Upper layers / L1
Circuit Switched interfaces

L1M_GS011-1

Ver 3.4

File: L1doc\dsp_mcu\GSM_GPRS\specs\L1M_GS011_1.doc

Department: Application Specific Product / Wireless Communications System

	Originator	Approval	Quality
Name	A. VALLAURI	J.L PEYRE	
Date	09/09/99		
Signature			

HISTORY

Version	Date	Author	Notes
Ver: 0.3	31/03/99	A. VALLAURI	1
Ver: 0.4	06/04/99	A. VALLAURI	2
Ver: 0.5	25/05/99	K. MATTHES	3
Ver: 0.6	10/06/99	K. MATTHES	4
Ver: 0.7	08/07/99	K.MATTHES	5
Ver: 0.8	09/09/99	J.L PEYRE	6
Ver: 0.9	10/04/99	J.L PEYRE	7
Ver: 1.0	11/16/99	J.L PEYRE / K. MATTHES	8
Ver: 1.1	06/29/00	F. BONAVITA	9
Ver: 1.2	08/08/00	F. BONAVITA	10
Ver: 1.3	09/01/00	F. BONAVITA	11
Ver: 1.4	10/04/00	J.L Peyre / K. Matthes / P.Martinez / F.Olivero	12
Ver: 1.5	11/22/00	F. Bonavita/ J.L Peyre / F.Mazard	13
Ver: 1.6	04/25/01	J.L Peyre / F. Bonavita	14
Ver: 1.7	10/09/01	F. Bonavita / F. Mazard	15
Ver: 1.8	21/11/01	F. Mazard	16
Ver: 1.9	05/12/01	K. MATTHES	17
Ver: 1.10	06/11/02	M. Campos	18
Ver: 1.11	03/11/03	M. Vanin	19
Ver: 1.12	19/02/04	Fady Labib	20
Ver: 2.0	16/03/04	Fady Labib	21
Ver: 2.1	03/06/04	Fady Labib	22
Ver: 2.2	07/20/04	N. Benech	23
Ver: 2.3	09/03/04	M.Vanin	24
Ver: 2.4	11/10/04	Fady Labib	25
Ver: 3.2	26/10/04	Fady Labib	26
Ver: 3.3	04/12/04	Krishnaprasad Madathil	27
Ver: 3.4	06/22/05	Francois Mazard	28

NOTES :

1. Draft following the meeting End of March.
2. Added history field.
   Added comments concerning “Read Non-Serving Cell BCCH”.
   Added comments within “Read Non-Serving Cell SCH” concerning the Neighbor cell read control.
3. Renamed this spec, now called “Upper layers / L1 circuit switched interface”.
   Added dedicated mode interface (cut/paste from S920 v2.1).
   Split the “Read non serving cell SCH” into 2 interfaces: “Cell selection network synchronization” and “Read non serving cell SCH in Idle mode”.
   “Read non serving cell BCCH” interface is finalized: we use the bitmap approach.
   Corrected block/bit order in T_MPHC_CBCH_UPDATE_REQ.
   Corrected typing mistake within “Read non serving cell BCCH” message summary table (MPHC_STOP_NCELL_BCCH_REQ is now using type T_MPHC_STOP_NCELL_BCCH_REQ).
   Changed ***ARFCN*** into ***radio_freq*** all over the document.
   Replaced all old data types by the UWORD8/WORD8/UWORD16/WORD16/…
   All remaining MPH5 or MPH messages have been renamed MPHC.
   Dedicated Ncell Sync interface result message has been aligned with idle mode message (MPHC_NCELL_SYNC_IND).
   Completed FACCH/DL interface with function prototype.
   Corrected FACCH/DL and DATA interfaces: now use type API (chipset dependant type).
   Enhanced L2data/L1 interface figure within DATA interface.
   Added layer structure figure within chapter INTERFACES.
   Include flow chart diagrams.
4. Correction of new cell selection interface (MPHC_NEW_SCELL_REQ): The bsic has to be passed instead of the tsc as this information is needed by the DSP for sending the RACH burst.
   Correction of the example for circuit switched message flow in Appendix A.
   Version in line with L1 release 0x1321.
5. Correction of dedicated measurement interface (MPHC_MEAS_REPORT) concerning the types of accumulated power results (cf. problem report TI_15). Precision on GSM reference documents.
   Correction of typing errors. Alignment of TXPWR management I/F with L1 object code (config).
6. **Section 8.1**: moved **training_sequence_code** to **tsc**.
   Section 9: reword definition of **ba_id** in MPHC_RXLEV_PERIODIC_REQ: ba_id is an identifier for BA_LIST.
   Section 0: radio_freq is type (UWORD16), ba_id is a sequence message number matching the MPHC_RXLEV_PERIODIC_REQ ba_id identifier.
   Section 10.1: Add comments: 1) on the shared memory to use between L3 and L1 for the measurement results: MPHC_RXLEV_REQ/IND message put back to L1 only a pointer, 2) before the first request, L3 needs to clean the structure. There is no procedure in L1 to detect the first request.
   Section 11.1: T_MPHC_CONFIG_CBCH_REQ detailled and corrected
   Section 12.1: Editorial correction: tsc -> bsic
   Section 14.3: Editorial correction: **rf_chan_cnt** defined in **frequency_list_bef_sti** structure.
   T_BCCH_LIST structure described.
   Section 18.3: Editorial correction: MPH5_ -> MPHC_.

Section 21: Editorial correction made in figure.
Section 30: in **T_MPHC_UPDATE_BA_LIST** structure **num_of_chans** can be coded on 1 Byte.
Section 32.3: Editorial correction.

7. **Section 30:** For Automatic Gain Control Algorithm reason, the Serving Cell Beacon radio frequency value must be always included in the BA list (cf. Problem Report PB512.doc TI_22).
8. **Section 2.2:** Insert "*time_alignment*" and "*fn_offset*" definition whilst we are in Idle mode.
   Section 10.1: Add a "note" to explain that full list measurement process timing is reduced in L1 (various Pm can be made per TDMA, up to 4 in current implementation, more for next RF generation), it then becomes a L3 responsibility to manage the timing of the full list measurement process to be in a range of [3, 5s]. Remark: This is not applicable to current L1 implementation, due to the fact that L1 hasn't been modified (cf Pb512.doc, TI_57 still OPEN)
   Section 38: Update of TST_SLEEP_REQ for ULPD S/W.
9. **Section 6:** specify the message MPHC_NCELL_BCCH_REQ is allowed in Packet Transfer mode.
   Section 30: specify the messages MPHC_NCELL_FB_SB_READ and MPHC_NCELL_SB_READ are allowed in Packet Transfer mode.
10. **Section 6:** add the priority parameter in the message MPHC_NCELL_BCCH_REQ
11. **Section 8:** Power measurement request is also used for the Extended measurements in GPRS.
    Section 4: Add the ccch_lev field in the message MPHC_DATA_IND
    Section 6: Rename the "priority" field in "gprs_priority". Specify this field exists only with a GPRS software. Only the first BCCH neighbor read is sent back to L3.
12. Fill "APPROVAL" field.
    Added comment for MPHC_RXLEV_PERIODIC_REQ message structure.
    Add comments on the order of call of dll_data_ul and dll_data_dl functions (IDS interface 39.2).
    Section **Error! Reference source not found.**: Audio functions updated with OMEGA dependencies (Analog Based Band VEGA is no more mantained).
    T_MPHC_MEAS_REPORT: "ncell_meas.A[33].rxlev_acc" is now signed: WORD16
    Reserved information has been added in the following messages: T_MPHC_DATA_IND, T_MPHC_NCELL_SYNC_IND, T_MPHC_MEAS_REPORT
13. Section 36: Add description of TST_TEST_HW_REQ message
    Section 22: Add description of MPHC_NETWORK_LOST_IND message.
    Section 32.3: "sb_flag" and "bsic" in MPHC_NCELL_SB_READ message are not needed. In order to avoid a change in L3/L1 interface these parameters are kept but it's specified that they are invalid. Remove comment "(not used for MPHC_NCELL_SB_READ)" alongside both fn_offset and time_alignment.
    Section 11.1: update MPHC_CONFIG_CBCH_REG message (TS is inside [0..3] and Subchaneel is not applicable with this message)
    Section 37.2: update Auxilary conversion with number of ADC result up to 9 with OMEGA and change the type of adc_result[] from UWORD8 to UWORD16. Update ADC according to the ADC ReWork.
    Delete the AEC and Keybeep interface. These interfaces are defined in the S916 specification.
14. Section 2.2 and Section 7.3 "error_flag" field has to be changed by "sb_flag". Correct definition of "sb_flag".

Section 32:bad massage name: replace MPHC_NCELL_SB_IND by MPHC_NCELL_SYNC_IND Add "appendix B" at end of document.

Section 37.2.2: add comments in the message

15. Section 37.2.2: add ADC with CS_MODE0
    Section Network synchronization lost: add comment about ADC in No GSM mode.
    Section 21: add the new "Off Mode" chapter.
    Section 39: Remove this chapter and move it to S916 specifcation (CHG1756).

16. Update thefollowing messages for the AMR feature:
    Section 15: MPHC_ASSIGN_CHANNGEL_REQ.
    Section 16: MPHC_ASYNC_HO_REQ.
    Section 18: MPHC_SYNC_HO_REQ.
    Section 20: MPHC_PRE_SYNC_HO_REQ.
    Section 28: MPHC_CHANNEL_MODE_MODIFY_REQ.
17. Added message for multi-band support:
    Section 39: MPHC_INIT_L1_REQ, MPHC_INIT_L1_CON
    Update of message identifiers in Appendix B.
18. Added messages for Neighbor cells monitoring:
    Section 7: MPHC_NCELL_LIST_SYNC_REQ
    Update message for EOTD feature
    Section 7.2 : MPH_NCELL_SYNC_IND
    Create new section for dedicated Ncell SYNc interface (up to 12 neighbor cells)
    Section 33: dedicated Ncell SYNC interface (up to 12 neighbor cells)
19. Added range for timing advance value validity.
20. **Section 6.1** : Corrected the range of modulus in MPHC_SCELL_NBCCH_REQ to be (1-65535) rather than (1-65536)
    Section 9 : corrected the range and usage of next_radio_freq_measured, the correct range is (0-32) rather than (1-33).
    Section 10 : removed the line about extended measurement as the MPHC_RXLEV_REQ is not allowed in dedicated mode.
    Section 30 : Updated the description and the message sequence diagram for updating BA list in dedicated mode, removed the description of the old implementation (immediate BA list download mechanism)
    Section 39.1: replaced the old reference S820 by the new name L1_GS800
21. Changed the File path to match the new path under clearcase
    Replaced “noise_suppression_control_bit” by “noise_suppression_bit” to match the code
    Replaced “powerclass_gsm” by “powerclass_band1” to match the code
    Replaced “powerclass_dcs” by “powerclass_band2” to match the code
    Section 18.3 : added note for **nci** location in the structure
    Section 20.3: added note for **timing_advance_valid** and **timing_advance** location in the structure
22. **section 26.3** : updated this section with the new option for dll_dcch_downlink() where the frane number is passed to the protocl stack if the compilation option SEND_FN_TO_L2_IN_DCCH was enabled this was needed at protocol stack side (CQ 20076).
23. **Section 5**: Added 5.2 MPHC_STOP_CCCH_REQ section with description of implicit stop conditions
    Section 6: Added 6.4 MPHC_STOP_BCCH_REQ section with description of implicit stop conditions
24. **Section 32**: added of the STOP_DEDICATED interfaces old&new .
    Appendix B: L1 message identifier has been updated with the latest TCS211_L1_1444_0 version.
25. **Section 34**: added new value to support loop (I) .
26. Changed the version document to be v3.2 in order to match the clear case labeling process. A clear case labeling mistake occurred between version 2.3 and v2.4.
27. Updated the document so that the following messages are sent immediately after the corresponding event has been programmed and not after the actual occurrence of that event:
    Section 14,30,31 and Appendix A: MPHC_IMMED_ASSIGN_CON.
    Section 15,31: MPHC_CHANNEL_ASSIGN_CON.
    Section 16,17,18,20,22: L1C_DEDIC_DONE.
    Section 27: MPHC_CHANGE_FREQUENCY_CON.
    Section 28: MPHC_CHANNEL_MODE_MODIFY_CON.

Section 32 : MPHC_ASYNC_HO_CON.
Section 34 : OML1_CLOSE_TCH_LOOP_CON.
Section 35: OML1_START_DAI_TEST_CON.

