the same frame (as a Downlink TRAU Frame), then Tx_TRAU shall either substitute parameters from the last good TFO Frame, or shall encode the received PCM samples for the duration of this frame.

If Rx_TFO detects an error in the received TFO Frame synchronization or control bits or if a CRC error is detected, and the error is detected **after** beginning of the output of the same frame (as a Downlink TRAU Frame), then Tx_TRAU shall deliberately corrupt the remaining, still unsent synchronization bits by setting them all to "0" and deliberately shall corrupt the remaining CRC bits. This will result in the BTS discarding this TRAU Frame, and transmitting a Fill frame to the Mobile station (see GSM 08.60 and GSM 08.61). The effect of the frame error will subsequently be masked by the Mobile station's handling of bad frames.

8.3 Tx_TFO Process

The Tx_TFO Process gets directly the unaltered Uplink TRAU Frames (containing the speech parameters and the control bits) and the decoded speech PCM samples from Rx_TRAU. It further gets internal messages (commands) from TFO_Protocol via the Tx_Queue.

Tx_TFO has two major States: TFO == OFF (default at beginning) and TFO == ON, see Figure 9.

Toggling between these two States is commanded by TFO_Protocol with Begin_TFO respectively Discontinue_TFO.

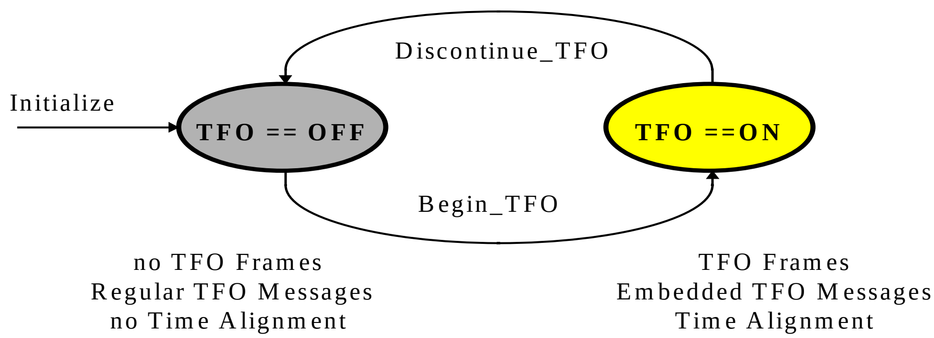

Figure 9: States of the Tx_TFO Process

During TFO == OFF, decoded speech PCM samples and regular TFO Messages (if any) are sent onto the A interface. Time Alignment takes place only once, just before the beginning of the first regular TFO Message.

During TFO == ON, TFO Frames and embedded TFO Messages (if any) are sent. Time Alignment takes place just before the first TFO Frame and then whenever required in between two TFO Frames.

The Tx_TFO Process builds the relevant TFO Frames and sends them in the correct phase relation onto the A-Interface. Time alignment of TFO Messages and TFO Frames are handled autonomously and independent of the TFO_Protocol Process. Rx_TRAU informs Tx_TFO about any changes in the phase position of the Uplink TRAU Frame and Tx_TFO inserts automatically the correct number of T_Bits before the next TFO Frame, and embeds autonomously the next TFO_Message or a TFO_FILL Message, if necessary.

The TFO_Protocol Process can send internal messages into the **Tx_Queue** (First In, First Out). Tx_TFO shall take the message from the Tx_Queue one by one, shall process them autonomously and shall send the resulting TFO Messages in correct order and phase position, as regular or as embedded TFO Messages.Tx_TFO shall generate a Runout Message to TFO_Protocol, if the last TFO Message is sent without guarantee of a direct continuation by another TFO Message, i.e. if the (possible) IPEs may have run out of synchronisation (see chapter 10). TFO_Protocol may delete messages from Tx_Queue, as long as they are in there:
Command "Clear Tx_Queue", at time *Tc*.

Basically, messages or commands that are already in processing by Tx_TFO at *Tc* can not be stopped, deleted or interrupted. The TFO Protocol is designed to work properly with that default handling, although not with fastest processing.

But: Tx_TFO shall investigate at *Tc*, how far the transmission of the current TFO Message is proceeded and shall **"Modify on the Fly"** this last TFO_Message before *Tc* into the first one after *Tc*, see Figure 10.

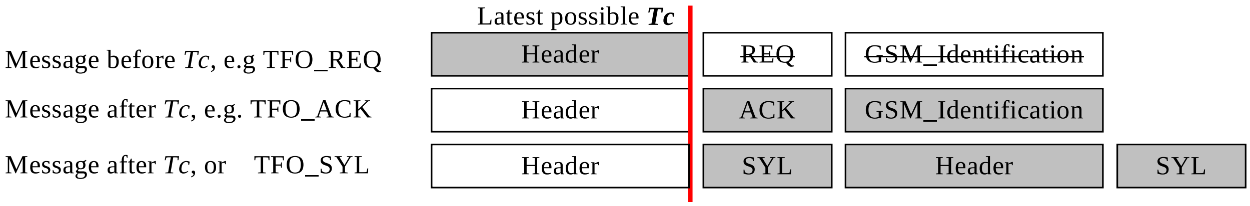

Figure 10: Modification on the Fly within the Header Transmission, examples

8.4 Rx_TFO Process

The Rx_TFO Process receives TFO Messages and TFO Frames from the A-Interface and synchronises to them, i.e. checks correct sync and contents. It bypasses all PCM samples and TFO Frames directly to Tx_TRAU for further processing. The Rx_TFO Process further extracts all the control bits and TFO Messages and sends corresponding Rx_TFO Messages to the TFO_Protocol Process.

8.4.1 Search for and Monitoring of TFO Synchronization

The monitoring of TFO Frame or TFO Message synchronisation shall be a continuous process. Typically, TFO Messages and TFO Frames follow each other with a well defined phase relation. Insertion of T_Bits or octet slips may, however, disturb that regular phase relation every now and then and shall be taken into account. In all error cases, the receiver shall investigate, if sync has been lost due to octet slip, phase adjustment or other events and shall try to resynchronize as fast as possible.

Typically, TFO Frame synchronisation is similar or identical to TRAU Frame synchronisation, see GSM 08.60 and 08.61.

During Tandem Free Operation, however, it is sometimes necessary, to exchange TFO Messages by embedding them into the TFO Frame flow. This is indicated by a control bit (C5). Some of the TFO Frame synchronization bits are then replaced by bits of the TFO Message. TFO Messages follow specific design rules, which can be used to check if synchronisation is still valid.

The first TFO Message or the first TFO Frame (whatever comes first) shall be completely error free to be acceptable by Rx_TFO. After that all "valid" (see subclause 8.4.2) TFO Messages shall be reported to TFO_Protocol with a respective message. If a TFO Message has been received before and synchronisation is not found again for more than 60 ms, i.e. no "present" or "valid" TFO Message can be found during that time, then Rx_TFO shall generate a MSL (Message_Sync_Lost) Message to TFO_Protocol. A "not-valid", but "present" TFO Message shall not be reported, but also no MSL shall be reported, i.e. synchronisation is regarded as not lost, but the TFO Message is ignored.

Similar, the first five "valid" TFO Frames shall be reported to TFO_Protocol with frame number n (n == 1,2,..5). Further valid TFO Frames do not need to be reported.

Similar, if for the first time after the PCM_Idle period, PCM_Non_Idle samples are received, then a PCM_Non_Idle Message shall be sent to TFO_Protocol. Further PCM_Non_Idle samples need not be reported.

If TFO Frame Synchronization is lost, or if too many errors are detected in TFO Frames (no present TFO Frames), then the Rx_TFO shall generate a FSL (Frame_Sync_Lost) Message to TFO_Protocol with frame number n (n == 1,2,3), the number of lost TFO Frames since the last valid TFO Frame. No more than three FSL Messages need to be reported in a series.

NOTE: The MSL and FSL Messages shall not be mixed up with the TFO_SYL Message, by which the distant TFO Partner reports lost synchronisation.

TFO Messages with Extension_Blocks that can not be understood by the receiving TRAU, but which follow the design rules for IS_Extension_Blocks, shall be ignored. This guarantees future expandability.

8.4.2 Errors in TFO Messages and TFO Frames

Some Definitions, which may serve as a guideline:

A TFO Message is called "**error-free**", if no error can be detected within the whole message.

A TFO Message is called "**single-error**", if no more than one bit position differs either in the IS_Header or the IS_Command_Block or the GSM_Ident_Block or the IPE_Mode_Block or the Sync bits and no errors are detectable within the CRC fields or the EX-fields.

A TFO Message may be regarded as "**correctable**", if the phase position is as in preceeding TFO Messages and

- no more than 2 bit positions differ in the IS_Header; and
- no more than 1 error is detected within the IS_Command_Block; and
- no more than 3 errors are detected within the IPE_Mode_Block; and
- no more than 3 errors are detected within the GSM_Ident_Block; and
- no more than 1 error is detected within the Sync-Bit(s); and
- no more than 0 error is detected within the EX-field(s); and
- no more than 0 error is detected within the CRC-fields; and
- the total number of detected errors is not higher than 3.

TFO Message, which are error-free, single-error or correctable are also called "valid" TFO Messages. All other TFO Messages are called "not-valid".

A TFO Message may be regarded as "**present**", if the phase position is as in preceeding TFO Messages and

- no more than 4 bit positions differ in the IS_Header; and
- no more than 2 errors are detected within the IS_Command_Block; and
- no more than 3 errors are detected within the IPE_Mode_Block; and
- no more than 3 errors are detected within the GSM_Ident_Block; and
- no more than 2 errors are detected within the Sync-Bit(s); and
- no more than 1 error is detected within the EX-field(s); and
- no more than 1 error is detected within the CRC-fields; and
- the total number of detected errors is not higher than 5.

Sequences, which differ in more than "present" may not be recognized as TFO Messages at all ("**not-present**").

Note that the insertion of T_Bits may change the phase position of the TFO Frames and of bits of an embedded TFO Message. The TFO Message shall in that case be classified after the removal of the T_Bits.

An octet slip may also change the phase position of bits within a regular or embedded TFO Message.

If an error-free or a single-error TFO Message can be found after considering a hypothetical octet slip (±1 sample), then it may be regarded as error-free or single-error and the new phase position shall be regarded as valid, if no valid or present TFO Message can be found at the old phase position.

A **TFO Frame** is called "**error-free**", if no error can be detected within the whole frame.

A TFO Frame is called "**single-error**", if no more than one bit position differs either in the synchronisation bits or the T_Bits and if no other errors can be detected. TFO Frames, which are error-free, or single-error are also called "**valid**" TFO Frames. All other TFO Frames are called "**not-valid**".

A TFO Frame may be regarded as "**present**", if

- no more than 4 bit positions differ in the synchronisation bits
- no more than 2 errors are detected within the T_Bits;
- no more than 1 error is detected within the control bits;
- no more than 1 error is detected within the CRC block; and
- the total number of detected errors is not higher than 5.

Sequences, which differ in more than "present" may not be recognized as TFO Frames at all ("**not-present**").

Note that the insertion or deletion of T_Bits may change the phase position of the TFO Frames .The TFO Frame shall in that case be classified after considering the T_Bits.

An octet slip may also change the phase position of bits within a TFO Frame. Typically a TFO Frame can not be corrected after an octet slip, but the next TFO Frame shall be found again.

The parameters of a valid TFO Frame shall be regarded as "bad", if the BFI flag is set to BFI == "1". Similar definitions hold for other valid TFO Frames, equivalent to Uplink TRAU Frames (see 08.60 and 08.61).

8.5 TFO_Protocol Process

The TFO_Protocol Process is typically invoked whenever a message is received, either from Rx_TRAU, Rx_TFO, Tx_TFO or the local BSS (i.e. the BSC).

Two events are due to modifications of the local MS-BSS configuration,

- a modification of the used speech Codec (New_Local_Codec); and
- a modification of the list of the alternative speech Codecs (New_Local_Codec_List).

The New_Local_Codec is extracted from the uplink TRAU Frames and reported by Rx_TRAU.

The New_Local_Codec_List is reported by the BSS in a manufacturer dependent way.

It may happen during an established TFO connection that the used Codec is identified as not optimal. Then the distant partner (e.g. a GCME) may inform the TRAU by a TFO_REQ_P Message that another Codec would be preferred.

The TRAU has to inform the local BSS about the preferred Codec, but continues with TFO until an optional In_Call_Modification is performed by the local BSS.

8.5.1 Messages from Rx_TRAU or local BSS

Rx == New_Speech_Call (Local_Used_Codec);	Rx_TRAU is activated by BTS.
Rx == New_Local_Codec (New_Local_Used_Codec);	In Call Modification to other Codec Type.
Rx == Data_Call;	In Call Modification to Data_Call.
Rx == Local_Codec_List;	Manufacturer dependent, optional, from BSS.
Rx == TRAU_Idle;	Manufacturer dependent, either from BTS or BSS.

8.5.2 Messages to Tx_TRAU

Tx_TRAU := Accept_TFO;	if TFO Frames are correctly received, they shall be used.
Tx_TRAU := Ignore_TFO;	TFO Frames, even if received correctly, shall be ignored.

8.5.3 Optional Messages to the local BSS

BSS := TFO (Distant_Used_Codec, Distant_Codec_List, Distant_Preferred_Codec, …).

8.5.4 Messages to and from Tx_TFO

The symbol () indicates that these Messages contain parameters, see clause 6.

Tx := TFO_REQ ();	main TFO_REQ Message.
Tx := TFO_ACK ();	main TFO_ACK Message, response only to TFO_REQ.
Tx := TFO_REQ_L ();	used in Mismatch, Operation and Periodic_Retry to inform about alternative Codecs.
Tx := TFO_ACK_L ();	response only to TFO_REQ_L.
(Tx := TFO_REQ_P ());	undefined for TRAU, defined only for GCME.
Tx := TFO_TRANS ();	command IPEs to go transparent.
Tx := TFO_NORMAL;	reset IPEs into their normal operation.
Tx := TFO_FILL;	mainly to pre-synchronise IPEs.
Tx := TFO_DUP;	"I receive TFO Frames in Establishment".
Tx := TFO_SYL;	"I lost TFO Frame synchronisation".
Tx := Begin_TFO;	Insert TFO Frames from now on.
Tx := Discontinue_TFO;	Discontinue inserting TFO Frames.
Clear Tx_Queue;	Clears all remaining commands from Tx_Queue.
Rx == Runout;	Reports that the continuous stream of outgoing TFO Messages may be interrupted.

8.5.5 Messages from Rx_TFO

The symbol () indicates that these Messages contain parameters, see clause 6.

Rx == TFO_REQ ().

Rx == TFO_ACK ().

Rx == TFO_REQ_L ().

Rx == TFO_ACK_L ().

Rx == TFO_REQ_P () ;requests an other, preferred Codec, plus Codec_List.

Rx == TFO_TRANS () ;may serve as alternative TFO_ACK in some cases!.

Rx == TFO_NORMAL.

Rx == TFO_FILL.

Rx == TFO_DUP.

Rx == TFO_SYL.

Rx == TFO_Frame () ;TFO_Frame (Distant_Used_Codec; Number_of_Received_Frames).

Rx == Frame_Sync_Lost () ;Frame_Sync_Lost (Number_of_Lost_Frames).

Rx == Mess_Sync_Lost ;Message_Sync_Lost.

Rx == PCM_Non_Idle ;at the beginning of a period with several samples/frame different from PCM_Idle.

The message "TFO_Frame ()" needs to be sent only at the first five occurrences, either after a not valid TFO Frame, or if the Distant_Used_Codec changed.

The message "Frame_Sync_Lost ()" needs to be sent only at the first five occurrences of errors in TFO Frames or loss of synchronisation, after a correctly received TFO Frame.

The message "Mess_Sync_Lost" is sent, when after a valid TFO Message no following TFO Message is found.

9 State Machine of the TFO_Protocol Process

The TFO_Protocol Process can be described by a State Machine, consisting of 15 States: five main States with several sub-States; exception handling needs further States, see figure 11:

Initialisation (• Not_Active, • Wakeup).

Establishment (• First_Try, • Continuous_Retry, • Periodic_Retry, • Monitor, • Mismatch).

Contact (• Contact).

Konnect (• Konnect).

Operation (• Operation).

Local Handover (• Fast_Try, • Fast_Contact).

Distant Handover (• Sync_Lost, • Re_Konnect).

Misbehavour (• Failure).

It is assumed that Events (Conditions checking, Actions and Transition to an other State) are handled almost instantaneous and in any case significantly shorter than the time required to complete the transmission of any one TFO Message or TFO Frame.