28. Stop dedicated confirmatiom message "**MPHC_STOP_DEDICATED_CON"** is also applied to TCS3.2

Table of content

INTRODUCTION *13*

1. INTERFACES *14*

1.1. Primitives **15**

1.2. Radio Frequency Coding **16**

2. Cell Selection Network synchronisation *17*

2.1. MPHC_NETWORK_SYNC_REQ **18**

2.2. MPHC_NETWORK_SYNC_IND **19**

3. Off Mode *21*

4. Network synchronization lost *22*

5. Read Common Control Channel Info (CCCH) *23*

5.1. MPHC_START_CCCH_REQ **25**

5.2. MPHC_STOP_CCCH_REQ **26**

6. Read Serving Cell BCCH *27*

6.1. MPHC_SCELL_NBCCH_REQ **30**

6.2. MPHC_SCELL_EBCCH_REQ **31**

6.3. MPHC_DATA_IND **32**

6.4. MPHC_STOP_BCCH_REQ **33**

7. Read Non-Serving Cell SCH in Idle mode *34*

7.1. MPHC_NCELL_SYNC_REQ **43**

7.2. MPHC_NCELL_LIST_SYNC_REQ **44**

7.3. MPHC_NCELL_SYNC_IND **46**

7.4. MPHC_STOP_NCELL_SYNC_REQ **48**

8. Read Non-Serving Cell BCCH *50*

8.1. MPHC_NCELL_BCCH_REQ **53**

8.2. MPHC_STOP_NCELL_BCCH_REQ **54**

9. Periodically Report Cell Signal Strengths *55*

MPHC_RXLEV_PERIODIC_REQ **58**

MPHC_RXLEV_PERIODIC_IND **58**

10. Power Measurement Request 60
10.1. MPHC_RXLEV_REQ / IND 63
11. Configure and Read Cell Broadcast Channel 64
11.1. MPHC_CONFIG_CBCH_REQ 68
11.2. MPHC_CBCH_SCHEDULE_REQ 69
11.3. MPHC_CBCH_INFO_REQ 70
11.4. MPHC_CBCH_UPDATE_REQ 71
11.5. MPHC_STOP_CBCH_REQ 72
12. New Serving Cell Selection 73
12.1. MPHC_NEW_SCELL_REQ 74
13. Connection Establishment 75
13.1. MPHC_RA_REQ 78
13.2. MPHC_RA_CON 79
14. Dedicated Immediate Assignment Interface 80
14.1. Prerequisites 80
14.2. Figure comments 80
14.3. Message Structures 82
15. Dedicated Channel Assignment Interface 84
15.1. Prerequisites 84
15.2. Figure comments 84
15.3. Message Structures 86
16. Asynchronous Handover Procedure (Successful) 91
16.1. Prerequisites 91
16.2. Figure comments 92
16.3. Message Structures 93
17. Asynchronous Handover Procedure (Failure) 98
17.1. Prerequisites 98
17.2. Figure comments 99
17.3. Message Structures 99
18. Synchronous Handover Procedure (Successful) 100

18.1. Prerequisites 100
18.2. Figure comments 101
18.3. Message Structures 101
19. Synchronous Handover Procedure (Failure - TA out of range) 103
19.1. Prerequisites 103
19.2. Figure comments 103
19.3. Message Structures 104
20. Pre-Synchronous Handover Procedure (Successful) 105
20.1. Prerequisites 105
20.2. Figure comments 106
20.3. Message Structures 106
21. Pre-Synchronous Handover Procedure (Failure - TA out of range) 108
21.1. Prerequisites 108
21.2. Figure comments 108
21.3. Message Structures 109
22. Handover Failure Recovery (all cases) 110
22.1. Prerequisites 110
22.2. Figure comments 110
22.3. Message Structures 111
23. SACCH Uplink Interface 112
23.1. Prerequisites 113
23.2. Figure comments 113
23.3. L2 Function prototype 113
24. SACCH Downlink Interface 114
24.1. Prerequisites 114
24.2. Figure comments 114
24.3. Message Structures 115
25. FACCH/DCCH Uplink Interface 117
25.1. Prerequisites 117
25.2. Figure comments 118

25.3. L2 Function prototype 118
26. FACCH/DCCH Downlink Interface 119
26.1. Prerequisites 119
26.2. Figure comments 120
26.3. L2 Function prototype 120
27. Dedicated Frequency Redefinition Interface 121
27.1. Prerequisites 121
27.2. Figure comments 121
27.3. Message Structures 123
28. Dedicated Channel Mode Modify Interface 125
28.1. Prerequisites 125
28.2. Figure comments 125
28.3. Message Structures 126
29. Dedicated Set Ciphering Interface 128
29.1. Prerequisites 128
29.2. Figure comments 128
29.3. Message Structures 129
30. Dedicated Power Measurement Interface 130
30.1. Prerequisites 132
30.2. Figure comments 132
30.3. Message Structures 134
31. Stop Dedicated Mode 137
31.1. Message Structures 140
32. Dedicated NCell Sync Interface (up to 6 neighbor cells) 141
32.1. Prerequisites 141
32.2. Figure comments 141
32.3. Message Structures 145
33. Dedicated NCell Sync Interface (up to 12 neighbor cells) 146
33.1. Prerequisites 146
33.2. Messages structures 148

34. TCH Loopback 149
34.1. Prerequisites 149
34.2. Figure comments 149
34.3. Message Structures 150
35. DAI Testing 151
35.1. Prerequisites 151
35.2. Figure comments 151
35.3. Message Structures 152
36. Hardware and DSP tests 153
36. Hardware and DSP tests 153
36.1. TST_TEST_HW_CON 154
37. Custom Functions 155
37.1. Transmit Power Management 155
37.1.1. Fixed Transmit power and RAMP shapes 155
37.1.2. GSM controlled Transmit power with according RAMP shapes 156
37.2. Auxiliary conversions 159
37.2.1. ADC execution algorithm within L1 164
37.2.2. MMI_ADC_REQ 167
37.2.3. CST_ADC_RESULT 168
38. Power Management Interface 169
39. Data application Interface 171
39.1. General 171
39.2. L1 / L2data Interface 172
39.3. Use of L1 / L2data Interface for Rate Adaptation (RA) 177
39.4. Use of L1 / L2data Interface for RLP/FCS calculation 177
39.5. Use of L1 / L2data Interface for FAX 177
40. Multiband interface 179
40.1. MPHC_INIT_L1_REQ 179
Appendix A: Example of circuit switch message flow 182
Appendix B : L1 message Identifier 185

List of figures

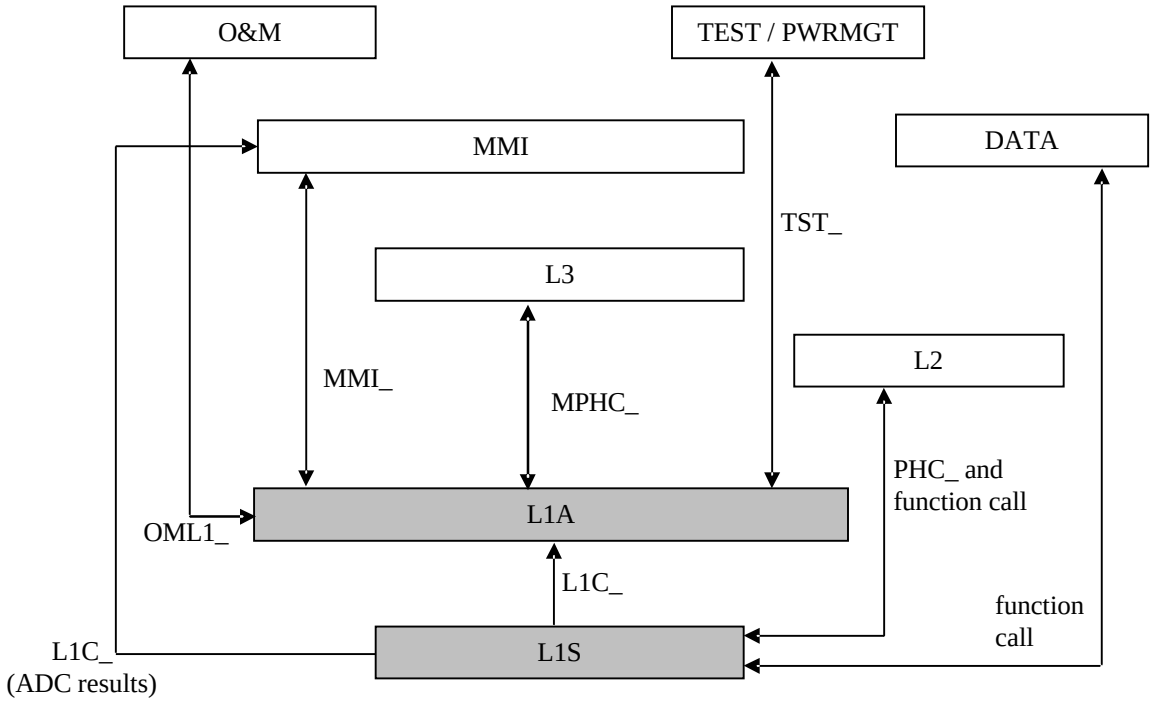
INTRODUCTION

This document specifies the interface of Texas Instruments Layer 1 code in all the areas to allow mapping to any upper layers (L2/L3/…) codes.

The Interface section details the interface messages and procedures between TI Layer 1 and upper layers.

1. INTERFACES

This section describes the revised interface required to enable TI's Layer 1 code to interface to the upper layers. The following figure shows the different communication interfaces specified within this document. Each of them is shown with its assigned message prefix.



1.1. Primitives

The following definitions apply to interface messages.

MPHC primitives: Circuit switched subsystem primitives. These primitives are either commands from L3 to L1 (MPHC_****_REQ) or results from L1 to L3 (MPHC_***_CON or MPHC_***_IND). They are implemented through OS message.

MPHP primitives: Packet subsystem primitives. These primitives are either commands from L3 to L1 (MPHP_****_REQ) or results from L1 to L3 (MPHP_***_CON or MPHP_***_IND). They are implemented through OS message. (Note: this is just for information since this document is not specifying the packet subsystem)

PHC primitives: Circuit switched subsystem primitives. Those primitives are used in the interface between L2 and L1. They are implemented through OS message or function call.

MMI primitives: These primitives are used for commands from MMI to L1 (MMI_****_REQ). They are implemented through OS message.

OML1 primitives: These primitives are used for commands from “Operation and Maintenance” layer to L1 (OML1_****_REQ) and for confirmation messages (OML1_****_CON) in the other direction. They are implemented through OS message.

L1C primitives: Packet subsystem primitives. These primitives are used for result messages from L1S to L1A or L1S to MMI (only for ADC results). They are implemented through OS message.

TST primitives: These primitives are used for commands from “TEST and POWER Management” layer to L1 (TST_****_REQ). They are implemented through OS message.

1.2. Radio Frequency Coding

The radio frequency coding within Layer 1 is given by the following table. All subsequent references to “radio_freq” are made with respect to this coding convention. The “radio_freq” coding within Layer1 is dependent on the standard. For dual band modes and E-GSM the “radio_freq” coding has been chosen so as to allow using it as a table index for power measurements. This concept is explained in the following section.

Standard	Compiler Switch STD	ARFCN coding according to ETSI GSM spec.	“radio_freq” coding used in L1
GSM	1	1-124	1-124
E-GSM	2	1-124 975-1023,0	1-124 125-174
PCS 1900	3	512-810	512-810
DCS (GSM 1800)	4	512-885	512-885
DUAL BAND GSM/DCS	5	1-124 512-885	1-124 125-498
DUAL BAND E-GSM/DCS	6	1-124 975-1023,0 512-885	1-124 125-174 175-548

2. Cell Selection Network synchronisation

Direction	Message name	Type
L3->L1	**MPHC_NETWORK_SYNC_REQ**	**T_ MPHC_NETWORK_SYNC_REQ**
L3<-L1	**MPHC_ NETWORK _SYNC_IND**	**T_MPHC_ NETWORK _SYNC_IND**
L3->L1	**MPHC_STOP_NETWORK _SYNC_REQ**	**Trigger**
L3<-L1	**MPHC_STOP_NETWORK _SYNC_CON**	**Trigger**

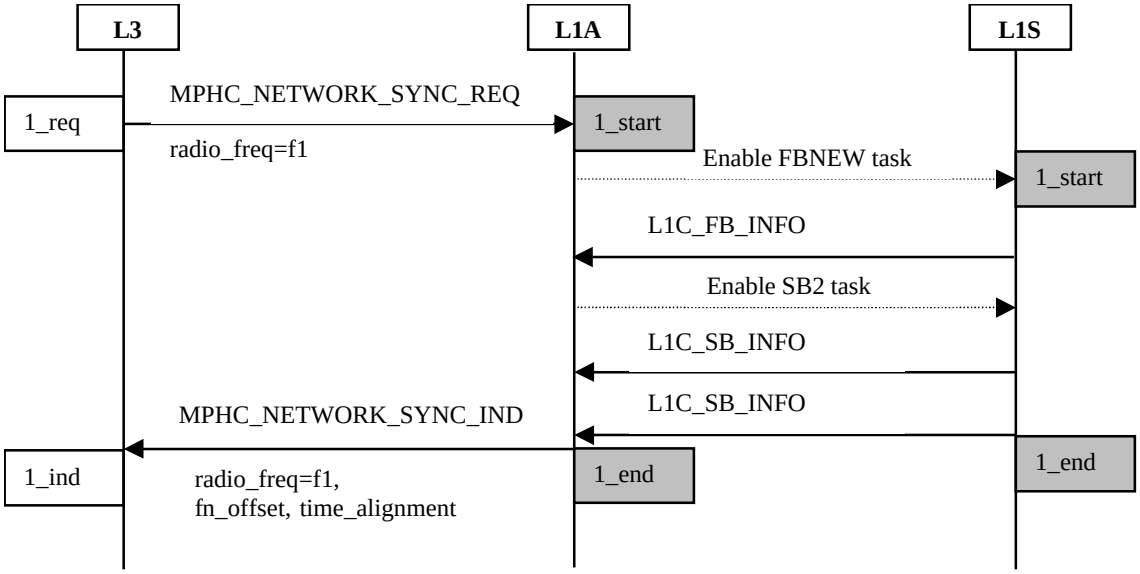


Figure 1: Network synchronization, normal flow

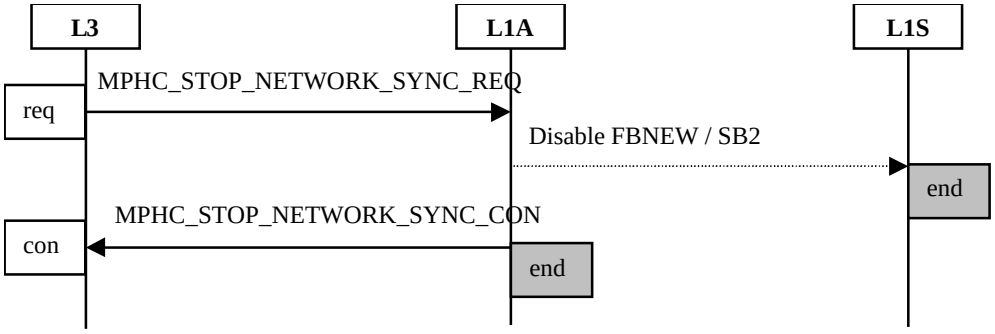


Figure 2: Network synchronization, STOP macro

The above figure represents a generic "STOP macro" which can be inserted at any place in the Figure 1. Running network synchronization processes is stopped immediately.