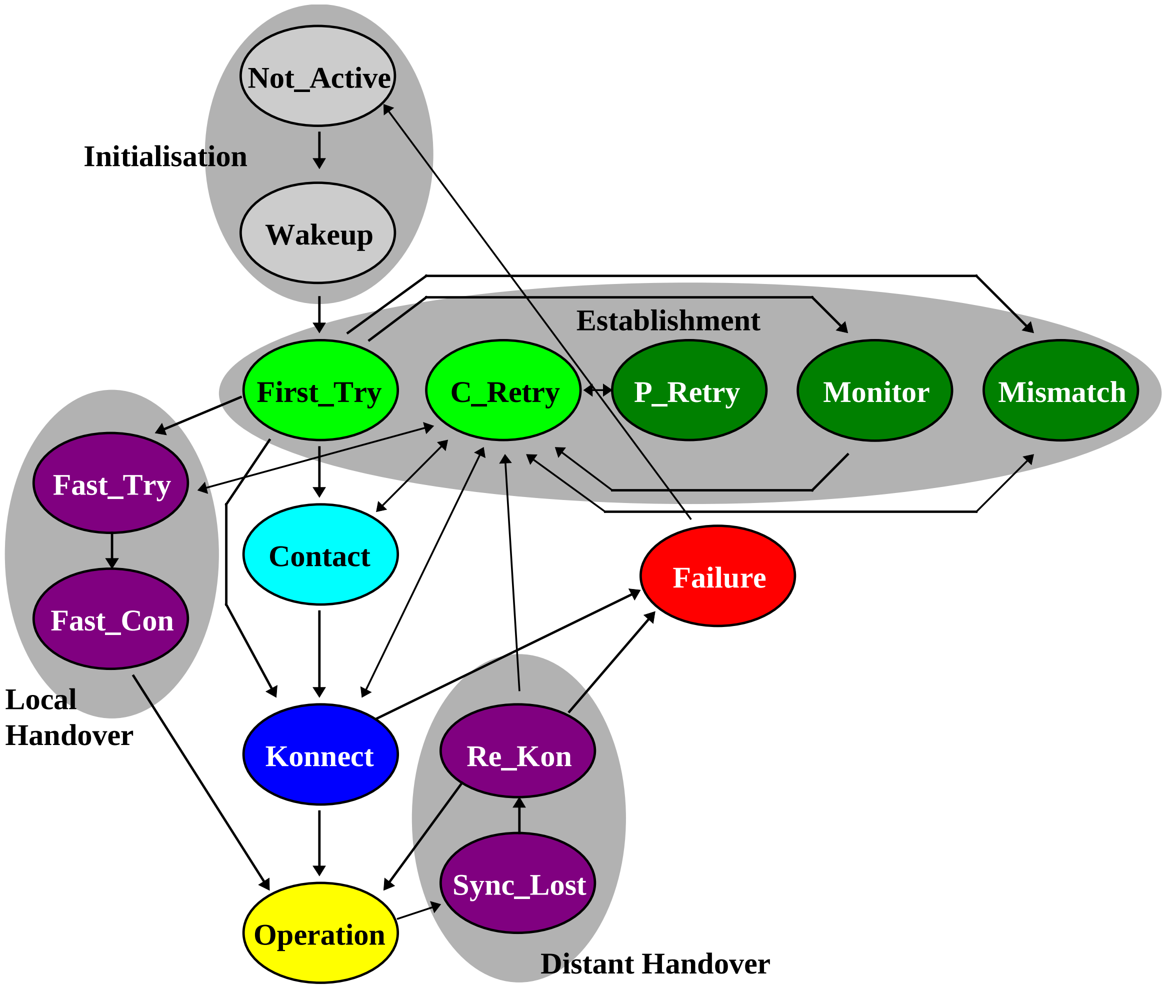

Figure 11: TFO_Protocol State Machine with most important transitions

9.1 Initialisation

9.1.1 Not_Active State

The TRAU in Not_Active receives and sends the PCM_Idle patterns from and onto the A interface. Similarly, it receives and sends Abis_Idle patterns from and onto the Abis/Ater interface. This is not described further.

The TRAU may also be in Data mode, which is also not described further, but is handled here as "Not_Active".

If PCM_Non_Idle patterns are received prior to TRAU Speech Frames, then these PCM_Non_Idle patterns shall be ignored - even if they contain possibly TFO Messages.

9.1.2 Wakeup State

The Wakeup State is entered, when the TRAU is activated by receiving uplink TRAU Speech Frames on the Abis/Ater interface. The TRAU then sends corresponding decoded PCM samples onto to A interface.

If TRAU Speech Frames are received, then the decoded PCM samples are sent to the A interface. Still the TRAU receives PCM_Idle patterns from the A interface. This Wakeup State may last for some while, until the normal (tandem) call connection is established and PCM_Non_Idle samples are received.

The transition to Establishment is performed, if both, TRAU Speech Frames and PCM_Non_Idle patterns, are received. This is the point in time where the time out for TFO Messages starts, i.e. a maximum number of TFO_REQ Messages shall be sent after that.

9.2 Establishment

The Establishment summits several slightly different situations:

First_Try	when the TRAU just has started; it sends TFO_REQ Messages continuously;
Continuous_Retry	when Contact to a TFO Partner has existed, but was interrupted recently;
Periodic_Retry	when Contact to a TFO Partner had existed, but was interrupted some time ago;
Monitor	when no TFO partner could be found, but the TRAU continues to monitor the A Interface;
Mismatch	when a TFO partner with a different Codec has been identified.

Loopback is a specific situation, when the call is still not through connected and the TRAU receives the own sent signals. No specific State is allocated to describe this situation. Instead, loopback is handled in First_Try and Continuous_Retry.

Common to all these situations is that the TRAU does not know, if there is a distant TFO partner and/or if the links are digitally transparent. Typically, TFO_REQ Messages are sent and expected.

Due to handover cases it might, however, happen that a TRAU is initialised into an existing connection and therefore the other TFO Partner may be in any State and all other TFO Messages may be received, too.

Especially important is, when TFO Frames are received, since then it can be assumed that an existing TFO Connection was handed over to a newly initialised TRAU and the TFO should be continued - if possible uninterrupted - as soon as possible. The TRAU may see from the TFO Frames the Distant_Used_Codec of a GSM Partner and that the receiving path is digitally transparent, but it can not assume that the path to the other TRAU is also (already) transparent. TFO_Protocol enters the exceptional State: Fast_Try, sending a specific, short TFO_DUP Message to test the other direction.

9.2.1 First_Try State

The TRAU sends and receives PCM samples on and from the A interface. Regular TFO_REQ Messages are sent onto the A interface continuously for a certain maximum time. After that, if no TFO Partner answers before, Tx_TFO reports a Runout of TFO Messages, and TFO_Protocol enters automatically into the Monitor State.

If TFO_REQ Messages are received with the same, own Signature, then a circuit loop back is assumed, i.e. the call is still not through-connected. The TRAU selects a new Signature and continues sending TFO_REQ Messages, until a different Signature is received. Since loop back delays may be substantial in some cases, the TRAU has to remember and compare also the previously selected own Signature. Care has to be taken that the Signature selection contains a true random element to avoid that two different TRAUs select by coincidence identical signatures again and again.

9.2.2 Continuous_Retry State

TFO Contact had existed, either by TFO Messages or by TFO Frames, but was interrupted and sync was lost. The TRAU sends a maximum number of regular TFO_REQ Messages continuously, to test, if TFO could be re-established. If Tx_TFO reports a Runout of TFO Messages, then the TFO_Protocol enters the Periodic_Retry State.

9.2.3 Periodic_Retry State

Entered from Continuous_Retry, TFO_Protocol tests from time to time by a single TFO_REQ_L, if TFO could be re-established. As soon as a TFO Message is received, TFO_Protocol leaves this State.

NOTE: Since no contiguous transmission of TFO Messages is ongoing, possible IPEs may be unsynchronized.

9.2.4 Monitor State

The TRAU monitors the A interface for TFO Messages or TFO Frames, but it does not send TFO Messages or TFO Frames. As soon as a TFO Message from a distant partner (a TRAU or a GCME) has been received, the TRAU knows that a TFO Partner exists and it knows that the transmission path from the partner is digitally transparent.

The TRAU may already now see, whether TFO is possible, but it must ensure that all IPEs are synchronised. It therefore transits into the Continuous_Retry State. In case of Codec Mismatch, it terminates the TFO Protocol by sending TFO_REQ_L back, informs its local BSS and transits into Mismatch.

NOTE: Since no contiguous transmission of TFO Messages is ongoing, possible IPEs may be unsynchronized.

9.2.5 Mismatch State

From an previous contact it is obvious, that a distant TFO Partner exists, but the Codecs do not match.

The TRAU waits without sending TFO Messages or TFO Frames, if either the distant TFO Partner changes to the Local_Used_Codec, or the local BSS solves the Codec mismatch situation by an intra cell handover to the Distant_Used_Codec.

NOTE: Since no contiguous transmission of TFO Messages is ongoing, possible IPEs may be unsynchronized.

9.3 Contact State

There is a distant TFO Partner, which has sent TFO_REQ. The Codecs do match. The link from the distant partner is transparent. Now TFO_ACK need to be sent to check the transparency of the link to the distant partner.

As soon as a TFO_ACK or TFO_TRANS from a distant partner has been received, the TRAU knows that the links in both directions are digitally transparent. The TRAU sends TFO_TRANS to bypass the IPEs and starts sending TFO Frames. It transits into Konnect State.

9.4 Konnect State

The TRAU sends TFO Frames and possibly embedded TFO Messages as long as it receives correct TFO Messages.

The first received TFO Frame causes the transition into the final Operation State.

If no TFO Frames are received within a certain period, the TRAU transits to the Failure State.

9.5 Operation State

In this State - the main State of TFO_Protocol - the TRAU sends and receives TFO Frames, thus the TFO Connection is fully operating. TFO Messages may occur embedded into TFO Frames.

9.6 Local Handover

9.6.1 Fast_Try State

When the TRAU in First_Try receives suddenly TFO Frames and the Codecs do match, then there is a high probability that a local handover has initialised the TRAU into an existing TFO connection and a fast TFO establishment is likely. The TFO_Protocol has still to check, whether the link to the distant TFO Partner is (already) transparent. This is done by the specific TFO_DUP Message.

Since the handover must have been a local handover, i.e. close to the (new) TRAU, it can be assumed that the possibly existing IPEs are still in transparent mode and TFO Messages therefore pass through directly.

9.6.2 Fast_Contact State

This State is entered from First_Try via Fast_Try, if TFO Frames and then TFO_SYL Messages are received. The TRAU continues to send TFO_DUP Messages, until TFO Frames are received again. Then it immediately starts to send TFO Frames, with a TFO_TRANS embedded into the first TFO Frames. The TRAU transits directly to Operation State.

9.7 Distant Handover, TFO Interruption

9.7.1 Sync_Lost State

If the TRAU was in Operation State and suddenly the TFO Frame synchronisation is lost, then the TRAU enters the Sync_Lost State for a short while, before it transits to Continuous_Retry.

If synchronisation was lost due to a distant handover, then a fast TFO establishment might be possible and the TRAU enters Operation State soon again. In Sync_Lost it expects TFO_DUP Message as confirmation of the distant handover. Then it transits to Re_Konnect.

9.7.2 Re_Konnect State

This State is entered from Operation via Sync_Lost, if TFO_DUP Messages are received. The TRAU starts immediately to send TFO Frames again, with a TFO_TRANS embedded into the first TFO Frames. The TRAU transits back to Operation State, as soon as TFO Frames are received, again.

9.8 Failure State

This State is entered when the distant partner shows an incorrect behaviour. The TRAU then sends pure PCM samples onto the A interface and waits for the failure to disappear. It does not send TFO Frames or TFO Messages.

10 Detailed Description of TFO_Protocol

The TFO_Protocol Process is always in one well defined State. An Event triggers Actions and a Transition into another State. The TFO Protocol is described in a table-wise manner, with a syntax as defined in Table 8.

Table 8: Definition of the Syntax for the State Machine Description

Event: Number:	<Received Message> <running number>	…	<Other Event> <running number>
Condition: &	[<Condition>] [<Condition>]		[<Condition>] [<Condition>]
Comment: **State:**	[<Comment>]		[<Comment>]
<Actual State>:	<Action Name>;[<Action Name>;] <Next State>; [<Comment>]		<Action Name>;[<Action name>;] <Next State>; [<Comment>]
…			
<Actual State>:	<Action Name>;[<Action Name>;] <Next State>; [<Comment>]		<Action Name>;[<Action Name>;] <Next State>; [<Comment>]

Several tables, table 11 to table 18, are necessary to describe the whole State Machine.

The Actions are described in Table 10, with a syntax as defined in table 9.

Table 9: Definition of Syntax for Action Table

Name	Action List	Comment
<Action Name>	<Action >;[<Action >;]	<Comment>
…		
<Action Name>	<Action >;[<Action >;]	<Comment>

Tx := TFO_REQ means, that TFO_Protocol places a command into Tx_Queue. Tx_TFO handles the details autonomously and generates a TFO_REQ Message for transmission over the A interface, when it comes to that command.

Tx := 31*TFO_REQ means: put 31 TFO_REQ commands into Tx_Queue. Not necessarily all will really trigger TFO_REQ Messages. In most cases Tx_Queue will be cleared before. Similar definitions hold for the other messages.

The Tx_Queue is a first_in_first_out command queue. It is filled by TFO_Protocol and read by Tx_TFO.

Clear Tx_Queue, means that all remaining commands are deleted from the Tx_Queue in that very moment (time *Tc*).

Note that due to the duration time to transmit a TFO_Message completely, the TFO_Protocol Process is often already within an other State while still TFO Messages commanded in earlier States are within the Tx_Queue or under transmission.

BSS := TFO () means that a message is sent to the local BSS; similar

Tx_TRAU := … means a message to Tx_TRAU.

An Event **TFO_REQ** means that a TFO_REQ Message was correctly received on the A interface. The Rx_TFO Process has sent a message to TFO_Protocol, containing the new values for the respective variables. TFO_Protocol updates its variables with the new values. Similar definitions hold for the other messages.

One Timer **T := <Time_out>** is necessary to describe time out situations. The notation **T := DIS** means that the Timer is disabled. Positive values are decremented in an hidden background process in steps of 20 ms. When T gets to the value "0", then the TFO_Protocol process is invoked.

Local_Used_Codec (short form: **Luc)** means the type of speech Codec used in the local TRAU and BSS (e.g. FR, EFR, HR).

New Local_Used_Codec (Nluc) refers to the new codec received in "In_Call_Modifications".

Distant_Used_Codec (Duc) means the type of speech Codec used by the distant partner, as reported in TFO_REQ… or TFO_ACK… (e.g. FR, EFR, HR).

Distant_Preferred_Codec (DPC) means the type of speech Codec that the distant partner would prefer, as reported in TFO_REQ_P (e.g. FR, EFR, HR).

Local_Codec_List (LCL) means the list of all Codecs that could alternatively be used, i.e. which are supported by both the local MS and the local BSS. It always contains at least the Local_Used_Codec.

It is reported in TFO_REQ_L, TFO_ACK_L or TFO_REQ_P.

Distant_Codec_List (DCL) means the list of all Codecs that could alternatively be used, i.e. which are supported by the distant MS and the distant BSS. It always contains at least the Distant_Used_Codec.

All these variables are initialized to **UNKNOWN**, which means that the contents of the variables are not defined.

Local_Signature (Lsig) means the 8-bit random number in TFO_REQ, which identifies the local TFO_REQ Messages. It is also used in TFO_REQ_L.

Distant_Signature (Dsig) means the 8-bit random number as received in TFO_REQ, TFO_REQ_L and TFO_REQ_P,

in TFO_ACK and TFO_ACK_L

If received in TFO_REQ, TFO_REQ_L and TFO_REQ_P, then it should be different to the Local_Signature, otherwise loop back must be assumed (exceptions exist).

If received in TFO_ACK or TFO_ACK_L, then it should be identical to the Local_Signature, otherwise the TFO_ACK is not a response to an own TFO_REQ respectively TFO_REQ_L, but maybe was created during an handover situation.

Local Channel Type (LCh) and **distant Channel Type (DCh)** refer to the 8 or 16 kBit/s transparent channel used by the local Tx_TFO respectively received by the distant TFO_TRANS.

Error protection and error handling: It is assumed that the defined error protection is strong enough for the error rates encountered on typical A interface links. The few occuring errors are in practically all cases detected and possibly even corrected by Rx_TFO, before reported to TFO_Protocol. Therefore TFO_Protocol can rely on the correctness of the received Events. The protocol is, however, "selfhealing" and will handle the unlikely erroroneous reported Events, too.

The Event "**PCM_Non_Idle**" is given if in State Wakeup, if more than one PCM samples are received that are different to PCM_Idle.

Fast Handover handling: The defined protocol assumes that the new TRAU, to which the handover is performed, is already in State Wakeup before the A-Interface is switched to that TRAU. Only then the TFO Frames can be received by that TRAU and fast handover handling is possible.

Timing: If two Events occur by coincidence at the same time, then they shall be processed in the order given by the tables 10 to 17 (left to right) . TFO Messages arrive always some time before the embedding TFO Frame and shall be handled therefore first.

Runout is the Event, when the last TFO Message has been taken from the Tx_Queue and the last 10 bits are going to be sent by Tx_TFO to the A interface. So there is still some time for TFO_protocol to react and place a further TFO Message into Tx_Queue, which then shall be transmitted without gap to the messages before.

Table 10: Defined Actions

Name	Actions	Comments
C	Clear Tx_Queue; T := DIS;	Initialise Tx_Queue and disable the timer
T1	T := 1s;	Set Timeout ot 1 second
T2	T := 2s;	Set Timeout to 2 seconds
T5	T := 5s;	Set Timeout ot 5 seconds
NoAc	.	No Action required
S	Lsig := New_Random_Number;	Generate new Signature and set Old_Sig to unknown;
	Old_Sig := UNKNOWN;	if no Loopback is assumed.
SO	Old_Sig := Lsig;	Remember old Signature and generate a new Signature,
	Lsig := New_Random_Number;	if Loopback is assumed.
U	Old_Sig := UNKNOWN;	Reset Old_Sig before leaving FIT or COR
F	Tx := 3*TFO_FILL;	"Hello IPEs! Please synchronise!"
T	Tx := TFO_TRANS ();	"Hello IPEs! Please open a transparent channel!"
N	Tx := TFO_NORMAL;	"Hello IPEs! Please return to normal operation!"
REQ	Tx := 35*TFO_REQ;	“Hello Partner? Can You do TFO with me?”
ACK	Tx := 7*TFO_ACK;	“Yes, I can do TFO with You!”
SYL1	Tx := TFO_SYL;	"Hello Partner! I lost one or more TFO_Frames!"
SYL	Tx := 4*TFO_SYL;	"Hello Partner! Serious interruption of TFO_Frames!"
DUP	Tx := 5*TFO_DUP;	Handover? “Hey, I see Your TFO Frames, Fine!”
L1	Tx := TFO_REQ_L;	"Here is my Codec_List! Can you hear me?"
L	Tx := 6*TFO_REQ_L;	"Here is my Codec_List, please acknowledge!"
LA	Tx := TFO_ACK_L;	"Yes, I received Your Codec_List! Here is mine!"
BT	Tx := Begin_TFO;	Begin Transmission of TFO Frames
DT	Tx := Discontinue_TFO;	Discontinue Transmission of TFO Frames
IT	Tx_TRAU := Ignore_TFO;	Tx_TRAU works as conventional downlink TRAU
AT	Tx_TRAU := Accept_TFO;	Tx_TRAU bypasses TFO_Frames
B	BSS := TFO ();	"Hello BSS! Some news from the TFO_Scene!"

Table 11: Call Setup and Loopback Handling

Event:	**New_Speech_Call**	**PCM_Non_Idle**	**TFO_REQ**	**TFO_REQ**
	.	.	.	.
Number:	24	29	0	0a
Condition:	.	.	Duc==Luc	Duc==Luc
&	.	.	Dsig==Lsig	Dsig==Old_Sig
Comment:	activate TRAU	A-Int. gets active	Loopback (LB)	Loopback (LB)
.	from BTS, e.g. by	occurs only at	or distant handover	or distant hand
State:	2 TRAU Frames	beginning	(HO)? wrong Sig	over (HO)?
NAC:	C;S;IT;	----------	----------	----------
Not_Active	WAK;	----------	----------	----------
.	**typ. 1rst Event**	.	.	.
WAK:	----------	C;F;REQ;	----------	----------
Wakeup	----------	FIT;	----------	----------
.	.	**typ. 2nd Event**	.	.
FIT:	----------	----------	C;SO;REQ;	NoAc;
First_Try	----------	----------	FIT;	FIT;
.	.	.	**LB!**	**Ignore LB**
COR:	----------	----------	C;SO;REQ;	NoAc;
Continuous	----------	----------	COR;	COR;
Retry	.	.	LB!?	**Ignore LB**
PER:	----------	----------	C;F;S;ACK;	----------
Periodic	----------	----------	CON;	----------
Retry	.	.	Dist. HO!	.
MON:	----------	----------	C;F;S;REQ;	----------
Monitor	----------	----------	FIT;	----------
.	.	.	Dist. HO!	.
MIS:	----------	----------	C;F;S;ACK;	----------
Mismatch	----------	----------	CON;	----------
.	.	.	Dist. HO!	.
CON:	----------	----------	C;SO;REQ;	----------
Contact	----------	----------	COR;	----------
.	.	.	save way	.
FAT:	----------	----------	C;SO;REQ;	----------
Fast	----------	----------	COR;	----------
Try	.	.	save way	.
FAC:	----------	----------	C;SO;REQ;	----------
Fast	----------	----------	COR;	----------
Contact	.	.	save way	.
KON:	----------	----------	C;DT;SO;REQ;T1;	----------
Konnect	----------	----------	COR;	----------
.	.	.	IPEs transparent!	.
REK:	----------	----------	C;DT;SO;REQ;IT;B;T1;	----------
Re_Konnect	----------	----------	COR;	----------
.	.	.	IPEs transparent!	.
SOS:	----------	----------	C;IT;S;REQ;B;T1;	----------
Sync_Lost	----------	----------	COR;	----------
.	.	.	Contact is back	.
OPE:	----------	----------	----------	----------
Operation	----------	----------	----------	----------
.	.	.	.	.
FAI:	----------	----------	NoAc;	----------
Failure	----------	----------	FAI;	----------

Table 12: Most Important Cases, Especially at Call Setup

Event:	**TFO_REQ**	**TFO_ACK**	**TFO_ACK**	**TFO_TRANS**	**TFO_FRAME**
	.	.	.	.	.
Number:	1	2	3	4	5
Condition:	Duc==Luc	Duc==Luc	Duc==Luc	DCh==LCh	Duc==Luc
&	Dsig!=Lsig	Dsig==Lsig	Dsig!=Lsig	.	n<3
Comment:	Distant REQ	Distant ACK	Wrong Response	similar to ACK	one or two
.	Good Signature	Good Signature	Handover?	As response	TFO Frames
State:	.	.	.	to loc ACK_?	.
NAC:	----------	----------	----------	----------	----------
Not_Active	----------	----------	----------	----------	----------
.	.	.	.	.	.
WAK:	----------	----------	----------	----------	----------
Wakeup	----------	----------	----------	----------	----------
.	.	.	.	.	.
FIT:	C;U;ACK;	C;U;T;BT;T;T1;	C;REQ;	NoAc;	C;U;DUP;
First_Try	CON;	KON;	FIT;	FIT;	FAT;
.	**typical**	**typical; IPEs!**	.	wait for Framee	**1: HO**
COR:	C;U;ACK;	C;U;T;BT;T;T1;	C;REQ;	NoAc;	C;U;DUP;
Continuous	CON;	KON;	COR;	COR;	FAT;
Retry	typical	**typical; IPEs!**	.	wait for Frames	1: Call is back?
PER:	C;F;ACK;	C;F;S;REQ;	C;F;REQ;	NoAc;	C;DUP;
Periodic	CON;	COR;	COR;	PER;	FAT;
Retry	OK, Contact is back	rare case, test	.	wait for Frames	1: Call is back?
MON:	C;F;REQ;	C;F;S;REQ;	C;F;REQ;	NoAc;	C;DUP;
Monitor	FIT;	FIT;	FIT;	MON	FAT;
.	IPEs?	Rare case, test	.	wait for Frames	1: Call is back?
MIS:	C;F;ACK;	C;F;S;REQ;	C;F;REQ;	NoAc;	C;DUP;
Mismatch	CON;	COR;	COR;	MIS;	FAT;
.	Mismatch resolved	rare case, test	.	wait for Frames	1: Call is back?
CON:	C;ACK;	C;T;BT;T;T1;	C;REQ;	C;T;BT;T;T1;	C;T;BT;T;T1;
Contact	CON;	KON;	COR;	KON;	KON;
.	**typical: wait**	**typical: yes!**	.	**yes! Fast way**	missed TRANS?
FAT:	C;REQ;	C;REQ;	C;REQ;	NoAc;	NoAc;
Fast	COR;	COR;	COR;	FAC;	FAT;
Try	save way	save way	save way	wait for Frames	**2: typ. Loc. HO**
FAC:	C;REQ;	C;REQ;	C;REQ;	NoAc;	C;BT;T;L;T2;AT;B;
Fast	COR;	COR;	COR;	FAC;	OPE;
Contact	save way	save way	save way	wait for Frames	**5: typ. Loc. HO**
KON:	C;DT;REQ;T1;	NoAc;	NoAc;	NoAc;	AT;L;T2;B;
Konnect	COR;	KON;	KON;	KON;	OPE;
.	IPEs transparent!	**Typical: wait**	.	**typical: wait**	**typ: call setup**
REK:	C;DT;REQ;IT;B;T1;	C;DT;REQ;IT;B;T1 ;	C;DT;REQ;IT;B;T 1	NoAc;	AT;L;T2;B;
Re_Konnect	COR;	COR;	COR;	REK;	OPE;
.	IPEs transparent!	.	.	wait for Frames	**5: typ. Dis. HO**
SOS:	C;IT;REQ;B;T1;	C;IT;REQ;B;T1;	C;IT;REQ;B;T1;	NoAc;	C;BT;T;L;T2;B;
Sync_Lost	COR;	COR;	COR;	SOS;	OPE;
.	Contact is back	Contact is back	Contact is back	wait for Frames	short Interrupt?
OPE:	----------	----------	----------	NoAc;	NoAc;
Operation	----------	----------	----------	OPE;	OPE;
.	.	.	.	typ in HO	**Main! TFO!**
FAI:	NoAc;	NoAc;	NoAc;	NoAc;	NoAc;
Failure	FAI;	FAI;	FAI;	FAI;	FAI;

Table 13: In Call Modification and Handover

Event:	**New_Local_Code c**	**New_Local_Code c**	**TFO_FRAME**	**TFO_SYL**	**TFO_DUP**
	.	.	.	.	.
Number:	25	26	6	7	8
Condition:	Duc==Nluc	Duc!=Nluc	Duc==Luc	.	.
&	.	.	n>2	.	.
Comment:	in Call Modif.	In Call Modif.	Three or more	the dist. TRAU	the dist. TRAU
.	Mismatch resolv.	Mismatch occurs!	TFO Frames	lost sync	recognised HO
State:	(Luc!=Nluc)	(Luc!=Nluc)	.	in OPE	
NAC:	----------	----------	----------	----------	----------
Not_Active	----------	----------	----------	----------	----------
.	.	.	.	.	.
WAK:	NoAc;	NoAc;	----------	----------	----------
Wakeup	WAK;	WAK;	----------	----------	----------
.	.	.	.	.	.
FIT:	C;REQ;	C;REQ;	----------	NoAc;	NoAc;
First_Try	FIT;	FIT;	----------	FIT;	FIT;
.	restart	restart	.	HO? Ignore	HO? ignore
COR:	C;REQ;	C;REQ;	----------	NoAc;	NoAc;
Continuous	COR;	COR;	----------	COR;	COR;
Retry	.	.	.	ignore	ignore
PER:	L1;T5;	L1;T5;	----------	C;F;REQ;	C;F;REQ;
Periodic	PER;	PER;	----------	COR;	COR;
Retry	.	.	.	rare case, test	rare case, test
MON:	NoAc;	NoAc;	----------	C;F;REQ;	C;F;REQ;
Monitor	MON	MON	----------	FIT;	FIT;
.	.	.	.	rare case, test	rare case, test
MIS:	C;F;REQ;	L;T2;B;	----------	C;F;REQ;	C;F;REQ;
Mismatch	COR;	MIS;	----------	COR;	COR;
.	**Mismatch res.**	**Direct info.**	.	rare case, test	rare case, test
CON:	----------	C;L;T2;B;	----------	C;F;REQ;	C;F;REQ;
Contact	----------	MIS;	----------	COR;	COR;
.	.	.	.	rare case, test	rare case, test
FAT:	----------	C;L;T2;B;	NoAc;	NoAc;	C;F;REQ;
Fast	----------	MIS;	FAC;	FAC;	COR;
Try	.	.	.	**3: typ. Loc. HO**	rare case, test
FAC:	----------	C;L;T2;B;	C;BT;T;L;T2;AT;B;	NoAc;	C;F;REQ;
Fast	----------	MIS;	OPE;	FAC;	COR;
Contact	.	.	.	**4: typ. Loc. HO**	rare case, test
KON:	----------	C;DT;L;T2;B;	----------	NoAc;	NoAc;
Konnect	----------	MIS;	----------	KON;	KON;
.	.	.	.	wait, short int?	other TRAU?
REK:	----------	C;DT;IT;L;T2;B;	----------	C;DT;SYL;	NoAc;
Re_Konnect	----------	MIS;	----------	SOS;	REK;
.	.	.	.	IPEs not transp?	**4: typ. Dist. HO**
SOS:	----------	C;IT;L;T2;B;	----------	NoAc;	C;BT;T;T1;
Sync_Lost	----------	MIS;	----------	SOS;	REK;
.	.	.	.	short Inter?	**3: typ. Dis. HO**
OPE:	----------	C;DT;IT;L;T2;B;	NoAc;	NoAc;	NoAc;
Operation	----------	MIS;	OPE;	OPE;	OPE;
.	.	.	Main! TFO!	Short interrupt?	Typical
FAI:	----------	NoAc;	NoAc;	NoAc;	NoAc;
Failure	----------	FAI;	FAI;	FAI;	FAI;

Table 14: Special Matching TFO Messages

Event:	**TFO_REQ_L**	**TFO_REQ_L**	**TFO_ACK_L**	**TFO_ACK_L**	**TFO_REQ_P**	**TFO_REQ_P**
	.	.	.	.	.	.
Number:	9	10	11	12	13	14
Condition:	Duc==Luc	Duc==Luc	Duc==Luc	Duc==Luc	.	.
&	Dsig==Lsig	Dsig!=Lsig	Dsig==Lsig	Dsig!=Lsig	Dsig==Lsig	Dsig!=Lsig
Comment:	Only sent in	Only sent in	Only sent in	.	sent by GCME	sent by GCME
.	MIS/OPE/PER	MIS; / OPE / PER	MIS;	.	only	only
State:	HO? Loop?	Codec_List	HO?	HO?	embedded	embedded
NAC:	----------	----------	----------	----------	----------	----------
Not_Active	----------	----------	----------	----------	----------	----------
.	.	.	.	.	.	.
WAK:	----------	----------	----------	----------	----------	----------
Wakeup	----------	----------	----------	----------	----------	----------
.	.	.	.	.	.	.
FIT:	NoAc;	NoAc;	NoAc;	NoAc;	----------	----------
First_Try	FIT;	FIT;	FIT;	FIT;	----------	----------
.	ignore	ignore	ignore	ignore	.	.
COR:	NoAc;	NoAc;	NoAc;	NoAc;	----------	----------
Continuous	COR;	COR;	COR;	COR;	----------	----------
Retry	ignore	ignore	ignore	ignore	.	.
PER:	C;F;S;REQ;	C;F;REQ;	C;F;S;REQ;	C;F;REQ;	----------	----------
Periodic	COR;	COR;	COR;	COR;	----------	----------
Retry	start again	start again	test	test	.	.
MON:	C;F;S;REQ;	C;F;REQ;	C;F;S;REQ;	C;F;REQ;	----------	----------
Monitor	FIT;	FIT;	FIT;	FIT;	----------	----------
.	test	test	test	test	.	.
MIS:	C;F;S;REQ;	C;F;REQ;	C;F;S;REQ;	C;F;REQ;	S;LA;B;	LA;B;
Mismatch	COR;	COR;	COR;	COR;	MIS;	MIS;
.	test	test	test	test	acknowledge	acknowledge
CON:	C;S;REQ;	C;REQ;	C;S;REQ;	C;REQ;	----------	----------
Contact	COR;	COR;	COR;	COR;	----------	----------
.	save way!	Save way!	Save way!	Save way!	.	.
FAT:	C;S;REQ;	C;REQ;	C;S;REQ;	C;REQ;	S;LA;B;	LA;B;
Fast	COR;	COR;	COR;	COR;	FAT;	FAT;
Try	save way!	Save way!	Save way!	Save way!	Acknowledge	acknowledge
FAC:	C;S;REQ;	C;REQ;	C;S;REQ;	C;REQ;	S;LA;B;	LA;B;
Fast	COR;	COR;	COR;	COR;	FAC;	FAC;
Contact	save way!	Save way!	Save way!	Save way!	Acknowledge	acknowledge
KON:	C;DT;S;REQ;T1;	C;DT;REQ;T1;	C;DT;S;REQ;T1;	C;DT;REQ;T1;	S;LA;B;	LA;B;
Konnect	COR;	COR;	COR;	COR;	KON;	KON;
.	save way!	Save way!	Save way!	Save way!	Acknowledge	acknowledge
REK:	C;DT;S;REQ;T1;	C;DT;REQ;T1;	C;DT;S;REQ;T1;	C;DT;REQ;T1;	----------	----------
Re_Konnect	COR;	COR;	COR;	COR;	----------	----------
.	save way!	Save way!	Save way!	Save way!	.	.
SOS:	C;IT;S;REQ;B;T1;	C;IT;REQ;B;T1;	C;IT;S;REQ;B;T1;	C;IT;REQ;B;T1;	S;LA;B;	LA;B;
Sync_Lost	COR;	COR;	COR;	COR;	SOS;	SOS;
.	save way!	Save way!	Save way!	Save way!	Acknowledge	acknowledge
OPE:	S;L;T2;B;	C;LA;B;	C;B;	S;L;T2;B;	S;LA;B;	LA;B;
Operation	OPE;	OPE;	OPE;	OPE;	OPE;	OPE;
.	tx Codec_List	Ack List, stop	Ack ok, stop	exchange list	acknowledge	acknowledge
FAI:	NoAc;	NoAc;	NoAc;	NoAc;	NoAc;	NoAc;
Failure	FAI;	FAI;	FAI;	FAI;	FAI;	FAI;