2.1. MPHC_NETWORK_SYNC_REQ

Notes:
- Only one instance of this message can be sent at a time to L1.

T_MPHC_NETWORK_SYNC_REQ message type

- **radio_freq**
- **fn_offset**
- **time_alignment**
- **timing_validity**
- **search_mode**

radio_freq (UWORD16)
Radio frequency channel number of the required carrier.

fn_offset (UWORD32)
Difference in absolute frame numbers between the serving cell and the non-serving cell. Validity of fn_offset dependent upon timing_validity indication.
fn_offset is in the range 0 - 2715647 (0 to (max GSM frames - 1))

time_alignment (UWORD32)
Difference in quarter bits between the first bit in a frame of the serving cell and the first bit in the next frame of the non-serving cell. Validity of time_alignment dependent upon timing_validity indication.
time_alignment is in the range 0 - 4999.

timing_validity (UWORD8)
Identifies which timing information fields are used by Layer 1.
0 - No valid timing information is supplied (i.e. fn_offset and time_alignment shall be considered invalid). Layer 1 shall carry out a full FB and SB detect with no previous synchronisation information provided.
1 – Approximate timing information is supplied. Layer 1 shall carry out a full FB and SB detect but may delay the opening of the receive window based upon the fn_offset and time_alignment information supplied.

search_mode (UWORD8)
Specifies the 'mode' being used by Layer 3. Determines the Layer 1 frequency control algorithms to use.
0 – Wideband / Cell Selection. A wideband capture and open-loop afc algorithm shall be used.
1 – Narrowband / Cell Selection. A narrowband capture and open-loop afc algorithm shall be used.

2.2. MPHC_NETWORK_SYNC_IND

T_MPHC_NETWORK_SYNC_IND message type

- **radio_freq**
- **sb_flag**
- **fn_offset**
- **time_alignment**
- **bsic**

radio_freq (UWORD16)
Radio frequency channel number of the required carrier.

sb_flag (BOOL)
Flag indicating whether synchronisation channel was read correctly or not. Reason for failure is not required by layer 3 (i.e. failure to read FB or failure to read SB).
If sb_flag is TRUE the fn_offset, time_alignment and bsic are valid.
If sb_flag is FALSE the fn_offset, time_alignment and bsic shall be considered invalid.
In both cases Layer 1 remains synchronised with the serving cell.

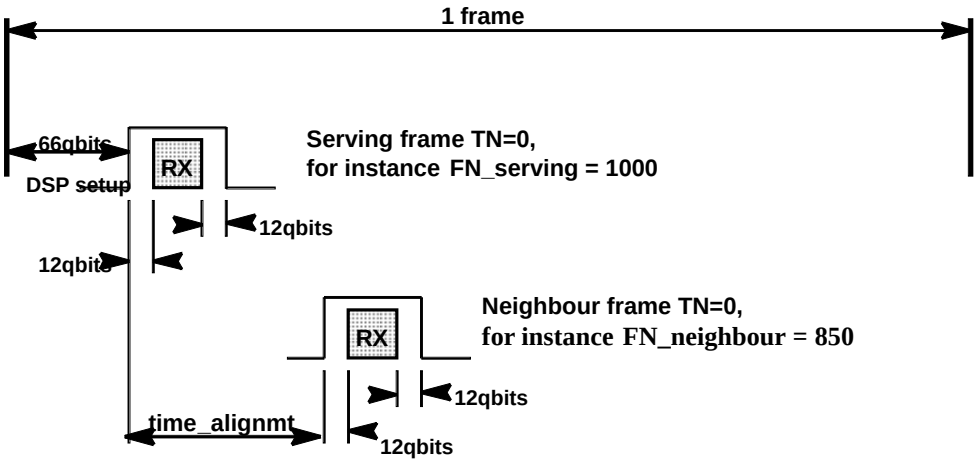
fn_offset (UWORD32)
Difference in absolute frame numbers between the serving cell and the non-serving cell. Validity of fn_offset dependent upon error_flag indication.
fn_offset is in the range 0 - 2715647 (0 to (max GSM frames – 1))

time_alignment (UWORD32)
Difference in quarter bits between the first bit in a frame of the serving cell and the first bit in the next frame of the non-serving cell. Validity of time_alignment dependent upon error_flag indication.
time_alignment is in the range 0 - 4999.

bsic (UWORD8)
Base station identity code as decoded from the synchronisation channel. This actually consists of only six bits comprising the base station colour code (3 bits) and the network colour code (3 bits). The spare bits are set to zero

Bit	7	6	5	4	3	2	1	0
	0	0	NCC(2)	NCC(1)	NCC(0)	BCC(2)	BCC(1)	BCC(0)

Note: Layer 1 has successfully read the FB, followed by the SB information. It returns the bsic, arfcn and timing information to layer 3. The timing information generated by layer 1 includes the ***time_alignment***, and the ***fn_offset***, defined here below.



fn_offset is the offset in frames between the serving cell TS0 and the neighbor cell TS0 (see figure above and example below). It is always positive within the range 0 (cells are aligned) to MAX_FN-1.
In the above example we have,

fn_offset = (FN_neighbour – FN_serving) % MAX_FN
= (850 – 1000) % 2715648 = 2715498

time_alignment is the fine quarter bit alignment within the frame between the serving cell TS0 and the neighbor cell TS0 (see figure above). It is always within the range 0 (cells are exactly aligned) to 4999.

3. Off Mode

This mode is activated by the L3 with the **MPHC_NETWORK_LOST_IND** message.

This mode is required when the upper layers do not want to perform GSM or GPRS activities. When the L1 is in this mode its only activities are:

- Auxiliary conversions (MMI_ADC_REQ message)
- Management of the Power consumption (Deep Sleep, Big Sleep, Small Sleep)

The L1 leaves this mode with the **MPHC_NETWORK_SYNC_REQ** message sent by L3.

•

4. Network synchronization lost

Notes:

- Single shot process (not continuous)
- This message is used to inform the L1 when the MS has lost the network synchronization.
- On reception of this message the L1 enters in a specific mode : “No GSM mode”. In this mode the low power consumption is enabled (deep sleep) and the 2 process authorized are the Power measurement process (MPHC_RXLEV_PROCESS) and the Auxiliary conversions (MMI_ADC_REQ) . The L1 leaves this mode on reception of the message MPHC_NETWORK_SYNQ_REQ.

This message is required to avoid a waste of consumption when the MS looses the network synchronization.

Direction	Message name	Type
L3->L1	**MPHC_NETWORK_LOST_IND**	**Trigger**

5. Read Common Control Channel Info (CCCH)

Notes:

- This process is continuous.

Direction	Message name	Type
L3->L1	**MPHC_START_CCCH_REQ**	**T_MPHC_START_CCCH_REQ**
L3->L1	**MPHC_STOP_CCCH_REQ**	**Trigger**
L3<-L1	**MPHC_STOP_CCCH_CON**	**Trigger**
L3<-L1	**MPHC_DATA_IND**	**T_MPHC_DATA_IND**

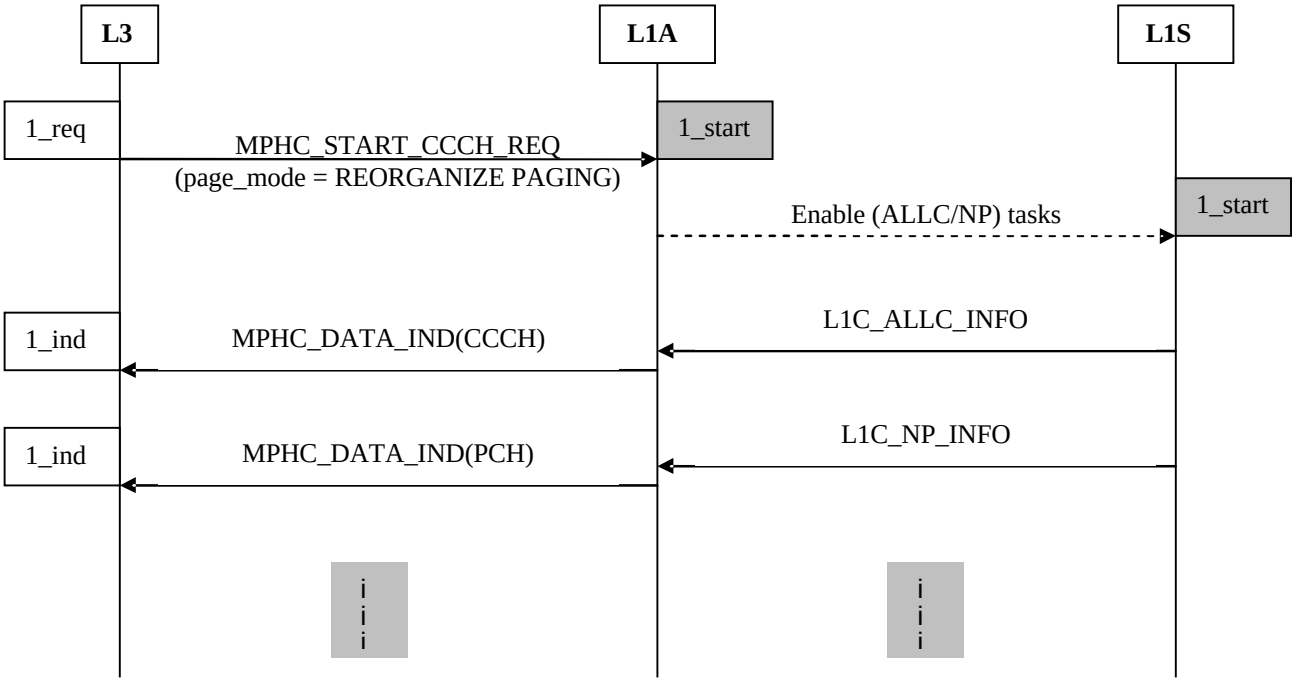


Figure 3: Read common control channel info, paging reorganization mode

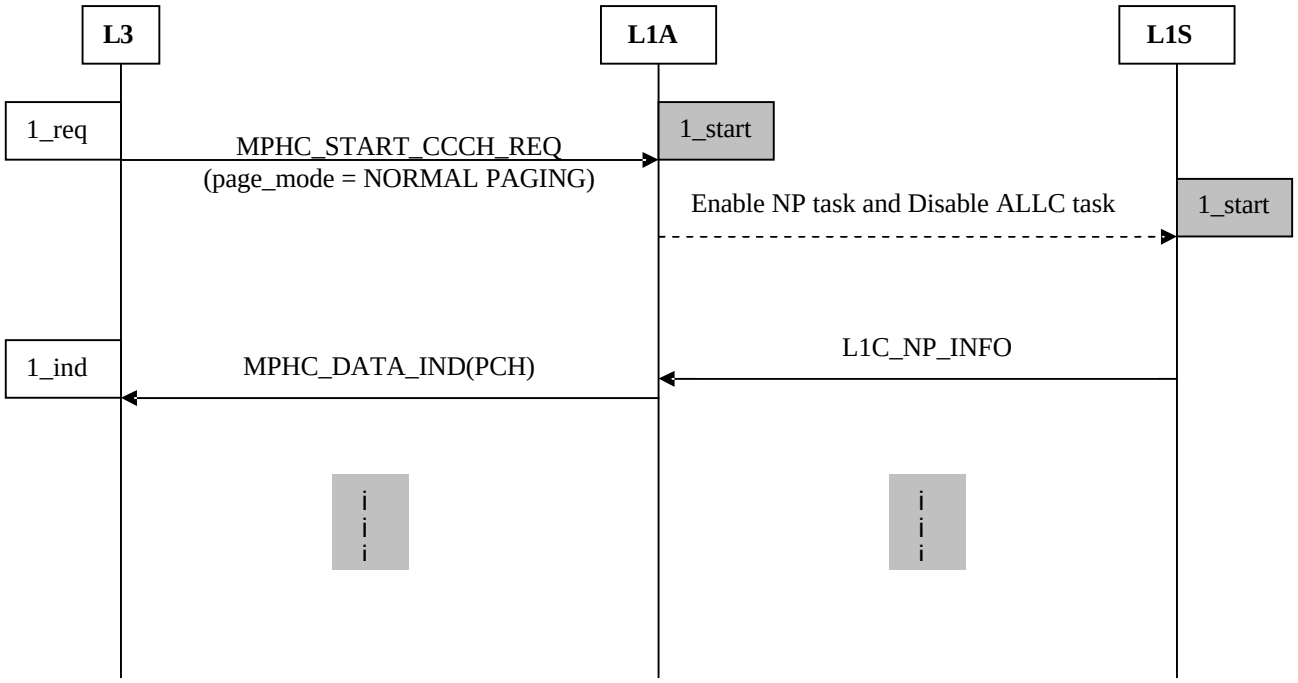


Figure 4: Read common control channel info, normal paging mode

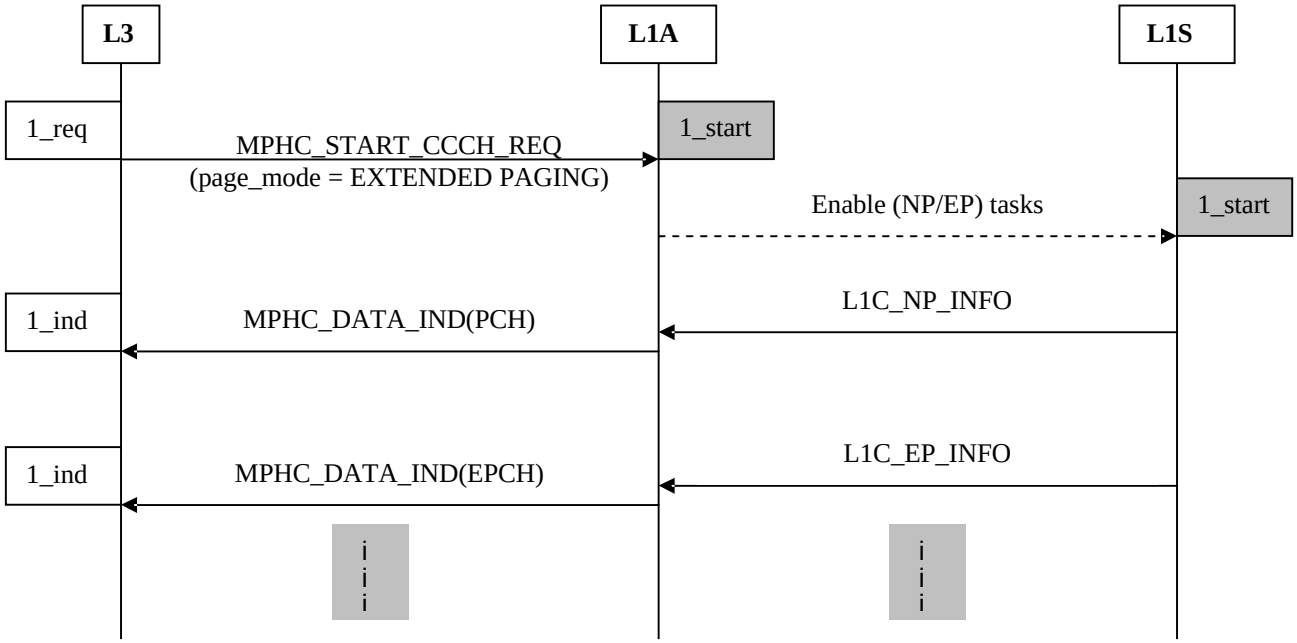


Figure 5: Read common control channel info, extended paging mode

5.1. MPHC_START_CCCH_REQ

T_MPHC_START_CCCH_REQ message type

- **bs_pa_mfrms**
- **bs_ag_blks_res**
- **bcch_combined**
- **ccch_group**
- **page_group**
- **page_block_index**
- **page_mode**

bs_pa_mfrms (UWORD8)
Broadcast value from the broadcast control channel which determines the multiframe period of PAGING REQUEST messages to the same paging subgroup. (As defined in GSM 04.08 (ETS 300 940) section 10.5.2.11):

Bits	3	2	1	0	
	0	0	1	0	2 'multiframe 51's between page blocks
	0	0	1	1	3 'multiframe 51's between page blocks
	0	1	0	0	4 'multiframe 51's between page blocks
	0	1	0	1	5 'multiframe 51's between page blocks
	0	1	1	0	6 'multiframe 51's between page blocks
	0	1	1	1	7 'multiframe 51's between page blocks
	1	0	0	0	8 'multiframe 51's between page blocks
	1	0	0	1	9 'multiframe 51's between page blocks

Thus range for bs_pa_mfrms is 2 to 9 (all bits 4-7 shall be zero)

bs_ag_blks_res (UWORD8)
Indicates how many available CCCH blocks in a MF51 are reserved solely for use as access grant blocks, and as such, do not need to be monitored for PAGING REQUEST messages. This is determined by layer 3 from decoded BCCH messages described in GSM 04.08 (ETS 300 940) section 10.5.2.11.
Range from 0 to 2 when boolean bcch_combined is TRUE (ie combined CCCH) or from 0 to 7 when boolean bcch_combined is FALSE (ie non-combined CCCH).

bcch_combined (BOOL)
Indicates whether the CCCH is combined with SDCCHs.
The CCCH is combined with SDCCHs if the boolean is TRUE, non-combined if FALSE.

ccch_group (UWORD8)
The identity of the timeslot in which the mobile should attempt to monitor a CCCH. Description can be found in GSM 05.02 (ETS 300 908) section 6.5.2:

The CCCH can be broadcast on timeslots 0,2,4 and 6.
ccch_group has the range 0 to 3 such that, when
ccch_group = 0, timeslot = 0;
ccch_group = 1, timeslot = 2;
ccch_group = 2, timeslot = 4;
ccch_group = 3, timeslot = 6.

page_group (UWORD8) Indicates which blocks within the MF51 x bs_pa_mfrms paging period the mobile should attempt to monitor. page_group has the range 0 - 80 but the upper figure is dependent upon the CCCH configuration (ie bs_pa_mfrms x number of PCH blocks per MF51).
page_block_index (UWORD8) Index, based on the number of available page blocks within a MF51, indicating the position where a page block of the mobiles own page group would occur. page_block_index has can be in the range 0 - 8 for a non-combined CCCH or 0 - 2 for a combined channel.
page_mode (UWORD8) Paging mode to be used by Layer1 0 NORMAL PAGING 1 EXTENDED PAGING 2 REORGANISE PAGING

5.2. MPHC_STOP_CCCH_REQ

When the trigger MPHC_STOP_CCCH_REQ is explicitly sent to L1, the trigger MPHC_STOP_CCCH_CON is sent back.
This confirmation is also used in case of implicit stop occurring when a Starting Time (STI) has passed (only L1 knows that STI is passed). The following cases are covered:

- Entering in GPRS single block 2 phase access
- TBF assignment after access phase or idle mode
- Dedicated channel (SDCCH or TCH) establishment
- TBF release

6. Read Serving Cell BCCH

Notes:

- This process is continuous.
- Update on fly is supported.
- SI9 scheduling (where parameter) is translated by L3 in a modulus and relative_position.
- Common stop for NBCCH and EBCCH.
- Full BCCH reading is achieved requesting (modulus=1, relative_position=0) couple for both NBCCH and EBCCH.
- Default scheduling (when SI9 not yet received) is the translation of the 05.02/TC scheduling into (modulus, relative_position) couples.

Direction	Message name	Type
L3->L1	**MPHC_SCELL_NBCCH_REQ**	**T_MPHC_SCELL_NBCCH_REQ**
L3->L1	**MPHC_SCELL_EBCCH_REQ**	**T_MPHC_SCELL_EBCCH_REQ**
L3->L1	**MPHC_STOP_SCELL_BCCH_REQ**	**Trigger**
L3<-L1	**MPHC_STOP_SCELL_BCCH_CON**	**Trigger**
L3<-L1	**MPHC_DATA_IND**	**T_MPHC_DATA_IND**

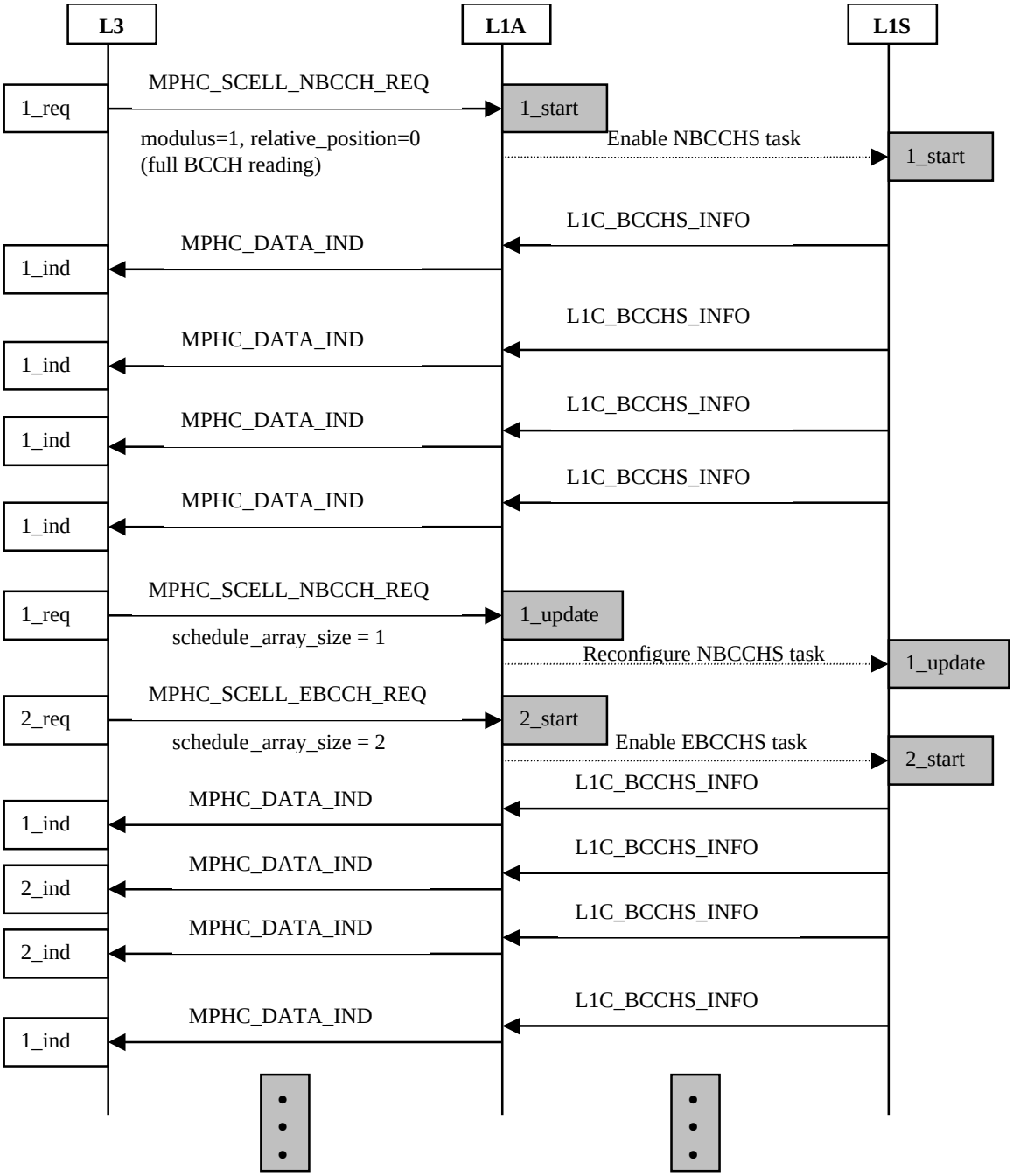


Figure 6: Serving cell BCCH reading, normal flow

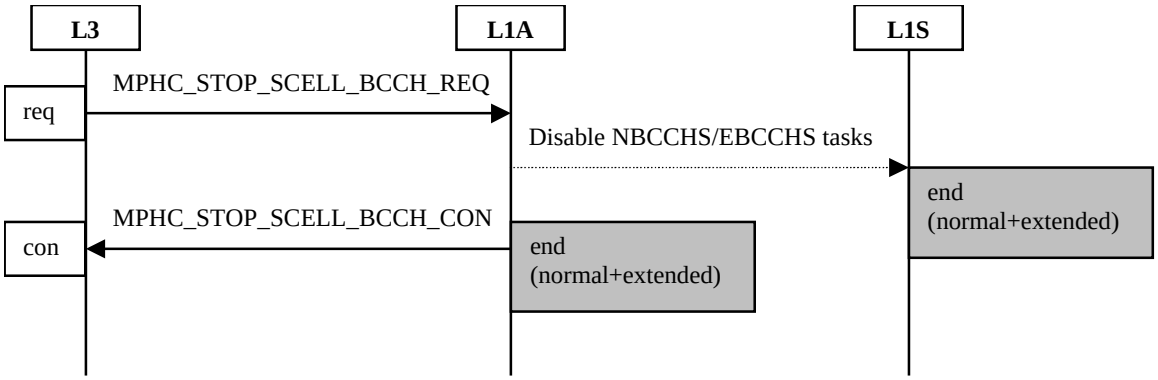


Figure 7: Serving cell BCCH reading, "STOP ALL" macro

The figure above represents a generic "STOP macro" which can be inserted at any place in the Figure 6. Running normal or extended serving BCCH reading processes are stopped immediately.

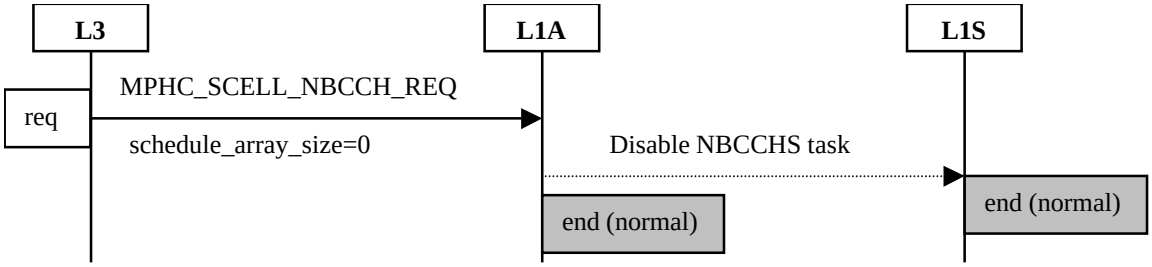


Figure 8: Serving cell BCCH reading, "STOP NORMAL BCCH" macro

The figure above represents a "STOP macro" for NBCCHS which can be inserted at any place in the Figure 6. The running normal serving BCCH reading process is stopped immediately.

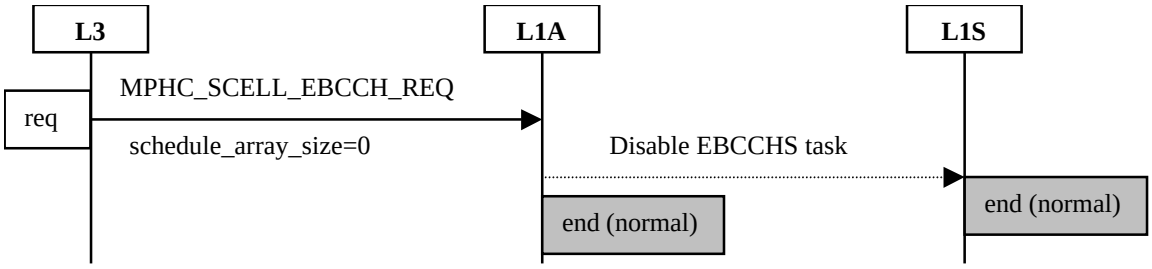


Figure 9: Serving cell BCCH reading, "STOP EXTENDED BCCH" macro

The figure above is a "STOP macro" for EBCCHS which can be inserted at any place in the figure above. The running extended serving BCCH reading process is stopped immediately.