Table 15: TFO Messages with mismatching Codec Type

Event:	**TFO_REQ**	**TFO_REQ**	**TFO_ACK**	**TFO_REQ_L**	**TFO_REQ_L**	**TFO_ACK_L**
	.	.	.	.	.	.
Number:	15	16	17	18	19	20
Condition:	Duc!=Luc	Duc!=Luc	Duc!=Luc	Duc!=Luc	Duc!=Luc	Duc!=Luc
&	Dsig==Lsig	Dsig!=Lsig	Dsig==?	Dsig==Lsig	Dsig!=Lsig	Dsig==?
Comment:	Mismatch	Mismatch	Mismatch	Mismatch	Mismatch	Mismatch
.	Wrong Sig, HO?	Good Sig	w/wo HO	Codec_List	Codec_List	Codec_List
State:	.	.	.	Wrong Sig, HO?	.	.
NAC:	----------	----------	----------	----------	----------	----------
Not_Active	----------	----------	----------	----------	----------	----------
.	.	.	.	.	.	.
WAK:	----------	----------	----------	----------	----------	----------
Wakeup	----------	----------	----------	----------	----------	----------
.	.	.	.	.	.	.
FIT:	C;S;L;T2;B;	C;U;L;T2;B;	C;U;L;T2;B;	C;S;LA;B;	C;U;LA;B;	C;U;LA;B;
First_Try	MIS;	MIS;	MIS;	MIS;	MIS;	MIS;
.	rare	**typical: Setup**	HO?	rare	**typical: Setup**	HO?
COR:	C;S;L;T2;B;	C;U;L;T2;B;	C;U;L;T2;B;	C;S;LA;B;	C;U;LA;B;	C;U;LA;B;
Continuous	MIS;	MIS;	MIS;	MIS;	MIS;	MIS;
Retry	.	.	.	.	.	.
PER:	C;F;S;L;T2;B;	C;F;L;T2;B;	C;F;L;T2;B;	C;F;S;LA;B;	C;F;LA;B;	C;F;LA;B;
Periodic	MIS;	MIS;	MIS;	MIS;	MIS;	MIS;
Retry	.	.	.	.	.	.
MON:	C;F;S;L;T2;B;	C;F;L;T2;B;	C;F;L;T2;B;	C;F;S;LA;B;	C;F;LA;B;	C;F;LA;B;
Monitor	MIS;	MIS;	MIS;	MIS;	MIS;	MIS;
.	.	.	.	.	.	.
MIS:	C;S;L;T2;B;	C;L;T2;B;	C;L;T2;B;	C;S;LA;B;	C;LA;B;	C;LA;B;
Mismatch	MIS;	MIS;	MIS;	MIS;	MIS;	MIS;
.	.	.	.	.	**terminate prot.**	**Terminate prot.**
CON:	C;S;L;T2;B;	C;L;T2;B;	C;L;T2;B;	C;S;LA;B;	C;LA;B;	C;LA;B;
Contact	MIS;	MIS;	MIS;	MIS;	MIS;	MIS;
.	.	.	.	.	.	.
FAT:	C;S;L;T2;B;	C;L;T2;B;	C;L;T2;B;	C;S;LA;B;	C;LA;B;	C;LA;B;
Fast	MIS;	MIS;	MIS;	MIS;	MIS;	MIS;
Try	.	.	.	.	.	.
FAC:	C;S;L;T2;B;	C;L;T2;B;	C;L;T2;B;	C;S;LA;B;	C;LA;B;	C;LA;B;
Fast	MIS;	MIS;	MIS;	MIS;	MIS;	MIS;
Contact	.	.	.	.	.	.
KON:	C;DT;S;L;T2;B;	C;DT;L;T2;B;	C;DT;L;T2;B;	C;DT;S;LA;B;	C;DT;LA;B;	C;DT;LA;B;
Konnect	MIS;	MIS;	MIS;	MIS;	MIS;	MIS;
.	.	.	.	.	.	.
REK:	C;DT;S;L;T2;IT;B;	C;DT;L;T2;IT;B;	C;DT;L;T2;IT;B;	C;DT;S;LA;IT;B;	C;DT;LA;IT;B;	C;DT;LA;IT;B;
Re_Konnect	MIS;	MIS;	MIS;	MIS;	MIS;	MIS;
.	.	.	.	.	.	.
SOS:	C;S;L;T2;IT;B;	C;L;T2;IT;B;	C;L;T2;IT;B;	C;S;LA;IT;B;	C;LA;IT;B;	C;LA;IT;B;
Sync_Lost	MIS;	MIS;	MIS;	MIS;	MIS;	MIS;
.	.	.	.	.	**In_Call_Mod.**	.
OPE:	----------	----------	----------	NoAc;	NoAc;	----------
Operation	----------	----------	----------	OPE;	OPE;	----------
.	.	.	.	trans. Error?	Trans. Error?	.
FAI:	NoAc;	NoAc;	NoAc;	NoAc;	NoAc;	NoAc;
Failure	FAI;	FAI;	FAI;	FAI;	FAI;	FAI;

Table 16: Mismatching TFO_TRANS and TFO Frames

Event:	**TFO_TRANS**	**TFO_FRAME**	**TFO_FRAME**
	.	.	.
Number:	21	22	23
Condition:	DCh!=LCh	Duc!=Luc	Duc!=Luc
&	.	n==1	n>1
Comment:	Mismatch	Mismatch	Mismatch
.	of channel type	for one	for at least
State:	.	TFO Frames	two TFO Frames
NAC:	----------	----------	----------
Not_Active	----------	----------	----------
.	.	.	.
WAK:	----------	----------	----------
Wakeup	----------	----------	----------
.	.	.	.
FIT:	C;U;L;T2;B;	NoAc;	C;U;L;T2;B;
First_Try	MIS;	FIT;	MIS;
.	HO?	HO? be tolerant	**typical in HO**
COR:	C;U;L;T2;B;	NoAc;	C;U;L;T2;B;
Continuous	MIS;	COR;	MIS;
Retry	.	Call Forw.?	.
PER:	C;F;L;T2;B;	NoAc;	C;F;L;T2;B;
Periodic	MIS;	PER;	MIS;
Retry	.	Call Forw.?	.
MON:	C;F;L;T2;B;	NoAc;	C;F;L;T2;B;
Monitor	MIS;	MON	MIS;
.	.	Call Forw.?	.
MIS:	C;L;T2;B;	NoAc;	C;L;T2;B;
Mismatch	MIS;	MIS;	MIS;
.	.	Call Forw.?	.
CON:	C;L;T2;B;	NoAc;	C;L;T2;B;
Contact	MIS;	CON;	MIS;
.	.	.	.
FAT:	C;L;T2;B;	NoAc;	C;L;T2;B;
Fast	MIS;	FAT;	MIS;
Try	.	.	.
FAC:	C;L;T2;B;	NoAc;	C;L;T2;B;
Fast	MIS;	FAC;	MIS;
Contact	.	.	.
KON:	C;DT;L;T2;B;	NoAc;	C;DT;L;T2;B;
Konnect	MIS;	KON;	MIS;
.	.	.	.
REK:	C;DT;L;T2;IT;B;	NoAc;	C;DT;L;T2;IT;B;
Re_Konnect	MIS;	REK;	MIS;
.	.	.	.
SOS:	C;L;T2;IT;B;	NoAc;	C;L;T2;IT;B;
Sync_Lost	MIS;	SOS;	MIS;
.	.	.	.
OPE:	NoAc;	NoAc;	C;DT;L;T2;IT;B;
Operation	OPE;	OPE;	MIS;
.	ignore?	Hard HO?	hard HO into TFO
FAI:	NoAc;	NoAc;	NoAc;
Failure	FAI;	FAI;	FAI;

Table 17: Local Events, Call Termination

Event:	**New_L_Codec_List**	**Data_Call**	**TRAU_Idle**	**TFO_FILL**	**TFO_NORMAL**
	.	.	.	.	.
Number:	30	27	28	37	33
Condition:	.	.	.	.	.
&	.	.	.	.	.
Comment:	from BSS	in Call Modif.	Command from	ignore	ignore
.		.	BTS or BSC	is just	alternative:
State:	.	stop TFO	to Reset TRAU	Filler	Soft Resert
NAC:	NoAc;	NoAc;	NoAc;	----------	----------
Not_Active	NAC;	NAC;	NAC;	----------	----------
.	.	.	.	.	.
WAK:	NoAc;	NoAc;	NoAc;	----------	----------
Wakeup	WAK;	NAC;	NAC;	----------	----------
.	.	.	.	.	.
FIT:	NoAc;	C;N;	C;N;	NoAc;	NoAc;
First_Try	FIT;	NAC;	NAC;	FIT;	FIT;
.	update loc. Par.	.	.	.	.
COR:	NoAc;	C;N;	C;N;	NoAc;	NoAc;
Continuous	COR;	NAC;	NAC;	COR;	COR;
Retry	.	.	.	.	.
PER:	NoAc;	C;N;	C;N;	NoAc;	NoAc;
Periodic	PER;	NAC;	NAC;	PER;	PER;
Retry	.	.	.	.	.
MON:	NoAc;	C;N;	C;N;	NoAc;	NoAc;
Monitor	MON	NAC;	NAC;	MON	MON
.	.	.	.	.	.
MIS:	C;L;T2;	C;N;	C;N;	NoAc;	NoAc;
Mismatch	MIS;	NAC;	NAC;	MIS;	MIS;
.	**direct info**	.	.	.	.
CON:	NoAc;	C;N;	C;N;	NoAc;	NoAc;
Contact	CON;	NAC;	NAC;	CON;	CON;
.	.	.	.	.	.
FAT:	NoAc;	C;N;	C;N;	NoAc;	NoAc;
Fast	FAT;	NAC;	NAC;	FAT;	FAT;
Try	.	.	.	.	.
FAC:	NoAc;	C;N;	C;N;	NoAc;	NoAc;
Fast	FAC;	NAC;	NAC;	FAC;	FAC;
Contact	.	.	.	.	.
KON:	NoAc;	C;DT;N;	C;DT;N;	NoAc;	NoAc;
Konnect	KON;	NAC;	NAC;	KON;	KON;
.	.	.	.	.	.
REK:	NoAc;	C;DT;IT;N;	C;DT;IT;N;	NoAc;	NoAc;
Re_Konnect	REK;	NAC;	NAC;	REK;	REK;
.	.	.	.	.	.
SOS:	NoAc;	C;IT;N;	C;IT;N;	NoAc;	NoAc;
Sync_Lost	SOS;	NAC;	NAC;	SOS;	SOS;
.	.	.	.	.	.
OPE:	L;T2;	C;DT;IT;N;	C;DT;IT;N;	NoAc;	NoAc;
Operation	OPE;	NAC;	NAC;	OPE;	OPE;
.	**direct info**	.	.	.	.
FAI:	NoAc;	C;	C;	NoAc;	NoAc;
Failure	FAI;	NAC;	NAC;	FAI;	FAI;
.	.	**exit from FAI**	**exit from FAI**	.	.

Table 18: Special Events, Timeouts

Event:	**Runout**	**T==0**	**Frame_Sync_Lost**	**Frame_Sync_Lost**	**Mes_Sync_Lost**
	.	.	.	.	.
Number:	31	32	34	35	36
Condition:	.	.	n<3	n>2	.
&	.	.	.	.	.
Comment:	IPEs may become	Time-Out	start to send	Stop TFO Frames	.
.	unsynchronised	.	SYL already	if 3 Frames missing	.
State:	.	.	.	.	.
NAC:	----------	----------	----------	----------	----------
Not_Active	----------	----------	----------	----------	----------
.	.	.	.	.	.
WAK:	----------	----------	----------	----------	----------
Wakeup	----------	----------	----------	----------	----------
.	.	.	.	.	.
FIT:	U;N;	----------	----------	----------	NoAc;
First_Try	MON	----------	----------	----------	FIT;
.	**PSTN Call**	.	.	.	.
COR:	U;L1;T5;	C;N;REQ;	----------	----------	NoAc;
Continuous	PER;	COR;	----------	----------	COR;
Retry	**at end of COR**	Reset IPEs	.	.	.
PER:	NoAc;	L1;T5;	----------	----------	NoAc;
Periodic	PER;	PER;	----------	----------	PER;
Retry	.	**Periodic Test**	.	.	.
MON:	----------	----------	----------	----------	----------
Monitor	----------	----------	----------	----------	----------
.	.	.	.	.	.
MIS:	NoAc;	N;B;	NoAc;	NoAc;	NoAc;
Mismatch	MIS;	MIS;	MIS;	MIS;	MIS;
.	**typ. Final state**	List not Ack_ed!	.	.	.
CON:	REQ;	----------	----------	----------	C;REQ;
Contact	COR;	----------	----------	----------	COR;
.	can this occur?	.	.	.	.
FAT:	REQ;	----------	NoAc;	NoAc;	C;REQ;
Fast	COR;	----------	FAT;	FAT;	COR;
Try	fast HO failed	.	typical in HO	typical in HO	fast HO failed
FAC:	REQ;	----------	NoAc;	NoAc;	C;REQ;
Fast	COR;	----------	FAC;	FAC;	COR;
Contact	fast HO failed	.	typical in HO	typical in HO	fast HO failed
KON:	NoAc;	C;DT;N;	----------	----------	C;DT;REQ;T1;
Konnect	KON;	FAI;	----------	----------	COR;
.	may happen	Misbehaviour!	.	.	after Timeout: N
REK:	NoAc;	C;DT;N;IT;B;	----------	----------	C;DT;REQ;IT;B;T1;
Re_Konnect	REK;	FAI;	----------	----------	COR;
.	may happen	Misbehaviour!	.	.	after Timeout: N
SOS:	REQ;IT;B;T1;	----------	----------	NoAc;	C;REQ;IT;B;T1;
Sync_Lost	COR;	----------	----------	SOS;	COR;
.	after Timeout: N	.	.	wait for Runout	after Timeout: N
OPE:	NoAc;	B;	SYL1;	C;DT;SYL;	NoAc;
Operation	OPE;	OPE;	OPE;	SOS;	OPE;
.	**typ. Final event**	List not Ack_ed!	**1: Alarm, go on**	**2: Alarm, stop!**	**Typ. Final event**
FAI:	NoAc;	----------	----------	----------	NoAc;
Failure	FAI;	----------	----------	----------	FAI;
.	typical	.	.	.	don´t trust!

11 Codec Mismatch Resolution and Codec Optimization

It is not mandatory for a BSS to support the resolution of Codec Mismatch or the Codec Optimization. In that case the Local_Codec_List shall include only the Local_Used_Codec. However, in the optional case, if a BSS sends a Local_Codec_List that includes more than the Local_Used_Codec, then it is mandatory for that BSS to support the resolution of Codec Mismatch or the Codec Optimisation, considering the reported Codec_Types.

The determination of the Local_Codec_List (i.e. the list of all Codecs supported by the local radio leg, consisting of the local MS, the local BSS and the local radio resources) and the communication of the TRAU with the local BSS, is a BSS specific matter and is outside the scope of this specification. However, only Codec_Types that are real alternatives, considering all resources, shall be reported within the Local_Codec_List. The Local_Codec_List shall be updated and resent as soon as these local resource conditions have changed, if the BSS wants a have these new conditions considered within the Codec Mismatch Resolution or Codec Optimisation.

Whenever a new Distant_Codec_List or a new Local_Codec_List becomes available, then the BSS shall attempt to resolve the Codec_Mismatch or optimize the Codec_Type as soon as possible by following the rules outlined in Table 19 and shall perform a subsequent intra cell handover to the new Local_Used_Codec.