6.1. MPHC_SCELL_NBCCH_REQ

Notes:

- This message is sent to start or reconfigure the reading of the serving cell Normal BCCH.

T_MPHC_SCELL_NBCCH_REQ message type

- **schedule_array_size**
- **schedule_array[ ]**

schedule_array_size (UWORD8)
Number of element in the schedule_array table.
This field is Range 0 to 10.
Value 0 means stop any Normal BCCH reading.

schedule_array[ ] (structure)

modulus (UWORD16)
Modulus part (M) of the positional information within the GSM multiframe structure (GSM 04.08 section 10.5.2.37a).
Range 1 – 65535

relative_position (UWORD16)
Relative position part (P) of the positional information within the GSM multiframe structure (GSM 04.08 section 10.5.2.37a).
Range 0 – 65535

6.2. MPHC_SCELL_EBCCH_REQ

Notes:

- This message is sent to start or reconfigure the reading of the serving cell Extended BCCH.

T_MPHC_SCELL_EBCCH_REQ message type

- **schedule_array_size**
- **schedule_array[ ]**

schedule_array_size (UWORD8)
Number of element in the schedule_array table.
This field is Range 0 to 10.
Value 0 means stop any Extended BCCH reading.

schedule_array[ ] (structure)

modulus (UWORD16)
Modulus part (M) of the positional information within the GSM multiframe structure (GSM 04.08 (ETS 300 940) section 10.5.2.37a).
Range 1 – 65536

relative_position (UWORD16)
Relative position part (P) of the positional information within the GSM multiframe structure (GSM 04.08 (ETS 300 940) section 10.5.2.37a).
Range 0 – 65535

6.3. MPHC_DATA_IND

Notes:

- This message is used to report any serving cell block reception to L3 in any mode except Dedicated mode with a TCH channel.

T_MPHC_DATA_IND message type

- **radio_freq**
- **l2_channel**
- **error_flag**
- **l2_frame[23]**
- **tc**
- **ccch_lev**
- **fn**

- **neigh_id (RESERVED)**

radio_freq (UWORD16)
Radio frequency channel number from which the read was attempted.

l2_channel (UWORD8)
Indicates the MF51 block type read (enum type), maps to l1s msg signal code:

1	(L2_CHANNEL_SACCH)	indicates SACCH block
2	(L2_CHANNEL_SDCCH)	indicates SDCCH block
3	(L2_CHANNEL_FACCH_H)	indicates half rate FACCH
4	(L2_CHANNEL_FACCH_F)	indicates full rate FACCH
5	(L2_CHANNEL_CCCH)	indicates CCCH block
6	(L2_CHANNEL_NBCCH)	indicates normal BCCH
7	(L2_CHANNEL_PCH)	indicates PCH block
8	(L2_CHANNEL_EPCH)	indicates extended PCH block
9	(L2_CHANNEL_CBCH)	indicates CBCH block
10	(L2_CHANNEL_EBCCH)	indicates extended BCCH block

error_flag (BOOL)
Indication as to whether the frame was decoded correctly. If error_flag is TRUE, the message is invalid. If error_flag is FALSE the message is valid.

l2_frame[23] (23 x UWORD8)
Array of data

tc (UWORD8)
Value of TC as defined in GSM 05.02 section 6.3.4. tc = (FN div 51) % 8.
Range of tc is 0 - 7.

ccch_lev (WORD8)
ccch_lev encodes the power strength level measured on CCCH. It's an running average value (without forgetting factor). The average is made on 4 bursts. **ccch_lev** is coded as rxlev defined in [6].
Note: **ccch_lev** is encoded without clipping (binary coded ranges -128 to +127 in spite of "0 to 63" range as it is specified in [6]).

fn (UWORD32)
Last frame number of the attempted block read.
For debug purpose only. No use of this frame number must be made to schedule the activity within the system.

RESERVED FOR L1A à L1S
neigh_id (UWORD8)

6.4. MPHC_STOP_BCCH_REQ

When the trigger MPHC_STOP_BCCH_REQ is explicitly sent to L1, the trigger MPHC_STOP_BCCH_CON is sent back.
This confirmation is also used in case of implicit stop occurring when a Starting Time (STI) has passed (only L1 knows that STI is passed). The following cases are covered:

- Entering in GPRS single block 2 phase access
- TBF assignment after access phase or idle mode
- Dedicated channel (SDCCH or TCH) establishment
- TBF release

7. Read Non-Serving Cell SCH in Idle mode

6 strongest neighbor cell monitoring in Idle mode has been extended to 12 cells and new primitive MPHC_NCELL_LIST_SYNC_REQ has been added. Those 2 modifications are enabled by **compilation switch L1_12NEIGH** (default value is 0 except for D-sample).

Notes:

- Single shot process (not continuous).
- L3 can send up to 12 neighbor cell monitoring requests, L1 stores them and process them independently.
- If L1 receives a new MPHC_NCELL_SYNC_REQ or MPHC_NCELL_LIST_SYNC_REQ and there is already 12 pending request within L1, the new request is ignored. Otherwise new cells are added to the neighbor cell list in progress
- If several requests are to be served in the same TDMA frame, L1 gives priority to the oldest stored request.
- L3 may avoid sending a stop message to terminate the process if there is no more pending request in L1.
- "Neighbor cell read control" from L3 is postponed to a second implementation phase. This control would consist in a trigger (MPHC_NCELL_PROCEED_REQ) from L3 to enable any neighbor cell activity in L1 after the normal paging reading for instance.
- L3 may use MPHC_NCELL_SYNC_REQ or / and MPHC_NCELL_LIST_SYNC_REQ (with **eotd**=FALSE) to launch either 1 neighbor cell monitoring or a list of neighbor cell monitoring respectively.

Here below are listed the rules applying for neighbor cells EOTD monitoring:

- Prior to start EOTD process, L3 must first stop all current neighbor cell monitoring process (BA list monitoring) with a MPHC_STOP_NCELL_SYNC_REQ message for the full list.
- EOTD process is started by MPHC_NCELL_LIST_SYNC_REQ with **eotd** field at TRUE and **timing_validity**=2 (SB confirmation) for all neighbor cells in the list.
- When EOTD is ongoing all sub sequent neighbor cell monitoring requests are ignored except a full MPHC_STOP_NCELL_SYNC_REQ.
- Stopping EOTD process is only possible by Full list MPHC_STOP_NCELL_SYNC_REQ.
- L1 automatically precedes the list of neighbor cells by a single Serving cell SB monitoring. Then it executes the list of neighbor cells monitoring requested. Then it concludes the EOTD monitoring by a Serving cell SB monitoring again.
- MPHC_NCELL_SYNC_IND results are sent independently to L3. One EOTD monitoring process will generate an MPHC_NCELL_SYNC_IND for the first Serving cell SB, then up to 12 results messages for the neighbor cells and a result message for the last Serving Cell SB. Then process is over.

Direction	Message name	Type
L3->L1	**MPHC_NCELL_SYNC_REQ**	**T_MPHC_NCELL_SYNC_REQ**
L3<-L1	**MPHC_NCELL_SYNC_IND**	**T_MPHC_NCELL_SYNC_IND**
L3->L1	**MPHC_STOP_NCELL_SYNC_REQ**	**T_MPHC_STOP_NCELL_SYNC_REQ**
L3<-L1	**MPHC_STOP_NCELL_SYNC_CON**	**Trigger**
L3->L1	**MPHC_NCELL_LIST_SYNC_REQ**	**T_ MPHC_NCELL_LIST_SYNC_REQ**

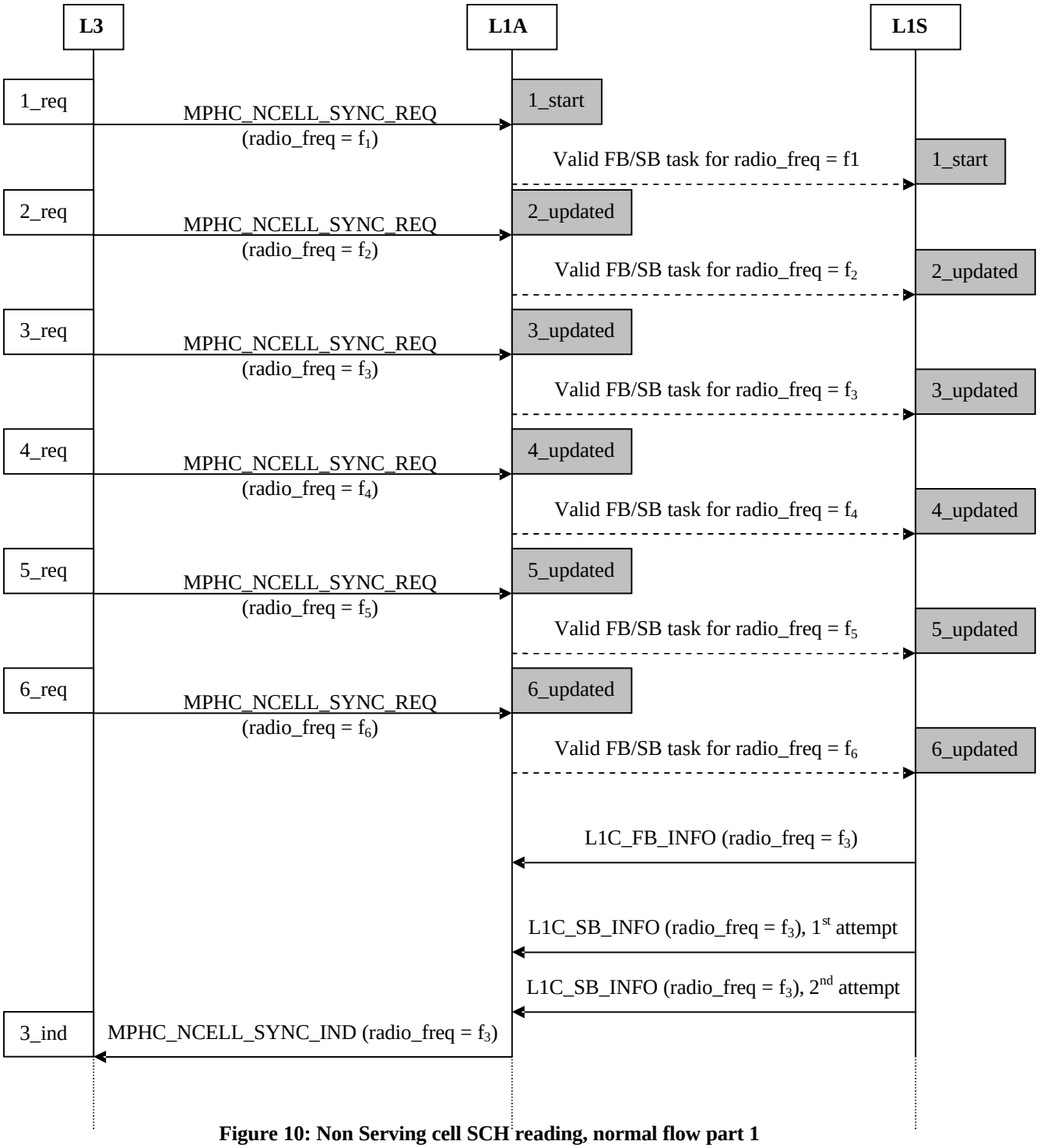


Figure 10: Non Serving cell SCH reading, normal flow part 1

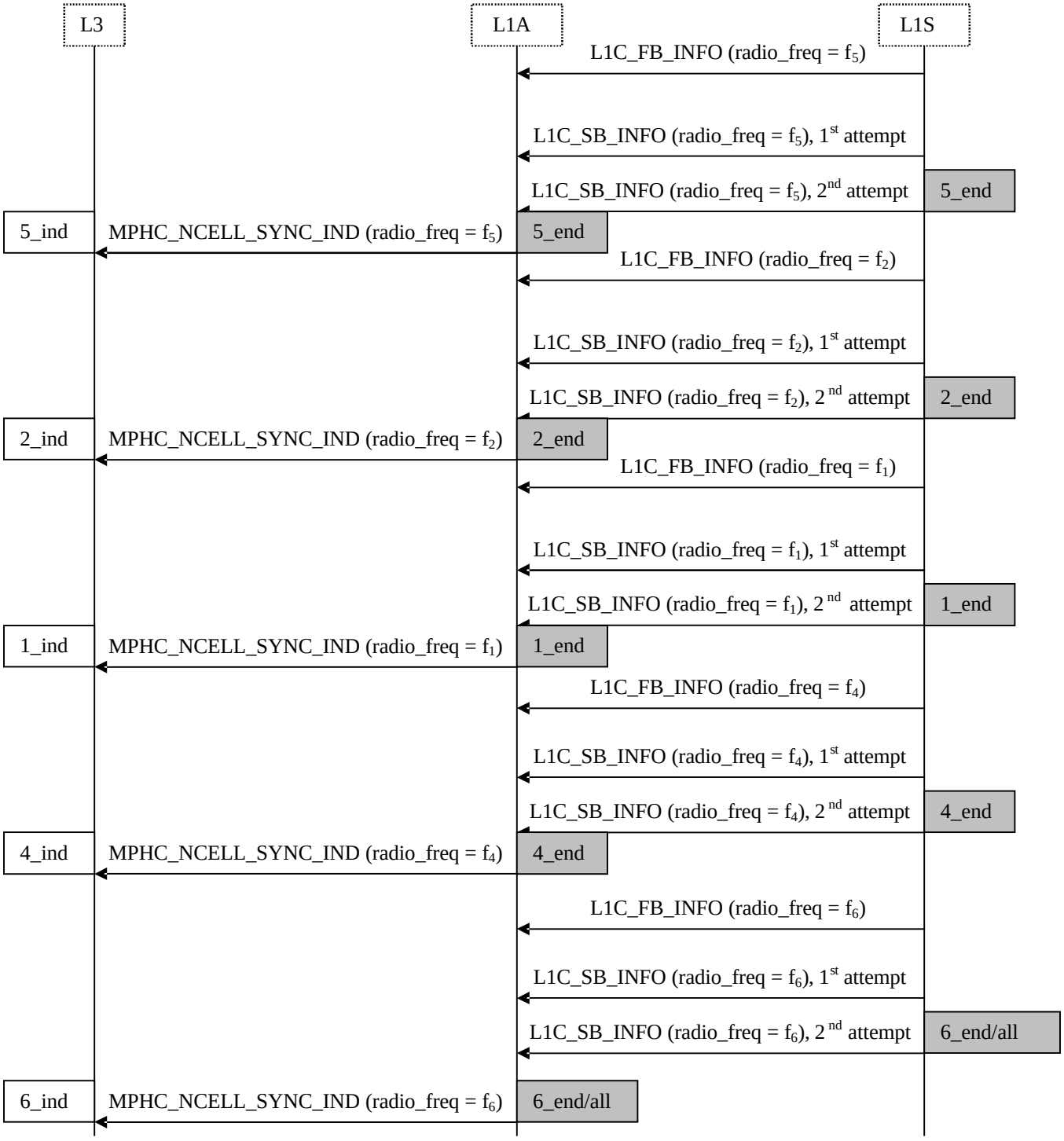


Figure 11: Non Serving cell SCH reading, normal flow part 2

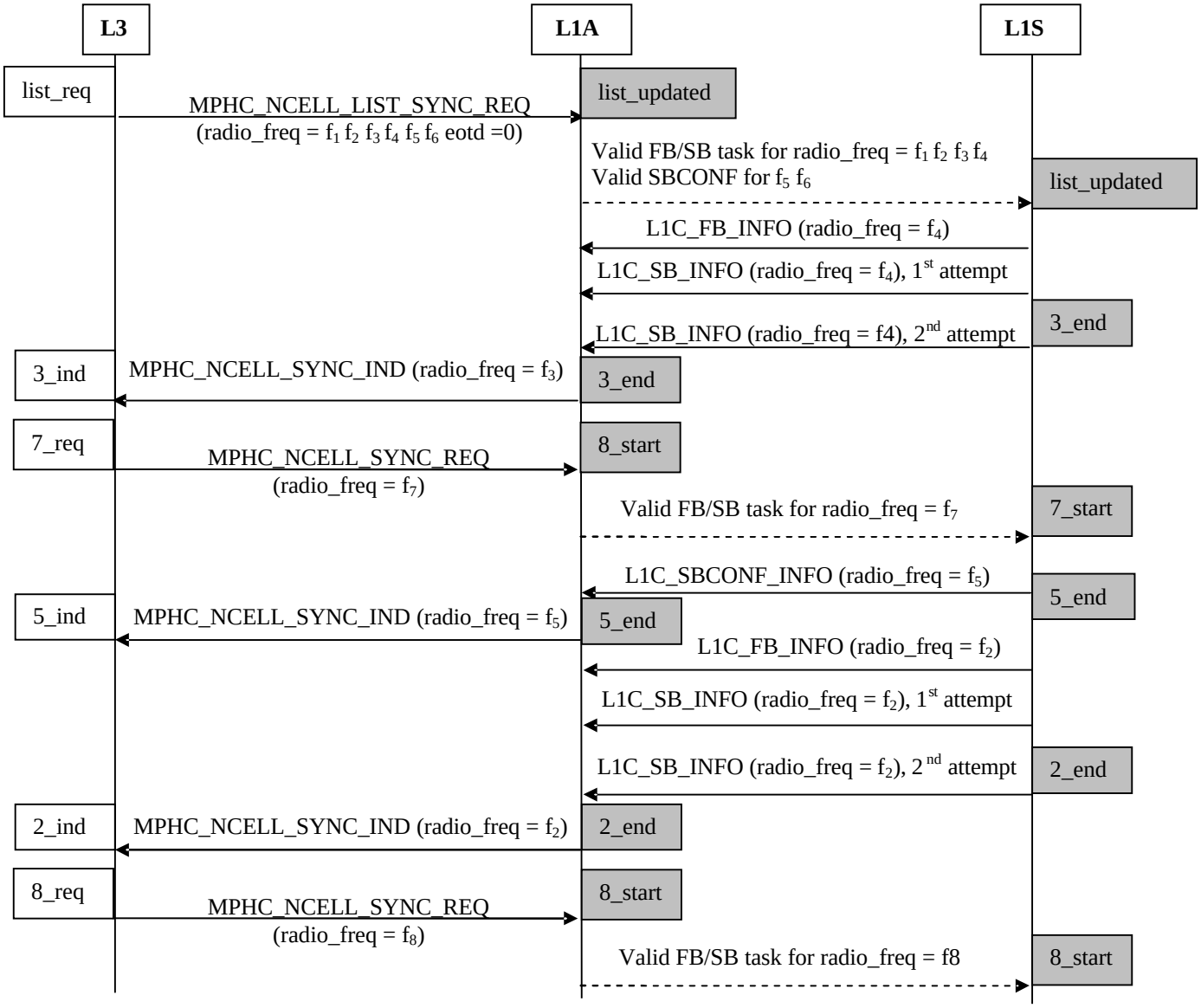


Figure 12: Example of Non Serving cell SCH reading by mix of single requests and list request (part 1)

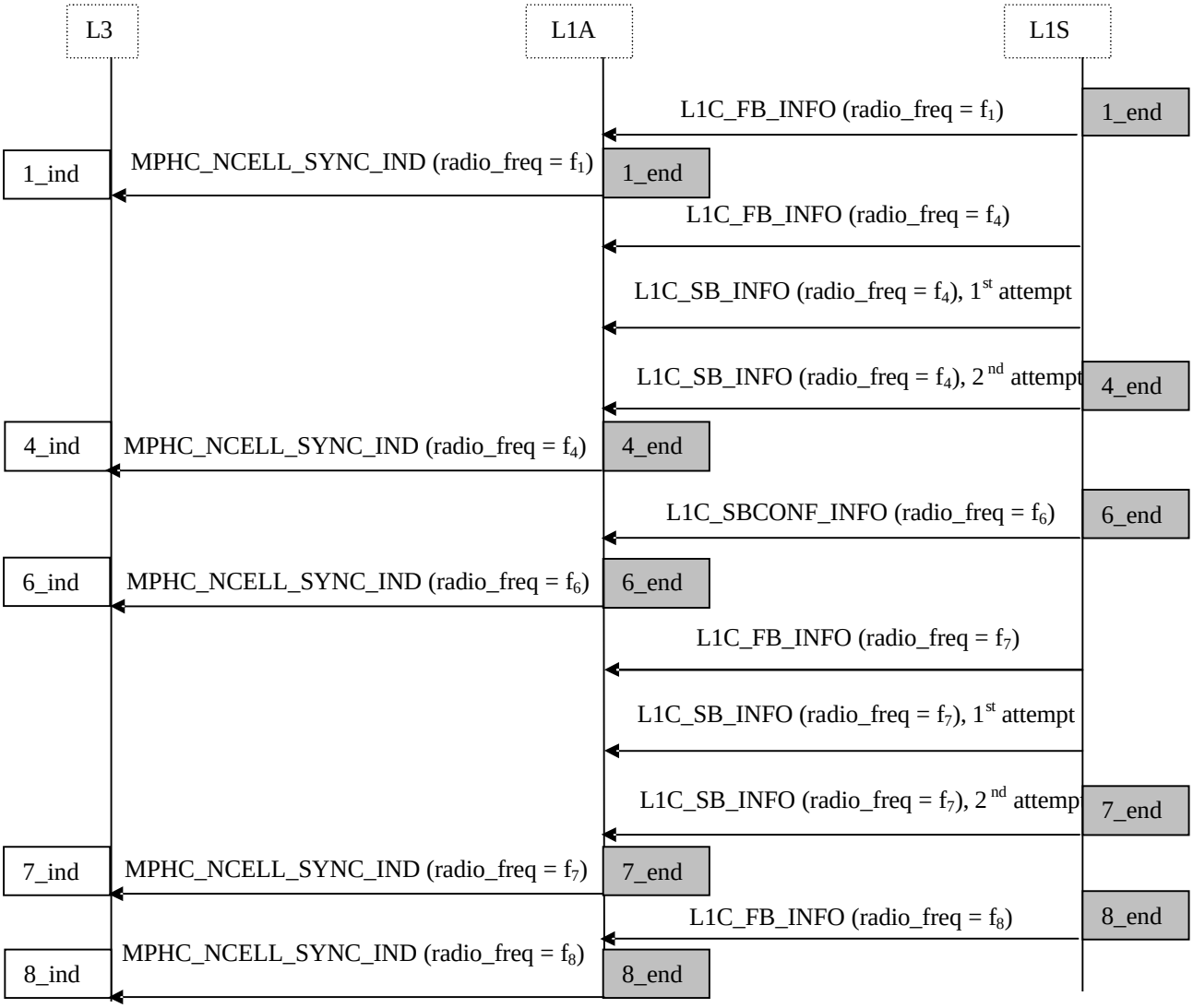


Figure 13: Example of Non Serving cell SCH reading by mix of single requests and list request (part 2)

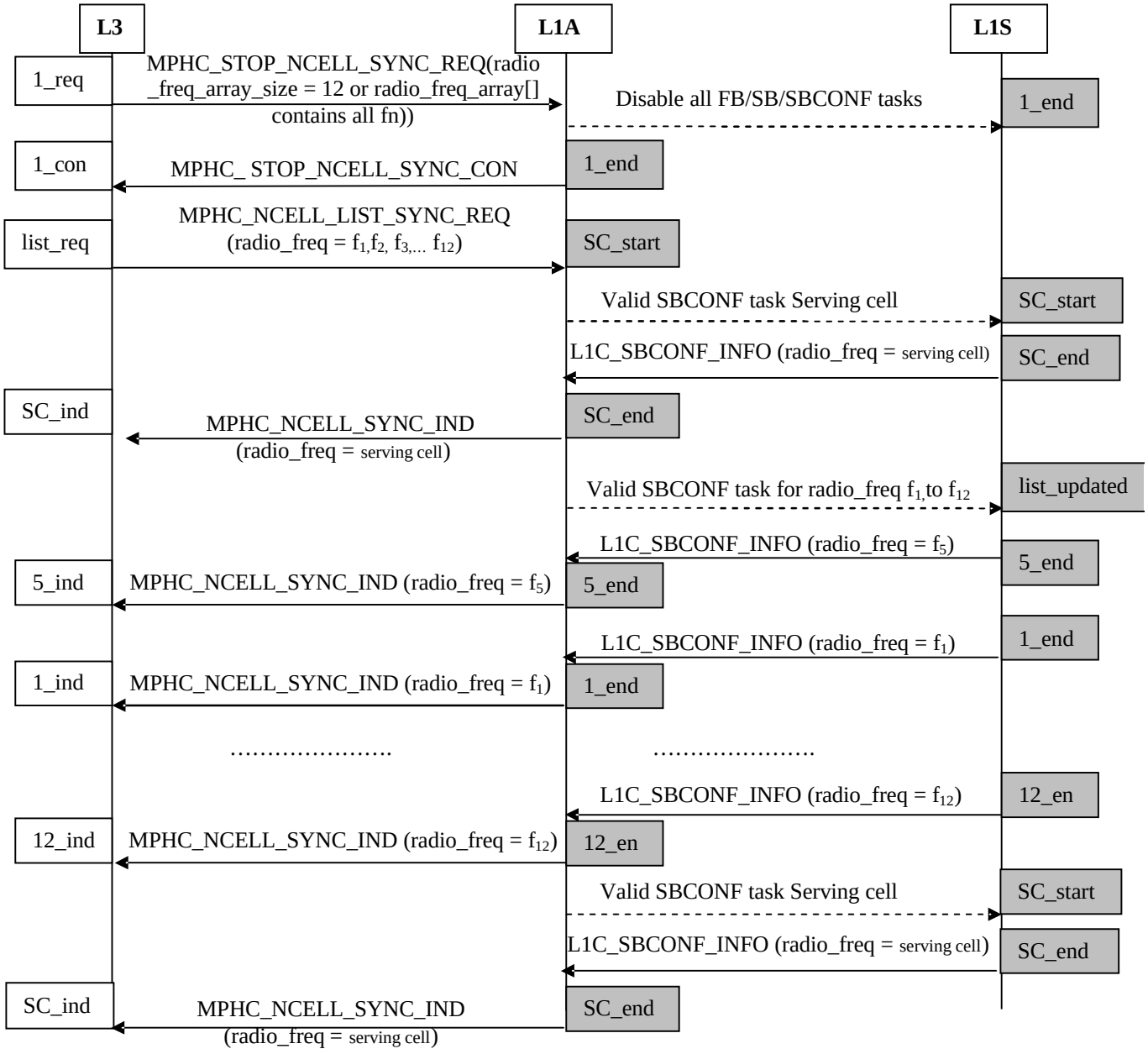


Figure 14: Example of EOTD monitoring process for 12 cells

Figure 14 represents an example of L3/L1 communication for EOTD monitoring when there are 12 neighbor cells in list (it could be less)

1_Req: L3 stops any monitoring in progress (Full STOP)
1_con: When receiving the stop confirmation, L3 starts an EOTD monitoring session by MPHC_NCELL_LIST_SYNC_REQ primitive with **eotd** field at TRUE and **timing_validity** = 2 for all requested cells (up to 12). L1 begins by monitoring the Serving Cell , then waits for the result.
SC_Ind: L1 forwards Serving Cell result to L3 and enables the full neighbor cells monitoring.
5/12_Ind: L1 monitors at best all neighbour cells. Each result is sent to L3.
SC_Ind: When completed, L1 monitors Serving cell again.and forwards result to L3.
EOTD process is over. L1 list is empty.