Table 19: Rules for Resolving Codec Mismatch

	Radio Leg 2 (T2)											
Radio Leg 1 (T1)	EFR FR HR	EFR FR	EFR HR	EFR	FR EFR HR	FR EFR	FR HR	FR	HR EFR FR	HR FR	HR EFR	HR
EFR FR HR	=	=	=	=	T2=EFR	T2=EFR	T1=FR	T1=FR	T2=EFR	T1=FR T2=FR	T2=EFR	T1=HR
EFR FR		=	=	=	T2=EFR	T2=EFR	T1=FR	T1=FR	T2=EFR	T1=FR T2=FR	T2=EFR	MIS
EFR HR			=	=	T2=EFR	T2=EFR	T1=HR T2=HR	MIS	T2=EFR	T1=HR	T2=EFR	T1=HR
EFR				=	T2=EFR	T2=EFR	MIS	MIS	T2=EFR	MIS	T2=EFR	MIS
FR EFR HR					T1=EFR T2=EFR	T1=EFR T2=EFR	=	=	T1=EFR T2=EFR	T2=FR	T1=EFR T2=EFR	T1=HR
FR EFR						T1=EFR T2=EFR	=	=	T1=EFR T2=EFR	T2=FR	T1=EFR T2=EFR	MIS
FR HR							=	=	T2=FR	T2=FR	T1=HR	T1=HR
FR								=	T2=FR	T2=FR	MIS	MIS
HR EFR FR									T1=EFR T2=EFR	T1=FR T2=FR	T1=EFR T2=EFR	=
HR FR										T1=FR T2=FR	=	=
HR EFR											T1=EFR T2=EFR	=
HR												=

The first column of Table 19 contains in each cell a definition of the Used_Codec in Radio Leg 1 followed on the next line by the list of supported Codecs. The first row contains similar information for the other, Radio Leg 2. The matrix elements indicate the change to be made in the Used_Codec. For example T2=HR means that Radio Leg 2 shall use the Half Rate Codec. The grey shaded area is intentionally left blank, since it would contain redundant information. The ´=´ sign indicates that no mismatch is present. The ´MIS´ indicates that mismatch can not be resolved. The light (green) shaded areas represent no Codec mismatch, but in several cases double sided handover is recommended to gain speech quality.

Annex A (Normative): Inband Signalling Protocol: Generic Structure

Scope

Inband Signalling Messages (IS Messages) can be used to construct a specific IS Protocol for the communication between telecommunication entities for various purposes. The original purpose is to establish tandem free operation of mobile-to-mobile calls in GSM networks. The IS Messages provide communication channels inside the speech signal paths between the speech transcoders.

In addition IS Messages allow the control of equipment within the speech signal paths between these telecommunication entities (e.g. speech transcoders). These equipments are termed „In Path Equipments" (IPEs).

Annex A defines the generic structure of these IS Messages and rules for the IS_Sender.

Annex B defines the generic rules with respect to these IS Messages for the IPEs.

Annex A is mandatory for TFO_TRAU Equipment and informative for IPEs.

Annex B is informative for TFO_TRAU Equipment.

Annex B shall be followed by IPEs, which want to be compatible to IS Messages.

A.1 Generic Structure of Inband Signalling Messages

All IS Messages follow a set of design rules, or a generic structure, which allow to identify and bypass them by IPEs without detailed knowledge of the IS Protocol served. The principle of the IS Protocol shall in that sense be future proof: it can be enhanced and extended to other applications without modifying the IPEs.

The IS Messages replace some of the LSBs of the PCM samples of the Speech, Audio or Modem signal.

By construction the introduced signal distortion is practically inaudible in case of Speech signals.

Modem signals will in most cases not be affected with respect to their data transmission performance.

A.1.1 Frequency and Order of Bit Transmission

IS Messages are transferred within the Least Significant Bit (LSB) of PCM samples on 64 kBit/s links, by replacing the LSB of every 16th consecutive PCM sample with one bit of the IS Message (16_PCM_Sample_Grid).

This is equivalent to an average bit rate of 10 bit per 20 ms or 500 bits per second. See Figure 12:

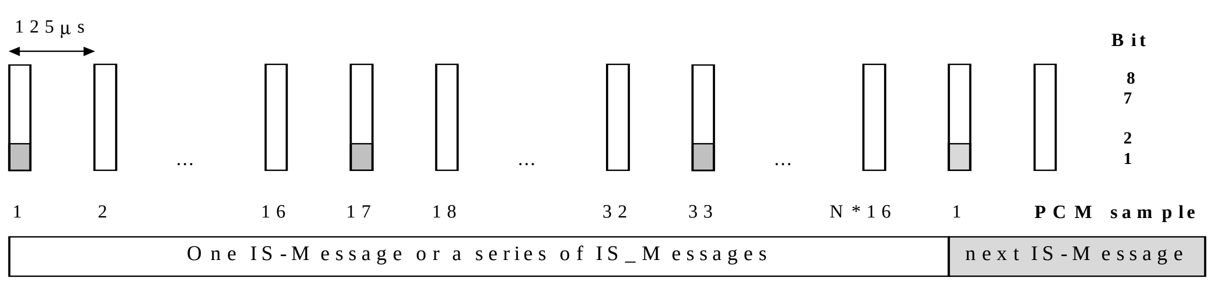

Figure 12: Inband Signalling Structure

A vertical bar denotes an 8-bit PCM sample, the shadowed box in bit 1 (LSB) represents an inserted bit of the IS-Message.

By definition each IS Message "occupies" an integer multiple of 16 PCM samples. Especially the 15 PCM samples after the last inserted bit of an IS Message "belong" still to that IS Message.

All IS Messages, whichever type, have by construction “0”-Bits at every 10th position, starting with position 1, 11, 21 and so on. This “0”-Bits occur therefor regularly every 20 ms and may be used for synchronization purposes.

Each IS Message consists of an IS_Header followed by an IS_Command_Block. Most IS Messages have a number of further IS_Extension_Blocks. Figure 13 shows an example with two IS_Extension_Blocks.

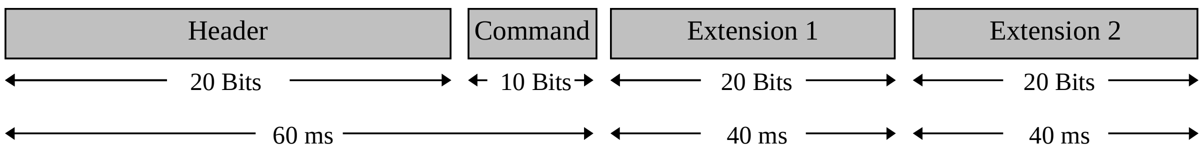

Figure 13: Example for IS Message with two IS_Extension_Blocks

The MSB of each constituent field is transmitted first. The IS_Header is transmitted first, followed by the IS_Command_Block and - if applicable - any further IS_Extension_Block(s).

By construction all IS Messages do have lengths of integer multiples of 10 bits, thus occupying integer multiples of 160 PCM samples, thus lasting integer multiples of 20 ms. The shortest IS Message has a length of 60 ms.

A.1.2 IS_Header

The IS_Header consists of a 20-Bit long sequence, as defined in Figure 14:

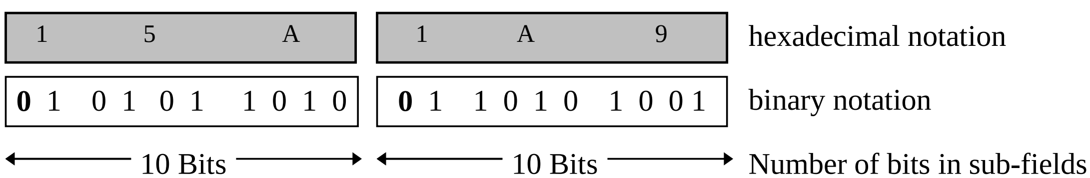

Figure 14: Structure of the 20 bit IS_Header

A.1.3 IS_Command_Block

The IS_Command identifies the IS Message and/or serves for the control of IPEs. The names of the IS_Commands and their codes in hexadecimal notation in the IS_Command_Block are given in the Table 20.

Table 20: Defined IS_Commands

Index	Command	Code	Meaning / Action
		hexadecimal Nibble 1-3	
0	reserved	0x000	no extension
1	**REQ**	0x05D	Denotes an IS_REQ Message, with extension
2	**ACK**	0x0BA	Denotes an IS_ACK Message, with extension
3	**IPE**	0x0E7	Denotes an IS_IPE Message, with extension, i.e. an IS_TRANS or the IS_NORMAL Message
4	**FILL**	0x129	Denotes the IS_FILL Message, no extension
5	**DUP**	0x174	Denotes the IS_DUP Message, no extension
6	**SYL**	0x193	Denotes the IS_SYL Message, no extension
7	reserved	0x1CE	no extension

All other values are reserved for future use.

Each IS_Command is protected by the binary, systematic (9,3) block code with generator polynomial $g(x) = x^6 + x^4 + x^3 + x^2 + 1$. The minimum Hamming distance of this code is dmin = 4, which allows the correction of up to one bit error within each code word of length 9 bits.

The first bit (MSB) of the IS_Command_Block is defined to be "0", for synchronisation purposes, see Figure 15.

Table 20 gives the hexadecimal notation of the complete IS_Command_Block.

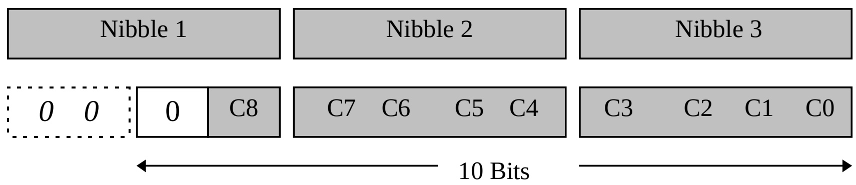

Figure 15: General Construction of an IS_Command_Block

A.1.4 IS_Extension_Block(s)

Most IS Messages have one or more IS_Extension_Block(s). Each IS_Extension_Block is 20 bits long and shall consist of two "0"-Synchronization_Bits at position 1 (MSB) and 11, a 16-bit Information_Field (split into two fields of 9 and 7 bits, respectively) and a 2-bit Extension_Field (EX), see Figure 16:

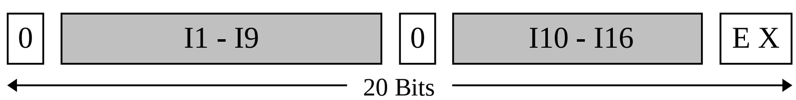

Figure 16: General Construction of an IS_Extension_Block

The Extension_Field indicates if an other IS_Extension_Block is following (EX :="1.1") or not (EX := "0.0").

All other codes are reserved. This may be used to detect transmission errors within the Extension_Field.

A.2 Detailed Specification of IS Messages

A.2.1 IS_REQ Message

With the IS_REQ Message an IS_Sender can test, if there is an IS Partner and indicates that it is willing to negotiate.

IS_REQ is used to initiate the IS Protocol or to indicate changes in the configuration, etc.

IS_REQ has at least one IS_Extension_Block, containing the IS_System_Identification. (see A.5).

Other IS_Extension_Blocks may follow, see Figure 17.

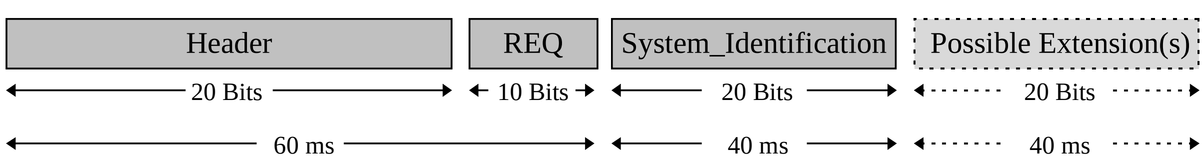

Figure 17: General Construction of an IS_REQ Message

In general an IS_REQ Message shall be as short as possible. Special care must be taken in the design of the IS_Extension_Blocks to avoid audible effects, since sometimes an IS_REQ Message may be transmitted for quite some time (several seconds).

A.2.2 IS_ACK Message

With the IS_ACK Message an IS Partner typically answers an IS_REQ Message or an IS_ACK Message. It can also be used to submit further information to the other IS Partner. IS_REQ and IS_ACK are the main message types between IS Partners.

The IS_ACK has at least an IS_Extension_Block containing the IS_System_Identification (see A.5).

Other IS_Extension_Blocks may follow, see Figure 18.

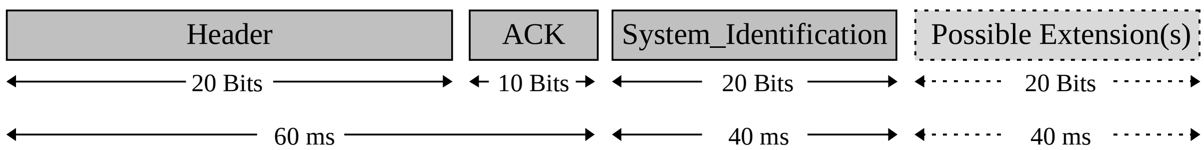

Figure 18: General Construction of an IS_ACK Message

No specific design constraints with respect to audibility exist, since IS_ACK is typically not sent very often.

A.2.3 IS_IPE, IS_TRANS and IS_NORMAL Messages

The IPE command denotes IS_IPE Messages. An IPE shall always look for this type of message and follow the instruction. An IS_Sender shall use this IS_IPE Message to command all IPEs into a specific mode of "Bit Transparency".

This Message has one IS_Extension_Block, indicating the requested IPE_Mode. See Figure 19.

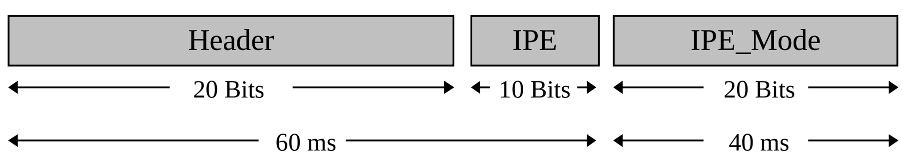

Figure 19: General Construction of an IS_IPE Message

No specific design constraints with respect to audibility exist, since IS_IPE is typically not sent very often.

Table 21 defines 16 out of 32 possible IPE_Commands. The other codes are reserved for future extensions.

Table 21: Defined IPE_Modes

Index	IPE_Mode	Code	MEANING / ACTION
		hexadecimal Nibble 1 - 5	
0	Normal	0x00000	Normal Operation
1	Trans_1_u	0x044DC	pass 1 LSB; 7 upper Bits are used
2	Trans_2_u	0x089B8	pass 2 LSBs; 6 upper Bits are used
3	Trans_3_u	0x0CD64	pass 3 LSBs; 5 upper Bits are used
4	Trans_4_u	0x11570	pass 4 LSBs; 4 upper Bits are used
5	Trans_5_u	0x151AC	pass 5 LSBs; 3 upper Bits are used
6	Trans_6_u	0x19CC8	pass 6 LSBs; 2 upper Bits are used
7	Trans_7_u	0x1D814	pass 7 LSBs; 1 upper Bit is used
8	Transparent	0x22CE0	Full Transparent Mode for all eight bits
9	Trans_1	0x2683C	pass 1 LSB; 7 upper Bits are free and unused
10	Trans_2	0x2A558	pass 2 LSBs; 6 upper Bits are free and unused
11	Trans_3	0x2E184	pass 3 LSBs; 5 upper Bits are free and unused
12	Trans_4	0x33990	pass 4 LSBs; 4 upper Bits are free and unused
13	Trans_5	0x37D4C	pass 5 LSBs; 3 upper Bits are free and unused
14	Trans_6	0x3B028	pass 6 LSBs; 2 upper Bits are free and unused
15	Trans_7	0x3F4F4	pass 7 LSBs; 1 upper Bit is free and unused
16	reserved	0x41D1C	reserved
17..31	reserved	reserved	reserved

The IPE_Mode is protected by the binary, systematic (16,5) block code with generator polynomial g(x) = x^11 + x^7 + x^5 + x^4 + x^2 + x + 1. The minimum Hamming distance of this code is dmin=7, which allows the correction of up to 3 bit errors within each code word of length 16 bits.

Bits 1 (MSB) and 11 are the synchronisation bits and set to "0", see Figure 20. The EX field is set to "0.0" in all currently defined IPE_Modes, i.e. no further IS_Extension_Block is following.

Table 21 defines the coding in hexadecimal notation for the complete IPE_Mode_Extension_Block, with EX := 00.