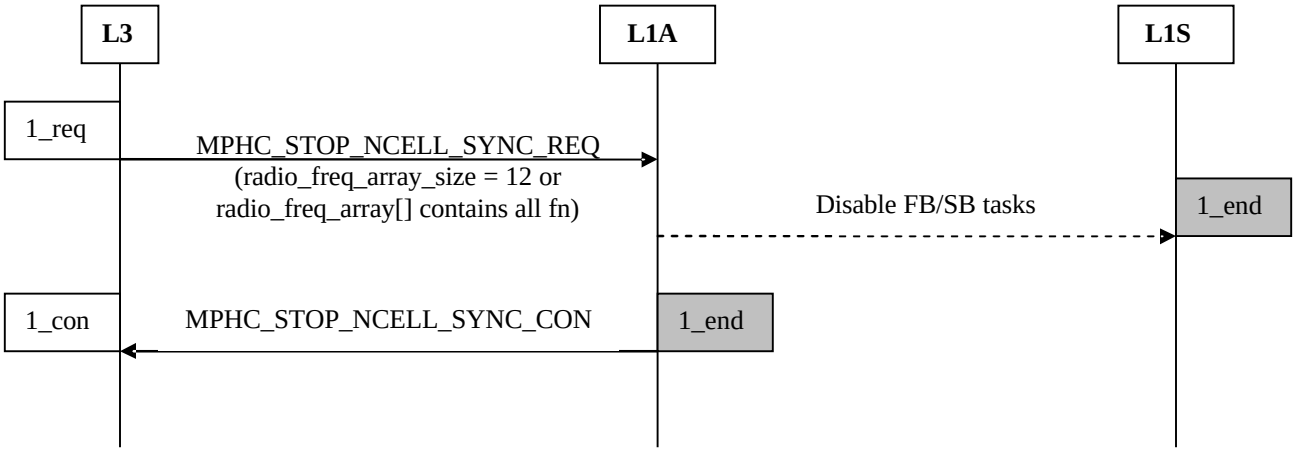


Figure 15: Non-Serving Cell SCH, full STOP macro

This figure represents a generic "STOP macro" which can be inserted at any place in the Figure 10 and Figure 11. The running non-serving cell SCH process is stopped immediately either if radio_freq_array_size equals 12 (maximum number of requests to be served in parallel) or if radio_freq_array[] contains all carriers for which non-serving cell SCH reading was request previously.

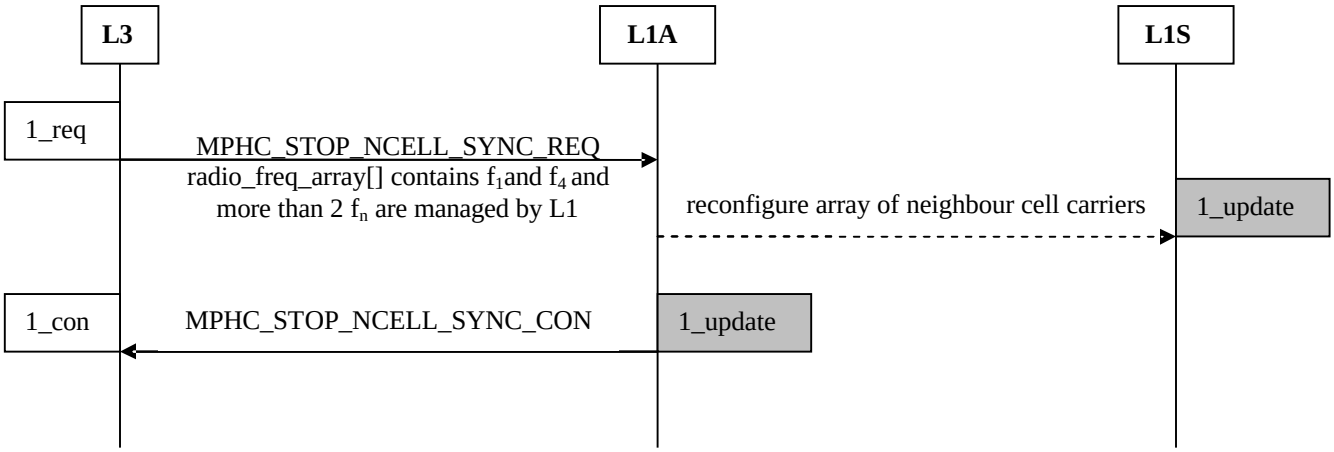


Figure 16: Non-Serving Cell SCH, partial STOP macro (array reconfiguration)

This figure represents a generic "STOP macro" which can be inserted at any place in the Figure 10 and Figure 11. The running non-serving cell SCH process is reconfigured i.e. the SCH reading of neighbour carriers f_1 and f_4 is stopped while the process continuous for all other previously requested carriers f_n.

7.1. MPHC_NCELL_SYNC_REQ

Notes:

- Several instance of this message can be sent to L1. L1 can cope with up to 6 such pending requests.

T_MPHC_NCELL_SYNC_REQ message type

- **radio_freq**
- **fn_offset**
- **time_alignment**
- **timing_validity**

radio_freq (UWORD16)
Radio frequency channel number of the required carrier.

fn_offset (UWORD32)
Difference in absolute frame numbers between the serving cell and the non-serving cell. Validity of fn_offset dependent upon timing_validity indication.
fn_offset is in the range 0 - 2715647 (0 to (max GSM frames - 1))

time_alignment (UWORD32)
Difference in quarter bits between the first bit in a frame of the serving cell and the first bit in the next frame of the non-serving cell. Validity of time_alignment dependent upon timing_validity indication.
time_alignment is in the range 0 - 4999.

timing_validity (UWORD8)
Identifies which timing information fields are used by Layer 1.
0 - No valid timing information is supplied (i.e. fn_offset and time_alignment shall be considered invalid). Layer 1 shall carry out a full FB and SB detect with no previous synchronisation information provided.
1 – Approximate timing information is supplied. Layer 1 shall carry out a full FB and SB detect but may delay the opening of the receive window based upon the fn_offset and time_alignment information supplied.
2 – Accurate timing information supplied. Layer 1 shall attempt an SB capture using the fn_offset and time_alignment provided by Layer 3.

7.2. MPHC_NCELL_LIST_SYNC_REQ

Notes:

- This message is sent to request up to 12 cells monitoring.
- For NCELL monitoring eotd field must be at **FALSE**
- For EOTD monitoring eotd field must be at **TRUE** and **timing_validity** field must be 2 for all cells.

T_MPHC_NCELL_LIST_SYNC_REQ message type

- **Eotd**
- **list_size**
- **ncell_list[]**
  - **radio_freq**
  - **fn_offset**
  - **time_alignment**
  - **timing_validity**

eotd (UWORD8)
Boolean- FALSE for neighbor cell monitoring- TRUE for EOTD monitoring session.

list_size (UWORD8)
1 to 12 neighbor cells defined in ncell_list

NCELL_LIST[12]

radio_freq (UWORD16)
Radio frequency channel number of the required carrier.

fn_offset (UWORD32)
Difference in absolute frame numbers between the serving cell and the non-serving cell. Validity of fn_offset dependent upon timing_validity indication.
fn_offset is in the range 0 - 2715647 (0 to (max GSM frames - 1))

time_alignment (UWORD32)
Difference in quarter bits between the first bit in a frame of the serving cell and the first bit in the next frame of the non-serving cell. Validity of time_alignment dependent upon timing_validity indication.
time_alignment is in the range 0 - 4999.

timing_validity (UWORD8)
Identifies which timing information fields are used by Layer 1.
0 - No valid timing information is supplied (i.e. fn_offset and time_alignment shall be considered invalid). Layer 1 shall carry out a full FB and SB detect with no previous synchronisation information provided.

1 – Approximate timing information is supplied. Layer 1 shall carry out a full FB and SB detect but may delay the opening of the receive window based upon the fn_offset and time_alignment information supplied. 2 – Accurate timing information supplied. Layer 1 shall attempt an SB capture using the fn_offset and time_alignment provided by Layer 3.

7.3. MPHC_NCELL_SYNC_IND

Notes:

- This indication message is generated to answer an individual neighbor cell synchronization request.

T_MPHC_NCELL_SYNC_IND message type

- **radio_freq**
- **sb_flag**
- **fn_offset**
- **time_alignment**
- **bsic**

- **neigh_id (RESERVED)**
- **attempt (RESERVED)**
- **pm (RESERVED)**
- **toa (RESERVED)**
- **angle (RESERVED)**
- **snr (RESERVED)**

- **eotd_data_valid**
- **mode**
- **d_eotd_first**
- **d_eotd_max**
- **d_eotd_nrj**
- **a_eotd_crosscor[18]**
- **timetag**
- **fn_sb_neigh**
- **fn_in_SB**
-
- **toa_correction (RESERVED)**
- **delta_fn (RESERVED)**
- **delta_qbit (RESERVED)**

radio_freq (UWORD16)
Radio frequency channel number of the required carrier.

sb_flag (BOOL)
Flag indicating whether synchronisation channel was read correctly or not. Reason for failure is not required by layer 3 (i.e. failure to read FB or failure to read SB).
If sb_flag is TRUE the fn_offset, time_alignment and bsic are valid.

If sb_flag is FALSE the fn_offset, time_alignment and bsic shall be considered invalid.
In both cases Layer 1 remains synchronised with the serving cell.

fn_offset (UWORD32)
Difference in absolute frame numbers between the serving cell and the non-serving cell. Validity of fn_offset dependent upon timing_validity indication.
fn_offset is in the range 0 – 2715647 (0 to (max GSM frames – 1))

time_alignment (UWORD32)
Difference in quarter bits between the first bit in a frame of the serving cell and the first bit in the next frame of the non-serving cell. Validity of time_alignment dependent upon error_flag indication.
time_alignment is in the range 0 - 4999.

bsic (UWORD8)
Base station identity code as decoded from the synchronisation channel. This actually consists of only six bits comprising the base station colour code (3 bits) and the network colour code (3 bits). The spare bits are set to zero

Bit	7	6	5	4	3	2	1	0
	0	0	NCC(2)	NCC(1)	NCC(0)	BCC(2)	BCC(1)	BCC(0)

RESERVED FOR L1S à L1A

neigh_id (UWORD8)

attempt (UWORD8)

RESERVED FOR TRACE / DEBUG

pm (UWORD 32)

toa (UWORD32)

angle (UWORD32)

snr (UWORD32)

EOTD area

eotd_data_valid(UWORD8)
TRUE indicates to RR that msg contains EOTD data (otherwise FALSE)

Mode (UWORD8)
0 Idle 1 Dedic/packet transfert

d_eotd_first(WORD16) Delay in GSM bits between the arrival of the first IQ sample and the position of the first correlation (max correlation "p- 4").
d_eotd_max(WORD16) Delay in GSM bits between the arrival of the first IQ sample and the position of the max correlation.
d_eotd_nrj(UWORD32) Sum of the amplitudes of the sampled I and Q data
a_eotd_crosscor(WORD16[18] Real and imaginary parts for m=p-4 to p+4 (9 points)
timetag(UWORD32) Difference in quarter bits between first correlation of SB serving cell and first correlation of SB neighbor cell.
Fn_sb_neigh(UWORD32) frame number in which SB window is started (in serving cell domain) . Used by L1A for timetag computation (method 1).
Fn_in_SB (UWORD32) frame number computed from T1,T2,T3 provided within burst. Used by L1A for timetag computation (method 1).
RESERVED FOR L1S à L1A
Toa_correction (UWORD32)
Delta_fn(UWORD32)
Delta_qbit(WORD32)

7.4. MPHC_STOP_NCELL_SYNC_REQ

Notes:

- This stop message indicates the neighbor radio frequency L1 must remove from its neighbor synchronization pending list.
- Up to 12 neighbor cell radio frequencies can be contained. When 12 are provided the message is interpreted as a complete process stop request.

T_MPHC_STOP_NCELL_SYNC_REQ message type

- **radio_freq_array_size**
- **radio_freq_array[]**

radio_freq_array_size (UWORD8)
Number of entries in the radio_freq_array.

radio_freq_array[] (up to 12 x UWORD16) Neighbor cell radio frequency array.

8. Read Non-Serving Cell BCCH

Notes:

- Single shot process (not continuous).
- Only the first scheduled BCCH neighbor read for a request is indicated back to L3. For next SI reading on the same neighbor, a new MPHC_NCELL_BCCH_REQ message is needed.
- L3 can send up to 6 MPHC_NCELL_BCCH_REQ, L1 stores them and process them independently.
- If L1 receives a new MPHC_NCELL_BCCH_REQ and there is already 6 pending request within L1, the new request is ignored.
- If several requests are to be served in the same TDMA frame, L1 gives priority to the oldest stored request.
- L3 may avoid sending a stop message to terminate the process if there is no more pending request in L1.
- Can be used for instance to read SI 3/4/7/8 from a neighbor cell to prepare the cell reselection.
- Can be used to read any SI during cell selection to speed up the cell selection process (with a draw back in power consumption).
- This interface does not allow the update “on fly” of a BCCH reading request. To achieve that, L3 must first stop the BCCH reading task for the considered carrier (MPHC_STOP_NCELL_BCCH_REQ) and send a new request for the same carrier.
- Process allowed in Packet Transfer and aborted on uplink and downlink TBF released. In this mode, the process is a one shoot process, i.e. to the message posted by L3, L1 puts back only data blocks corresponding to BCCH blocks requested.. If the process is activated in Idle mode, it must be aborted by the L3 before a Packet Transfer assignment.