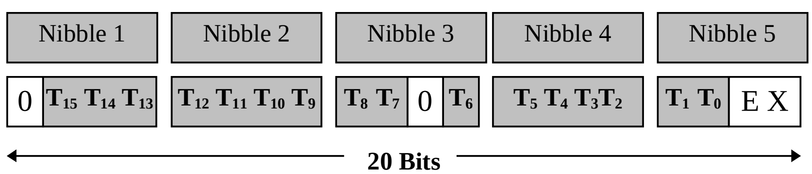

Figure 20: IPE_Mode_Extension_Block for the IS_IPE Message

An IS_ IPE Message containing the NORMAL command is termed **IS_NORMAL Message**.

An IS_ IPE Message containing a TRANS_x command is termed **IS_TRANS_x Message**.

An IS_ IPE Message containing a TRANS_x**_u** command is termed **IS_TRANS_x_u Message.**

The latter two are sometimes also termed **IS_TRANS Message**, if the details are not important.

The behaviour of IPEs, when receiving such commands, is described in Annex B.

The first IS Message in a series is often "swallowed" by IPEs (see Annex B). An IS_IPE Message must therefor never be the first message of a series of IS Messages, i.e. it shall be sent as an isolated IS Message or after a (sufficiently long) uninterrupted IS Protocol.

A.2.4 IS_FILL Message

The IS_FILL Message has no IS_Extension_Block and no specific meaning. An IS_ Sender can use the IS_FILL Message to fill a temporary gap in the protocol flow. This may be important to keep all IPEs in synchronization and open for further IS Messages. See Figure 23. An IS_FILL Message shall also be used by the IS_Sender to resynchronize all IPEs in case of a phase shift of the Keep_Open_Indication.

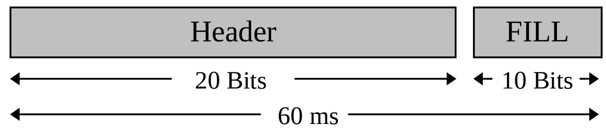

Figure 21: Construction of the IS_FILL Message

IS_FILL is designed in a way that multiple repetitions cause minimal audible effects.

A.2.5 IS_DUP Message

The IS_DUP Message may be used between IS Partners to indicate an half duplex mode. It may be especially important in Handover situations. The IS_DUP Message has no IS_Extension_Block, see Figure 22.

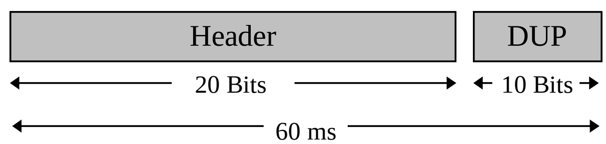

Figure 22: Construction of the IS_DUP Message

A.2.6 IS_SYL Message

The IS_SYL Message may be used between IS Partners to indicate the loss of synchronisation. It may be especially important in Handover situations. The IS_SYL Message has no IS_Extension_Block, see Figure 22.

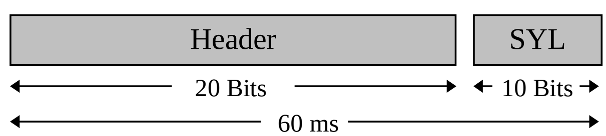

Figure 23: Construction of the IS_SYL Message

A.3 Keep_Open_Indication

In Transparent_Mode, i.e. after properly receiving an IS_TRANS Message, all IPEs shall monitor the bypassing bit stream for the Keep_Open_Indication (definition see below). If this Keep_Open_Indication is not seen for some time, then the IPEs shall fall automatically back into normal operation, i.e. the mode of operation before the IS_TRANS Message.

This automatic fall back shall have the same effect as the IS_NORMAL Message would have.

By definition the Keep_Open_Indication is a continuous bit stream of one "0"-Bit in the LSB of every 160^{th} PCM sample, i.e. every 20 ms. At least one "1"-Bit must be present within the LSBs of the other 159 PCM samples. See Figure 24.

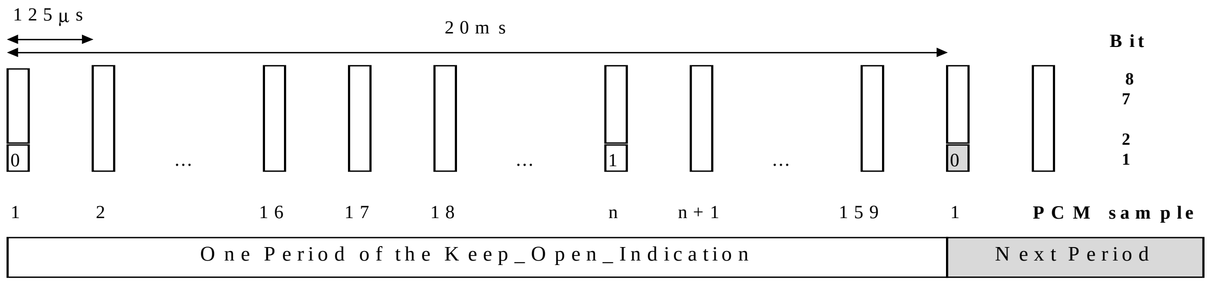

Figure 24: Keep_Open_Indication

The "0"-Bit stream of the Keep_Open_Indication shall always be present as long as the IPEs need to be in Transparent_Mode.

The Keep_Open_Indication shall be in phase with the preceding IS Messages., i.e. the first bit of the Keep_Open_Indication shall be in the position of the first bit of the (hypothetical) next IS Message. In fact, the IS Messages themselves contain this Keep_Open_Indication by definition.

In case of a known phase shift of the Keep_Open_Indication, the IS_Sender has to send at least one IS Message, which defines the new phase position of the Keep_Open_Indication. If no other IS Message is to be sent, then the IS_FILL Message shall be used. If an IS Message longer than 160 ms is scheduled for transmission, then an IS_FILL Message should be inserted before, to guarantee fast resynchronization of the IPEs.

A.4 Rules for Sending of IS Messages

IS Messages replace some bits of the PCM samples and therefor cause a minimal signal distortion. Therefore IS Messages shall be used with care and not longer than necessary. The IS Protocol is kept to a minimum to avoid unnecessary complexity. One basic assumption is that only one IS Protocol is active at a time between two IS Partners.

Only specific telecommunication entities shall be allowed to initiate IS Protocols. They are called **IS_Active** or active IS Partners. In principle these shall only be terminal devices or their "representatives" within the network. Examples are ISDN-Terminals, Speech-Servers, TRAUs (in GSM as representatives of the MSs).

Other telecommunication entities shall only react on IS Protocols. They are called **IS_Passive**. Most IPEs are of this type. They bypass the IS Messages, they obey the IS_IPE Messages, but they never initiate IS Messages.

Other telecommunication entities are IS_Passive by default. But if they receive IS Protocols that they can understand, then they may become IS_Active and start to initiate IS Protocols. They thus become active IS Partners and shall take care that only one IS Protocol is active on both of their sides. They are called **IS_Responsive.** Examples are GCMEs.

Active IS Partners shall send

either continuous sequences of IS Messages without interruption of the 16_PCM_Sample_Grid:

- or isolated IS Messages with same message lengths;
- or isolated IS Messages with sufficient distance between them, if shorter IS Messages follow longer IS Messages.

The latter case is important, because shorter isolated IS Messages travel faster through IPEs than longer ones, see annex B.

As said above, after initialization of an IS Message sequence, no interruption of the 16_PCM_Sample_Grid shall occur within the sequence. Adjustments of the phase position of the Keep_Open_Indication shall be done only after the IS_TRANS Message by inserting the necessary number n (with $0 < n < 160$) of "1" Bits (termed "T_Bits") into the LSBs of the PCM samples that have to be skipped. The first PCM sample for this insertion of T_Bits is the one where the next regular IS Message or next regular Keep_Open_Indication would begin. At the new phase position the next IS Message or the IS_FILL Message shall be sent, to allow IPEs to resynchronize fast. See Figure 25.

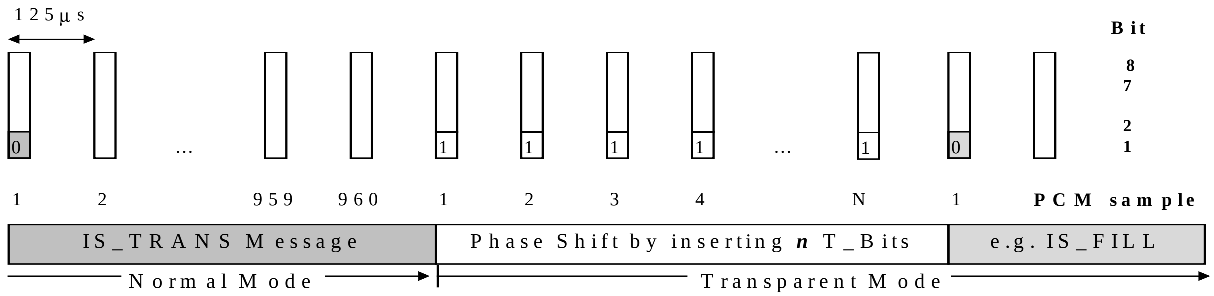

Figure 25: Phase Shift of the 16_PCM_Sample_Grid by inserting T_Bits

Similarly, the adjustment of the phase between two Keep_Open_Indications shall be done by inserting the necessary number of T_Bits and by sending an IS Message - preferably, but not necessarily - the IS_FILL.

Finally a "negative" phase adjustment between two Keep_Open_Indications shall be allowed by shortening the cycle by a maximum of 2 PCM samples and sending an IS Message (see above) at the new phase position.

A.5 IS_System_Identification_Block

The IS_System_Identification_Block is a mandatory IS_Extension_Block for the IS_ACK and IS_REQ messages with the 16-bit Information_Field containing the IS_System_Identification. It identifies the system within which the message is generated. Table 22 shows the defined IS_System_Identification codes.

Table 22: Defined IS_System_Identification Codes

System	Code (in hex)
GSM	either 0x53948, if EX == "0.0" or 0x5394B, if EX == "1.1"
	reserved

The only defined code so far is GSM_Identification, see also Figure 26.

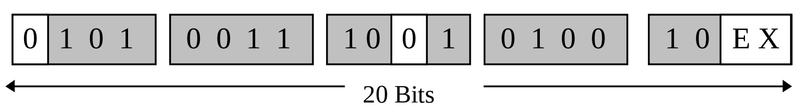

Figure 26: IS_System_Identification for GSM

All other codes are reserved. Further IS_System_Identification Codes for other systems shall be defined in a way that the audibility is minimal and the hamming distance to the already defined once is maximal.

The IS_System_Identification is protected by the binary, systematic (16,8) block code with generator polynomial g(x) = x^8 + x^7 + x^6 + x^4 + x^2 + x + 1. The minimum Hamming distance of this code is dmin=5, which allows the correction of up to 2 bit errors within each code word of length 16 bits.

Code word 0x0000 is per definition used for GSM.

The resulting 16 bits are placed into the IS_System_Extension_Block as shown in Figure 26 and then the whole 20 bit word is additionally EXORed with the fixed code word 0x53948 to minimise audible effects.

Annex B (Informative): In Path Equipment: Generic Rules and Guidelines

Scope

Inband Signalling Messages (IS Messages) can be used to construct a specific IS Protocol for the communication between telecommunication entities for various purposes. The original purpose is to establish tandem free operation of mobile-to-mobile calls in GSM networks. The IS Messages provide communication channels inside the speech signal paths between the speech transcoders.

In addition IS Messages allow the control of equipment within the speech signal paths between these telecommunication entities (e.g. speech transcoders). These equipments are termed „In Path Equipments“ (IPEs).

Annex A defines the generic structure of these IS Messages and rules for the IS_Sender.

Annex B defines the generic rules with respect to these IS Messages for the IPEs.

Annex A is mandatory for TFO_TRAU Equipment and **informative for IPEs**.

Annex B is informative for TFO_TRAU Equipment.

It shall be followed by IPEs, which want to be compatible to IS Messages.

B.1 Types of In Path Equipment

The term "In Path Equipment" (IPE) is used for any telecommunication equipment within the (64 kBit/s) transmission path for the speech signal between two entities, which want to communicate via IS Messages, i.e. the IS Partners.

In modern telecommunication networks most of these IPEs are digitally transparent for the complete 64 kBit/s data stream all the time after call establishment until call release. These IPEs are optimal and need no consideration here.

Some IPEs are most of the time digitally transparent, but disturb the link every now and then. Examples are:

- switches, which interrupt the link during Handover;
- switches, which insert a kind of conference bridge for a short while during Handover;
- links, which do octet deletions or insertions (octet slips);
- DTMF generators, which insert DTMF tones sometimes for a short while; and more.

- Other IPEs are digitally transparent in one direction, but not in the other. Examples are:
  - DTMF generators (again), which insert the DTMF tones only in one direction;
  - Network Echo Cancellers (NEC), which let the signal pass unaltered towards the PSTN, but cancel the echo; and more.

Other IPEs are semi-transparent, i.e. let most or some of the bits pass, but not all. Examples are:

A/µ_Law - converters;

- µ/A_Law converters; and
- especially the tandem connection of A/µ_Law and µ/A_Law converters, or vice versa.

links, which insert inband signalling by bit stealing (T1 links); and more.

Other IPEs are not transparent at all to the digital bit stream, although the speech signal pass more or less unaltered. Examples:

- level shifters, which adjust the signal levels, e.g. between national networks;
- DCMEs (Digital Circuit Multiplication Equipment), which compress the bit stream by encoding/decoding the speech signal for cost efficient transmission; and more.

Many of these IPEs - for some time - will be not compliant with the IS Message principle described above. The IS Messages will not pass these non-compliant IPEs or not in both directions, or not always. Care must be taken to identify situations where IPEs are part-time-transparent or semi-transparent, when applying IS Messages. Other IPEs - at some point in time in the future - will be compliant to the IS Message principle. The rules they have to fulfil are described below.

B.2 IS_Compliant IPEs

B.2.1 Typical IPEs are IS_Passive

General: An IPE shall ***never*** actively initiate the exchange of IS Messages. The active initiation is only done by terminals or their "representatives". This avoids uncontrolled and unnecessary fluctuation of IS Messages within the network.

Most IPEs shall never actively respond to IS Messages by sending other IS Messages. They are called ***IS_Passive***.

They need not and do not understand the IS Protocol, but let it just pass unaltered and obey the relevant IS_IPE Messages.

Some IPEs may, however, respond on received IS Messages, modify these and/or respond with own IS Messages, ***if*** they understand the IS Protocol and can take or bring advantage to the overall system performance or system quality. These IPEs are called ***IS_Responsive***. Examples are GSM-specific Digital Circuit Multiplication Equipments (GCMEs), which reduce transmission costs without degrading the speech quality. These IPEs may be able to step into the IS Protocol, interpret and respond to it and modify the speech signal in an system_compliant way. Thus they become ***IS_Active*** Partners themselves.

B.2.2 IS Message_Transparency

When commanded into a Transparent Mode, the IPEs are fully transparent at least for the LSBs in all PCM samples. Therefore the following rules are needed only and only do apply for the IPEs, when in Normal_Mode:

IPEs shall let the IS Messages bypass, respectively re-insert them, from their input to their respective output.

They shall not alter them, nor do any kind of error correction. Exceptions are the IS_Responsive IPEs.

B.2.2.1 First IS Message

During its **Normal_Mode** an IS_Compliant IPE shall always monitor the incoming PCM data stream for the occurrence of the IS_Header sequence. If the IS_Header is detected after a period without IS Messages, the IPE shall store the following IS_Command and IS_Extension_Block(s). During reception of this first IS Message, the normal operation of the IPE is maintained with the consequence that the first IS Message may not appear at the output of the IPE.

B.2.2.2 IS Messages within a Sequence

All further IS Messages which follow directly after the first detected IS Message in the same phase position shall be passed unaltered to the output of the IPE with exactly that delay the IPE would later introduce when commanded into Transparent_Mode by one of the IS_TRANS commands, see Figure 27.

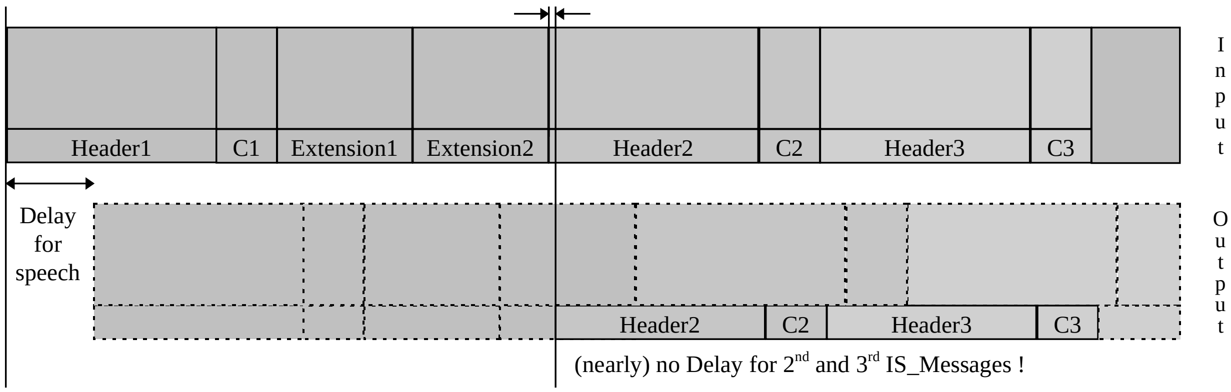

Figure 27: Transparency and Delay for first and following IS Messages

The upper row symbolizes the speech signal at the input of the IPE, with the PCM samples drawn vertically and the IS Messages inserted into the LSBs. The lower row symbolizes the speech signal at the output of the IPE. The vertical lines denote the boundaries of the IS Message elements.

Figure 27 shows an example where the first IS Message is detected, but not passed through. The distortion caused by the first IS Message is still "somehow" there (indicated by the empty dashed boxes in the LSB), but the message is destroyed. The second and third IS Messages are passed through unaltered. Note, however, that the delay of the speech signal is (in this example) substantially higher than the delay of the IS Messages. They travel faster than the speech signal through this IPE.

B.2.2.3 Isolated IS Message

In cases where the first detected IS Message is not immediately followed by further IS Messages, the IPE shall insert this first IS Message (which the IPE has stored) into its output PCM bit stream, with exactly the delay and phase position a second IS Message would have, see Figure 28, which shows an example where an isolated IS Message is travelling through an IPE.

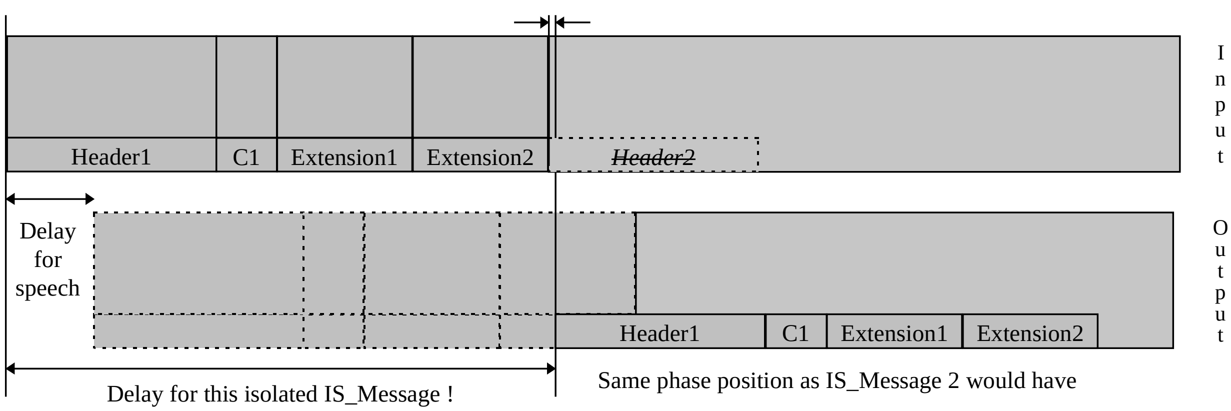

Figure 28: Transparency and Delay for an isolated IS Message

Note that the delay of an isolated IS Message is depending on its own length! Longer IS Messages will have more delay, shorter less. It could - in principle - happen that a second, shorter isolated IS Message would "bypass" the first longer IS Message - with the consequence that the first one would be destroyed. This is especially important when there are several IPEs in the path, since the delay effects accumulate. Therefore it is not allowed to send shorter isolated IS Messages too close after longer IS Messages. IS Messages with same length have no restriction.

In summary: the first IS Message in a series of IS Messages is "swallowed" by an IPE, while all the following IS Messages pass unaltered and with minimal delay. If an IS Message occurs isolated, then it is not swallowed, but delayed by exactly its own length. The latter mechanism ensures that isolated IS Messages can pass through an unlimited number of IPEs.

B.2.2.4 Check if IS Message is following

The checking, whether an other IS Message is following or not is done "on the fly", i.e. bit by bit. This is possible due to the fact that all messages begin with exactly the same IS_Header. The decision, whether an IS Message is an isolated message or the first message in a series, can be done latest after the last bit of the (next) IS_Header. See Figure 28.

Consequently: after detection of the first IS Message, the IS_Header is in any case inserted at the output in the correct position, regardless, whether a second message follows or not.

B.3 IPE State Representation

Concerning the IS Protocol, an IPE can be described with five major States in two main Modes, where the States describe the IPE with respect to the IS Protocol and the Modes describe the IPE with respect to the operation on PCM data. Figure 29 shows a graphical representation of the State diagram of an IPE.

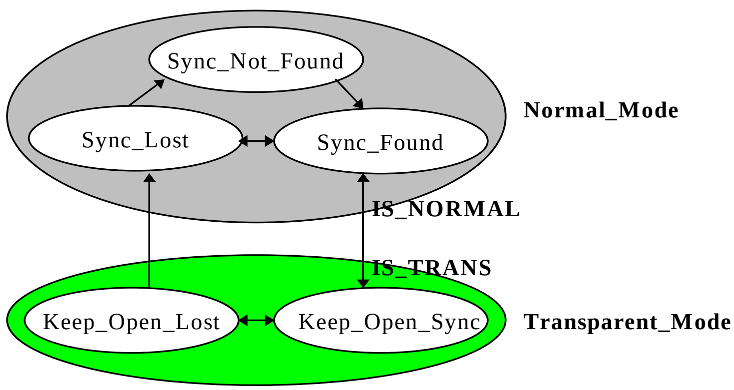

Figure 29: Principle of a State Diagram of an IPE

Some Definitions:

An IS Message shall be recognized as "*error-free*", if no error can be detected, neither within the IS_Header, nor in the IS_Command nor in any IS_Extension_Block.

An IS Message shall be recognized as "*single-error*", if no more than one bit position differs in the IS_Header or the IS_Command_Block or the IPE_Mode_Block or one EX-field or one Sync bit.

An IS Message shall be recognized as "*correctable*", if the phase position is as in preceeding IS Messages and:

- no more than 2 bit position differs in the IS_Header; and
- no more than 1 error is detected within the IS_Command_Block; and
- no more than 3 errors are detected within the IPE_Mode_Block; and
- no more than 0 error is detected within the EX-field(s); and
- no more than 1 error is detected within the Sync-Bit(s); and
- the total number of detected errors is not higher than 3.

IS Messages, which are error-free, single-error or correctable are also called "*valid*" IS Messages.

An IS Message shall be recognized as "*present*", if the phase position is as in preceeding IS Messages and:

no more than 4 bit position differs in the IS_Header and

- no more than 2 errors are detected within the IS_Command_Block; and
- no more than 3 errors are detected within the IPE_Mode_Block; and
- no more than 1 error is detected within the EX-field(s); and
- no more than 2 errors are detected within the Sync-Bit(s); and
- the total number of detected errors is not higher than 4.

Sequences, which differ in more than "present" are not recognized as IS Messages at all ("not_present").

Note that the insertion of T_Bits may change the phase position of an IS Message. The IS Message shall in that case be classified after the removal of the T_Bits.

An octet slip may also change the phase position of an IS Message. If an error-free or a single-error IS Message can be found after considering a hypotetical octet slip (±1 sample), then it may be regarded as error-free or single-error and the new phase position shall be regarded as valid, if no valid or present IS Message can be found at the old phase position.

B.3.1 IPE in Sync_Not_Found

After start-up or after a long interruption of the IS Protocol an IPE is in Normal_Mode, performing its normal operation. IS Messages have not been found and consequently no bypassing of IS Messages is performed.

The algorithm for initial synchronization shall be able to detect each single IS Message, especially the first or an isolated one. An IPE shall always, during Normal_Mode and during Transparent_Mode, search for the IS_Header and consequently for complete IS Messages. When found, it can be assumed that with high probability the following IS Messages and the Keep_Open _Indication will stay within the found "grid"or "phase" of every 16^{th} PCM sample, the ***16_PCM_Sample_Grid***.

An IPE transits from Sync_Not_Found into Sync_Found, if and only if an error_free IS Message is detected. Then the IPE lets the following IS Messages bypass, as described above.

If the first IS Message is an error_free IS_TRANS Message, then the IPE transits directly into the Transparent_Mode.

B.3.2 IPE in Sync_Found

The IPE continues its normal operation, but opens an "***IS_Door***" every 16th LSB for the bypassing IS Messages.

An IPE shall regard sync as continued, i.e. stay in Sync_Found, if after each IS Message another valid IS Message follows within the same phase position, i.e. within the 16_PCM_Sample_Grid.

For any deviations from a valid IS Message, the IPE transits to Sync_Lost.

If an error_free or correctable IS_TRANS is received in Sync_Found, then the IPE transits into the Transparent_Mode.

B.3.3 IPE in Sync_Lost

In Sync_Lost, an IPE shall search for IS Messages on all positions as for initial synchronisation. In parallel, an IPE shall bypass not_valid, but present IS Messages at the found phase position for a maximum of one second. An IPE shall close the IS_Door after that, if no valid IS Message is following, i.e. transit into Sync_Not_Found.

A single valid IS Message brings the IPE back into Sync_Found.

As soon as the IPE detects in Sync_Found or in Sync_Lost a single or more deviations from an error_free IS Message, then the IPE may optionally open the IS_Door also at positions ±1 around the present (0) phase position for a maximum of one second] to allow other IPEs in the path for parallel re-synchronization. See Figure 30. The IPE may try to find a continuation of the disturbed IS Message at these 3 positions. If the IPE can detect an error-free or a single-error IS Message in this way, then it shall accept the new phase position, if no IS Message can be found at the old phase position anymore.

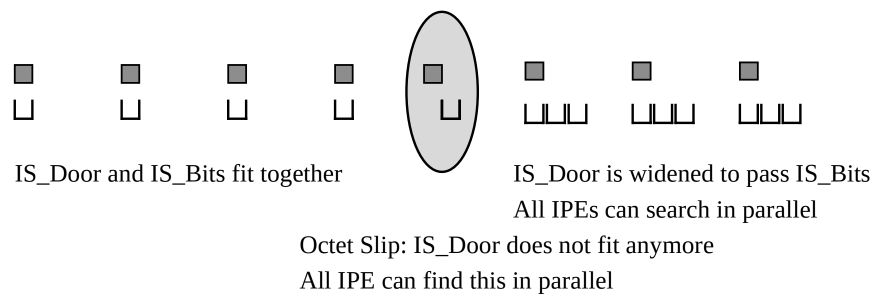

Figure 30: Handling of octet slip for fast and parallel re-synchronization of all IPEs (optional)

B.3.4 IPE in Keep_Open_Sync

The IPE enters this State by receiving a valid IS_TRANS Message. This is the main State of the Transparent_Mode.

It depends on the specific IPE, if this Transparent_Mode is active only for the commanded direction (that is the default assumption) or in both directions (because for a specific IPE it might be useless or impossible to maintain Normal_Mode in one direction and Transparent_Mode in the other one).

The IPE shall bypass the commanded LSBs and handle the upper bits accordingly (IPE specific).

The IPE shall search in parallel for IS_IPE Messages (IS_TRANS, IS_NORMAL) and

transit - if necessary - to Normal_Mode or an other Transparent_Modes (other number of transparent LSBs).

The IPE shall monitor the bypassing bit stream for the Keep_Open_Indication and accept the Keep_Open_Indication only at the phase position defined by the preceding IS Message.

If the Keep_Open_Indication is not seen anymore then the IPE transits into Keep_Open_Lost.

B.3.5 IPE in Keep_Open_Lost

The IPE shall continue its operation in Transparent_Mode and Keep_Open_Lost for a maximum of one second before it shall return to Normal_Mode.During that time the IPE shall try to resynchronize either by finding an IS Message or by finding the Keep_Open_Indication at positions ±1 and 0 around the present phase position (handle of Octet Slip).

The IPE may take advantage of the fact that T_Bits are inserted or deleted by the IS_Sender in case of an intentional phase adjustment.

An IS Message at any arbitrary phase position followed by a valid Keep_Open_Indication is accepted as re-defining the Keep_Open phase position, if and only if the Keep_Open_Indication is no longer present at the old phase position.
A Keep_Open_Indication at a phase position ±1 PCM sample interval around the old phase position is accepted as re-defining the Keep_Open phase position, if and only if the Keep_Open_Indication is no longer present at the old phase position.

The Keep_Open_Indication is ***valid***, as long as at least 40 "0"-Bits are seen at the correct positions within a sliding window of length of one second. At least one "1"-Bit must be seen in between each pair of the expected "0"-Bits.

B.4 IPE Error Handling

The **first** IS_Message shall only be accepted, if there is no detectable error.

For all following IS_Messages it shall apply:

Errors in IS Messages shall be passed unaltered through the IPEs. This shall hold for all IS Messages.

Only error-free or correctable IS_IPE Message shall be applied by the IPE to its own operation. Other IS_IPE Messages shall be ignored, but bypassed.

B.5 IPE Transmission Delay

The transmission delay introduced by an IPE for the speech, audio or modem signal is in general different in Normal_Mode and Transparent_Mode. Some IPEs may have several different Normal_Modes with possibly different signal delays. IS Messages are transmitted within the regular 16_PCM_Sample_Grid. It is important that this regularity is not disturbed. Therefor care must be taken at the transition between these modes.

The transmission delay of a specific IPE is in general lower for IS Messages than for speech, audio or modem signals.

B.5.1 IPE Transmission Delay in Normal_Mode

The delay for IS Messages in Normal_Mode shall be identical to the delay in that Transparent_Mode, that follows after the first IS_TRANS Message. If different Transparent_Modes with different delays could follow, then the shortest delay of all possible Transparent Modes shall be selected for IS Messages in Normal_Mode.

If an IPE in Normal_Mode has to change its transmission delay, then this shall not affect the delay of the IS Messages.

B.5.2 IPE Transmission Delay in Transparent_Mode

In the majority of all cases the IPE will keep the transmission delay for the IS Messages in Normal_Mode also in Transparent_Mode for the transmission of the commanded transparent LSBs. IPEs which do not understand the IS Protocol shall never modify the transparent bits, so they are also not allowed to change delay.

Some IPEs, which understand a specific IS Protocol, may have even different Transparent_Modes and also here the transmission delays may differ. Examples are GCMEs.

If an IPE has to change its transmission delay at the transition from Normal_Mode to Transparent_Mode, then the IPE shall readjust the phase of the Keep_Open_Indication after transition into the Transparent_Mode with higher delay by inserting the relevant number of T_Bits after the first IS_TRANS Message and before the next IS Message. If no other IS Message is following, then the IS_FILL shall be inserted, obeying all other relevant rules of the specific IS Protocol (e.g. EMBED bit C5 in TFO Frames).

If an IPE has to change from one Transparent_Mode to an other one with a different transmission delay, then the IPE shall readjust the phase of the Keep_Open_Indication after transition into the new Transparent_Mode by inserting the relevant number of T_Bits. If no other IS Message is following, then the IS_FILL shall be inserted at the new phase position to mark the new grid position of the 16_PCM_Sample_Grid and to allow other IPEs to resynchronize, obeying all other relevant rules of the specific IS Protocol (e.g. EMBED bit C5 in TFO Frames).

B.6 Compliance to IS Messages

An IS_Compliant IPE shall be capable of interpreting and obeying the IS_IPE Messages.

It depends on the intelligence and task of an IPE, how many and which of the other IS Messages it needs to understand.

The IPEs shall synchronise to all IS Messages, especially to find or refind the Keep_Open_Indication. All IPEs shall resynchronize, if they see an IS Message in a new phase position, and if the synchronization can not be found in the old phase position anymore.

B.6.1 Compliance to IS_REQ and IS_ACK Messages

Most IPEs need not and do not understand these messages. They just synchronise to them and let them pass unaltered.

Only IS_Responsive IPEs may take advantage. This is system specific and IPE specific.

B.6.2 Compliance to IS_NORMAL Message

The IPE shall act in response to the receipt of an IS_NORMAL Message such that:

The IPE shall synchronise to it. The message shall appear unchanged at the output of the IPE.

The IPE shall resume its Normal_Mode of operation for all data received subsequent to the IS_NORMAL Message, until a different command is received.

It depends on the type and operation of the specific IPE, whether the Normal_Mode is resumed in both directions, or only in the direction in which the IS_NORMAL Message flows. It must be assumed that in general only this one direction is affected.

B.6.3 Compliance to IS_TRANS_x Messages

The IPE shall act in response to the receipt of an IS_TRANS_x Message (x in the range 1 to 8) such that:

The IPE shall synchronise to it. The IS_TRANS_x Message shall appear unchanged at the output of the IPE.

The IPE shall be transparent in all x LSBs of all PCM samples received subsequent to the IS_TRANS Message.

The transparency shall persist as long as the Keep_Open_Indication persists, or until a different command is received.

The (8-x) upper bits of the PCM samples are not of interest and may be modified arbitrarily by the IPE.

It depends on the type and operation of the specific IPE, whether the Transparent_Mode is resumed in both directions, or only in the direction in which the IS_TRANS Message flows. It must be assumed that in general only this one direction is affected.

B.6.4 Compliance to IS_TRANS_x_u Messages

The IPE shall act in response to the receipt of an IS_TRANS_x_u Message (x in the range 1 to 7) such that:

The IPE shall synchronise to it. The messages shall appear unchanged at the output of the IPE.

The IPE shall be transparent in all x LSBs of all PCM samples received subsequent to the IS_TRANS Message.

The transparency shall persist as long as the Keep_Open_Indication persists, or until a different command is received.

The (8-x) upper bits of the PCM samples are important and in general shall not be modified by the IPE, but shall be bypassed transparently in exactly the same manner and delay as the x LSBs. It is important that this transparency for the upper bits is provided by IPEs that do not understand the specific IS Protocol (e.g. do not understand the IS_System_Identification or the protocol of the transmitted parameters).

Only IPEs which do ***exactly*** understand the specific IS Protocol shall take advantage of the opportunities given with the IS_TRANS_x_u Messages. An example is the GCME, which transmits internally only the coded speech parameters and re-generates the upper x bits at its output (termed here as "first solution"). The resulting delay in the upper 8-x bits shall be identical to the delay in the x LSBs.

If this transparency of the upper (8-x) bits or their re-generation can not be established, then the upper bits shall contain a constant pattern, giving the least output energy (PCM_Silence). This "second solution" may cause temporary interruptions of the speech signal in some transition cases (e.g. hand over in some tandem free GSM mobile-to-mobile calls). Therefore the first solution is the preferred one.

IPEs, which implements the second solution shall switch to the full transparent 64 kBit/s channel as soon as they loose synchronisation with the protocol of the transmitted parameters (e.g. the "TFO Frames" in GSM Systems). The full transparency shall be executed for both directions. The near side shall be fully transparent in less than 60 ms and the other side the one way delay of that IPE later.

It depends on the type and operation of the specific IPE, whether the Transparent_Mode is resumed in both directions, or only in the direction in which the IS_TRANS Message flows. It must be assumed that in general only this one direction is affected.

B.6.5 Compliance to IS_FILL Message

The IS_FILL Message has no specific meaning, but may serve for two purposes.

First of all, it can be used to close the gap in an IS Protocol to keep all IPEs synchronized. Otherwise - in case of an interruption - the ***n*** IPEs in the path would swallow the next ***n*** IS Messages again.

Second, an IS_FILL Message can be used to resynchronize all IPEs to a new grid position, if necessary.

B.6.6 Compliance to IS_DUP Messages

The IS_DUP Message is sent by an IS Partner to the distant IS Partner to inform about a specific Half_Duplex reception.

Most IPEs need not and do not understand this message. They just synchronize to it and let it pass unaltered.

Only IS_Responsive IPEs may take advantage. This is system specific and IPE specific.

B.6.7 Compliance to IS_SYL Messages

The IS_SYL Message is sent by an IS Partner to the distant IS Partner to inform about a specific Sync_Lost Situation.

Most IPEs need not and do not understand this message. They just synchronize to it and let it pass unaltered.

Only IS_Responsive IPEs may take advantage. This is system specific and IPE specific.

Annex C (Informative): The SDL model of the TFO protocol

This annex contains a few *selected* pages from the formal SDL model of the protocol for Tandem Free Operation described in the main body of this standard. The *complete* SDL specification, *which is fully simulateable*, is available in various electronic formats as described below.

The SDL model gives a precise description of the *logical* behaviour of the TFO protocol. It is not intended to imply a particular way of implementing the protocol nor does it intend to restrict an implementation only to what is specified in the SDL.

This is not a real-time model and critical timing requirements have not been included. These are fully described in the main text of this standard, for example clause 7. The purpose of this SDL specification is to give a clear and unambiguous understanding of the TFO protocol with respect to the temporal ordering and interchange of TFO messages over the A-interface.

The SDL specification models the TFO messages as described in clause 6, the TFO processes as described in clause 8 and the TFO protocol as described in clauses 9 and 10. Additionally, it illustrates the use of Table 19 (clause 11) for resolving codec mismatch. In the case of a conflict between the SDL model and clauses 9 and 10, clause 9 and 10 shall have precedence.

The SDL model is available in electronic format in the zip archive **TFO_SDL.zip**. This archive can be found on the ETSI CD-ROM together with the TFO standard.

TFO_SDL.zip contains the following files:

- **README.txt**

  how to install and use the simulateable model

- **TFO_PDF.pdf**

  the complete SDL specification in graphical format as a .pdf file

- **TFO_CIF.pr**

  the complete SDL specification in machine processable format as a .pr file

- **TFO_SDT**

  a directory containing all the SDT (version 3.2) source files

If you have any questions related to the SDL model please contact: *pex@etsi.fr*

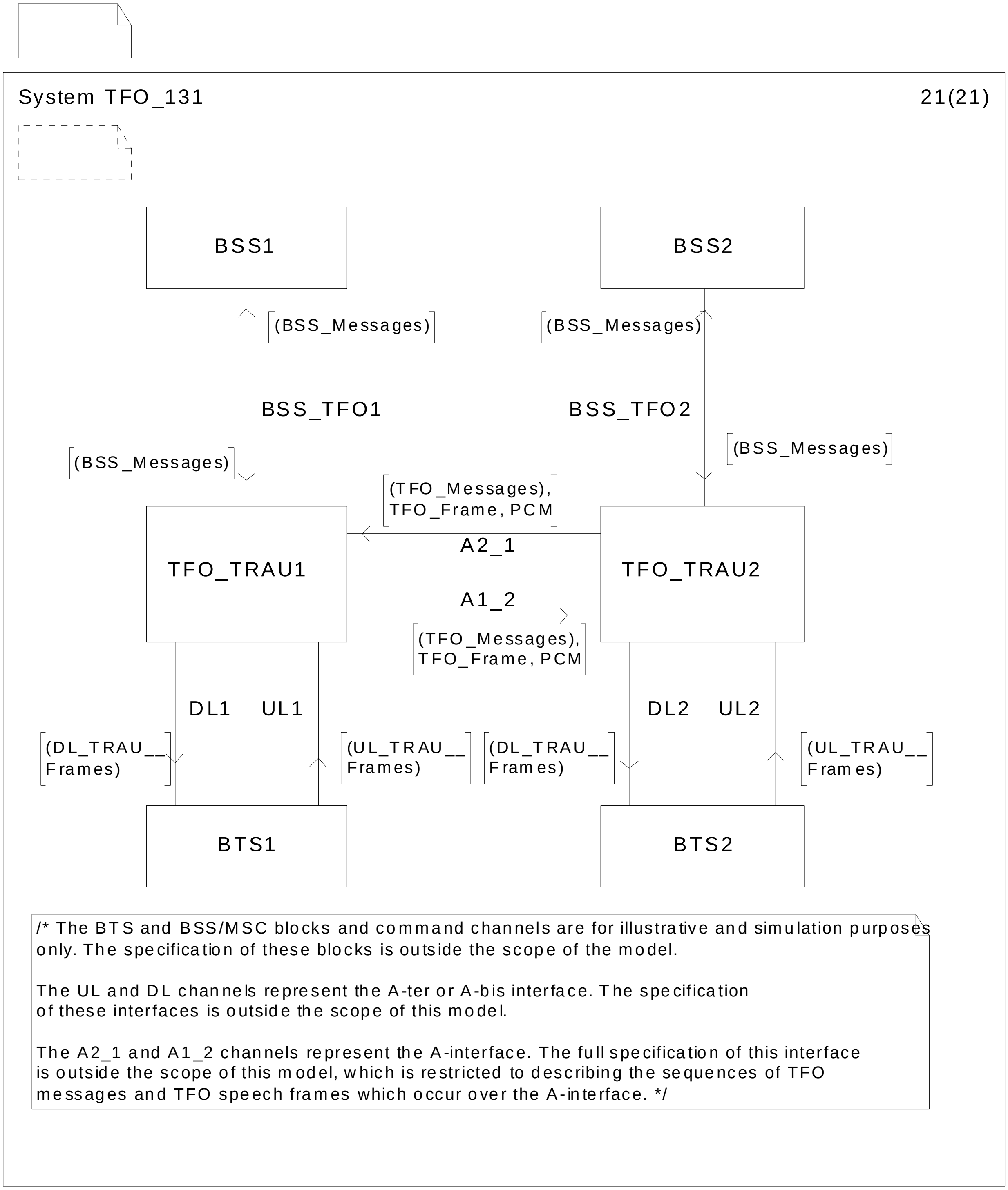

Figure C.1: Overall SDL model structure

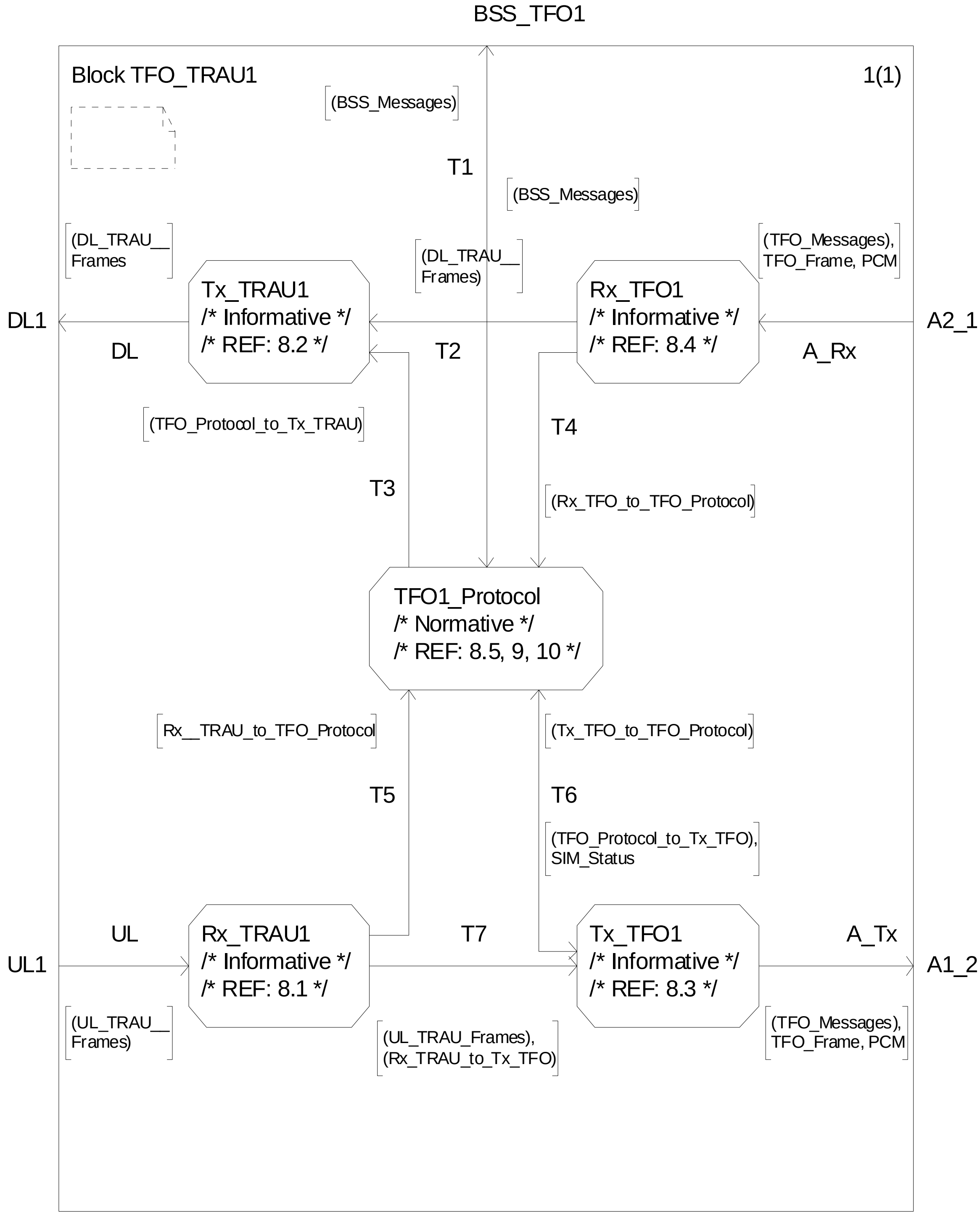

Figure C.2: TFO/TRAU SDL process diagrams

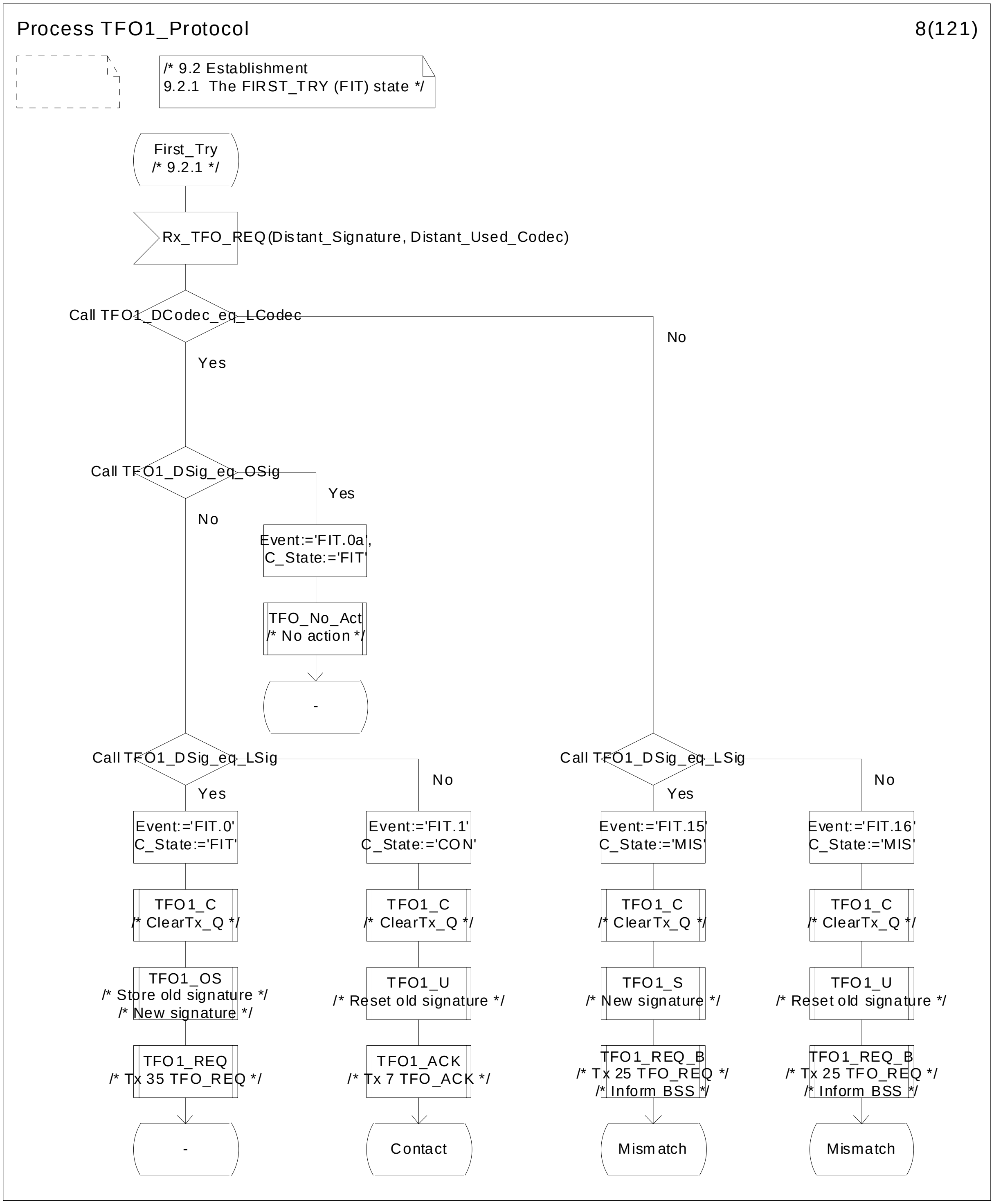

Figure C.3: Partial TFO protocol transition (taken from the First_Try state)

The complete SDL for the TFO messages, TFO process and the TFO protocol transitions corresponding to the protocol matrix given in clause 10 can be found in the electronic SDL files.

Annex D (informative): Change history

Change history							
Date	**TSG #**	**TSG Doc.**	**CR**	**Rev**	**Subject/Comment**	**Old**	**New**
March 00	SA#7	SP-000026	A002	1	TFO Message Extendibility	7.0.0	7.1.0

History

Document history		
V7.1.0	April 2000	Publication
V7.1.1	August 2000	Publication