Direction	Message name	Type
L3->L1	**MPHC_NCELL_BCCH_REQ**	**T_MPHC_NCELL_BCCH_REQ**
L3<-L1	**MPHC_NCELL_BCCH_IND**	**T_MPHC_DATA_IND**
L3->L1	**MPHC_STOP_NCELL_BCCH_REQ**	**T_MPHC_STOP_NCELL_BCCH_REQ**
L3<-L1	**MPHC_STOP_NCELL_BCCH_CON**	**Trigger**

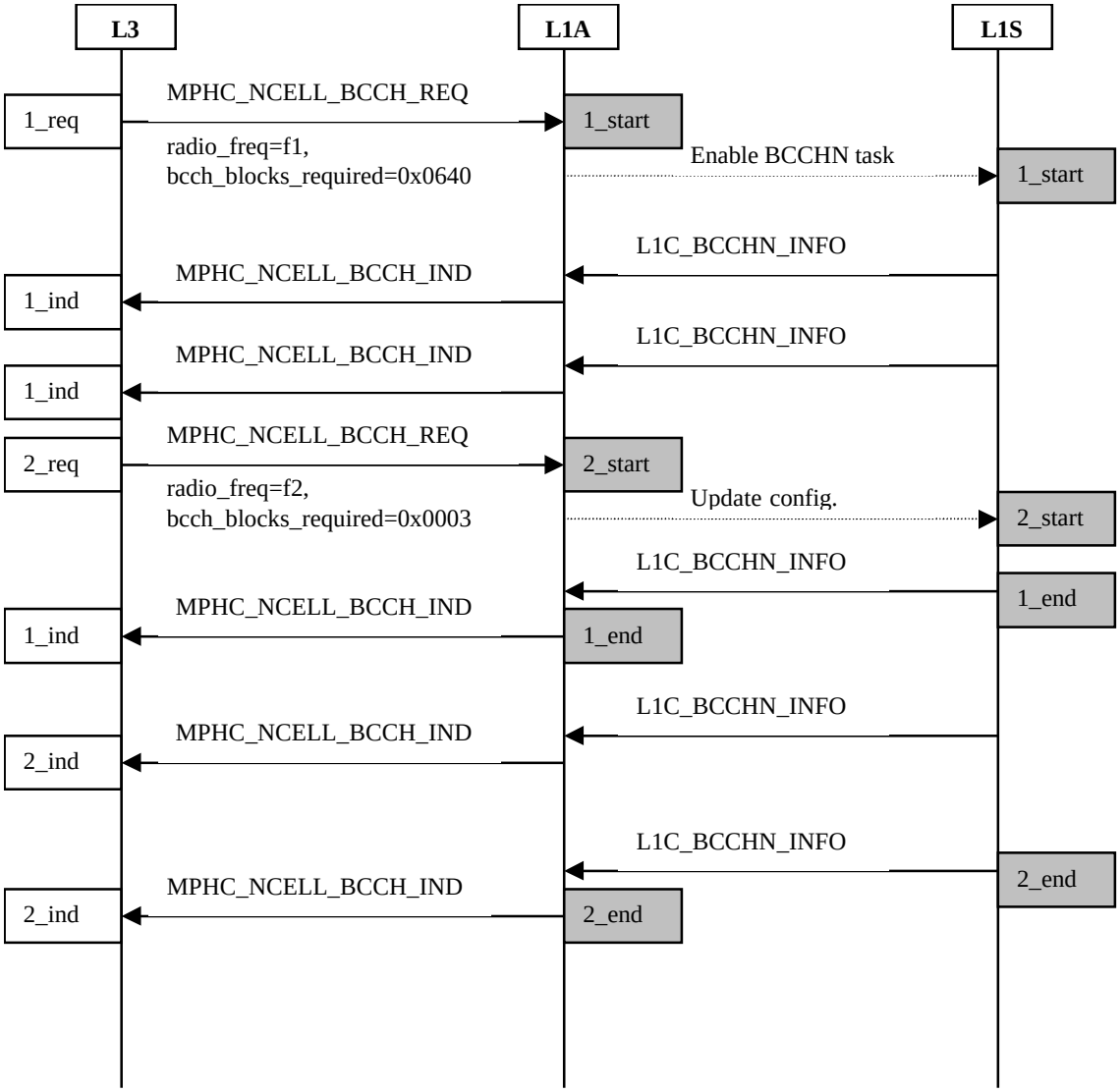


Figure 17: Neighbour cell BCCH reading, normal flow

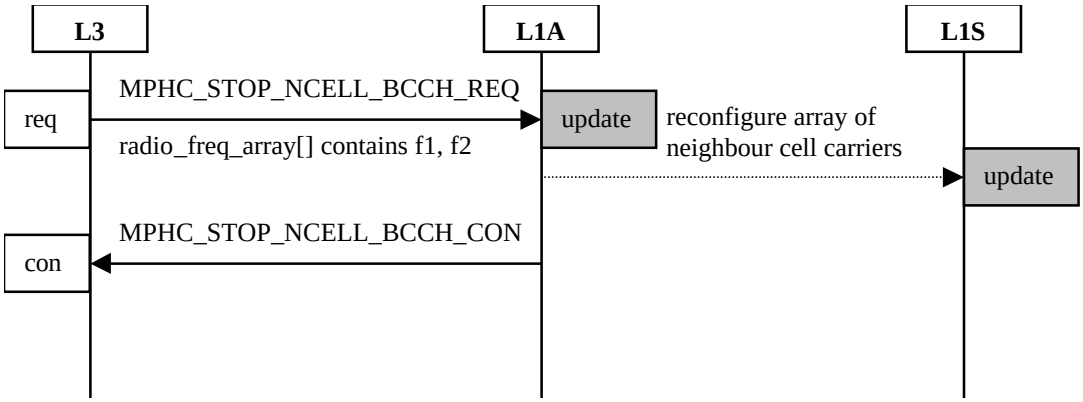


Figure 18: Neighbour cell BCCH reading, Partial STOP macro

The above figure represents a "STOP macro" which can be inserted at any place in the Figure 17. The running neighbor BCCH reading process is reconfigured, i.e. the BCCH reading of neighbor carriers f1 and f2 is stopped while the process continues for all other previously requested carriers fn.

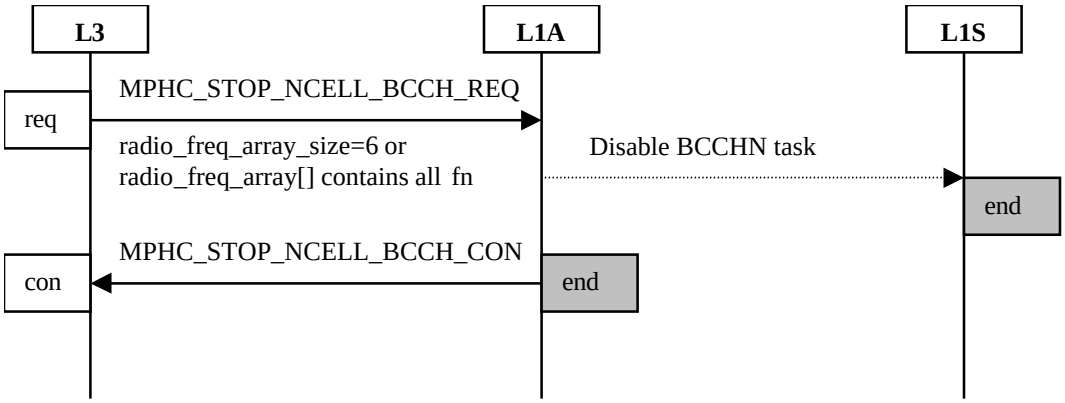


Figure 19: Neighbour cell BCCH reading, Full STOP macro

The above figure represents a generic "STOP macro" which can be inserted at any place in the Figure 17. The running neighbour BCCH reading process is stopped immediately either if radio_freq_array_size equals 6 (maximum number of requests to be served in parallel) or if radio_freq_array[] contains all carriers for which BCCH reading was requested previously.

8.1. MPHC_NCELL_BCCH_REQ

T_MPHC_NCELL_BCCH_REQ message type

- **radio_freq**
- **fn_offset**
- **time_alignment**
- **tsc**
- **bcch_blocks_required**

radio_freq (UWORD16)
Absolute radio frequency channel number of the required carrier.

fn_offset (UWORD32)
Difference in absolute frame numbers between the serving cell and the non-serving cell.
fn_offset is in the range 0 - 2715647 (0 to (max GSM frames – 1))

time_alignment (UWORD32)
Difference in quarter bits between the first bit in a frame of the serving cell and the first bit in the next frame of the non-serving cell.
time_alignment is in the range 0 - 4999.

tsc (UWORD8)
The training sequence code of the non-serving cell as described and coded in GSM 05.02 section 5.2.3.
tsc is in the range 0 - 7. The rest of the bits in this byte are set to zero.

bcch_blocks_required (UWORD16)
This word is a bitmap indicating the TC values during which layer 1 attempts to read the serving cell BCCH. The bitmap refers to both the normal BCCH blocks (in TDMA frames 2,3,4,5 in a MF51) and the extended BCCH blocks (in TDMA frames 6,7,8,9).
The bit numbers are mapped to the BCCH blocks as follows:

bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TC	7x	6x	5x	4x	3x	2x	1x	0x	7n	6n	5n	4n	3n	2n	1n	0n

where an 'x' represents extended blocks and 'n' represents normal blocks.

gprs_priority (UWORD8)(*)
0: TOP_PRIORITY, this request has priority over serving cell activity (either packet or circuit switched) and any other neighbor cells activity (used for GPRS). This priority is mandatory in packet transfer.

1: HIGH_PRIORITY, this request has priority over the neighbor cell BCCH reading with priority set to NORMAL_PRIORITY.

2: NORMAL_PRIORITY, this request has no special priority.

(*) this field exists only with a GPRS software. Otherwise the neighbor BCCH is scheduled with a normal priority.

8.2. MPHC_STOP_NCELL_BCCH_REQ

Notes:

- This stop message indicates the neighbor radio frequency L1 must remove from its neighbor BCCH reading pending list.
- Up to 6 neighbor cell radio frequencies can be contained. When 6 are provided the message is interpreted as a complete process stop request.

T_MPHC_STOP_NCELL_BCCH_REQ message type

- **radio_freq_array_size**
- **radio_freq_array[]**

radio_freq_array_size (UWORD8)
Number of entries in the radio_freq_array.

radio_freq_array[] (up to 6 x UWORD16)
Neighbor cell radio frequency array.

9. Periodically Report Cell Signal Strengths

Direction	Message name	Type
L3->L1	**MPHC_RXLEV_PERIODIC_REQ**	**T_MPHC_RXLEV_PERIODIC_REQ**
L3<-L1	**MPHC_RXLEV_PERIODIC_IND**	**T_MPHC_RXLEV_PERIODIC_IND**
L3->L1	**MPHC_STOP_RXLEV_PERIODIC_REQ**	**Trigger**
L3<-L1	**MPHC_STOP_RXLEV_PERIODIC_CON**	**Trigger**

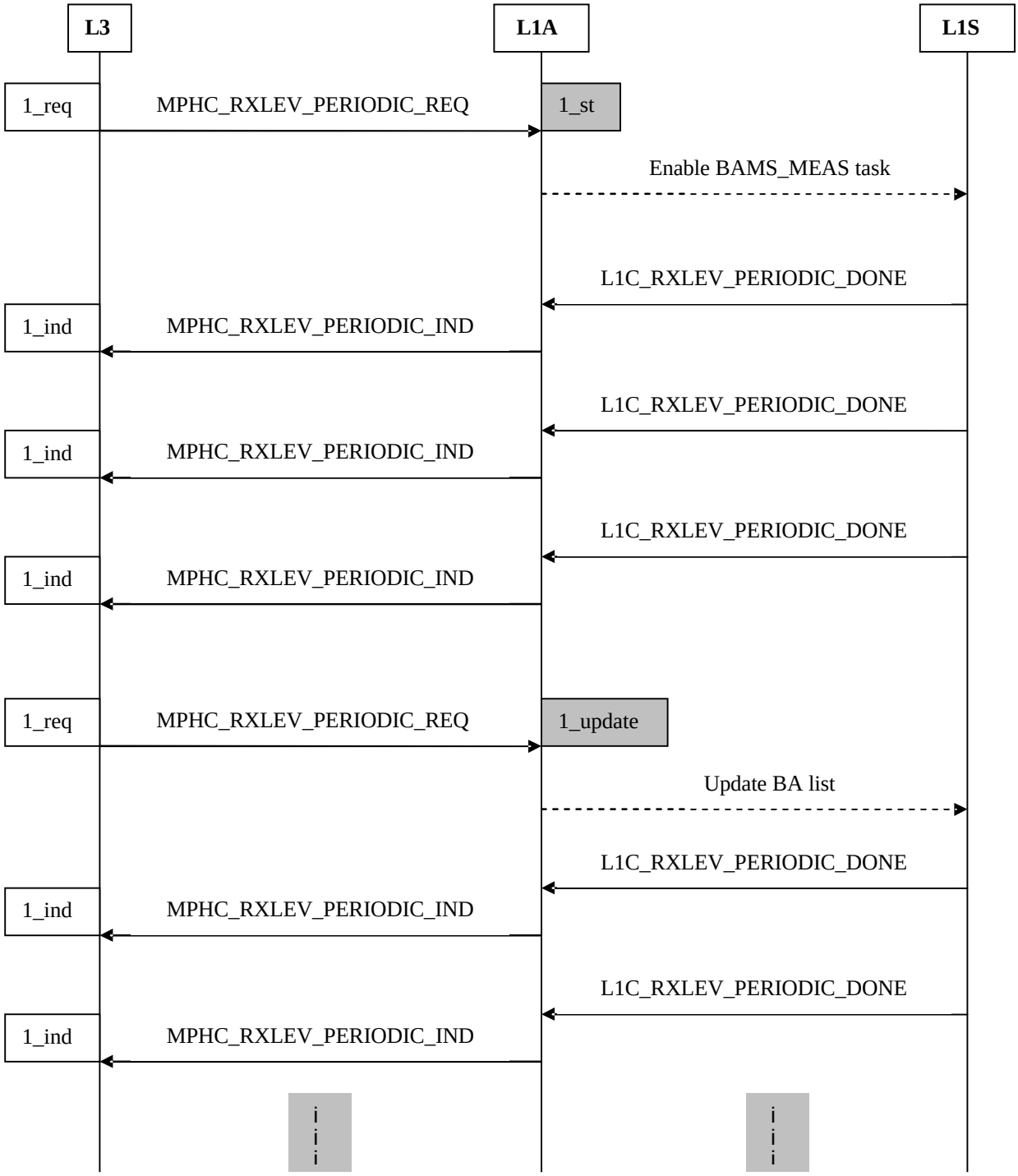


Figure 20: Periodically report cell signal strengths, normal flow followed by one update

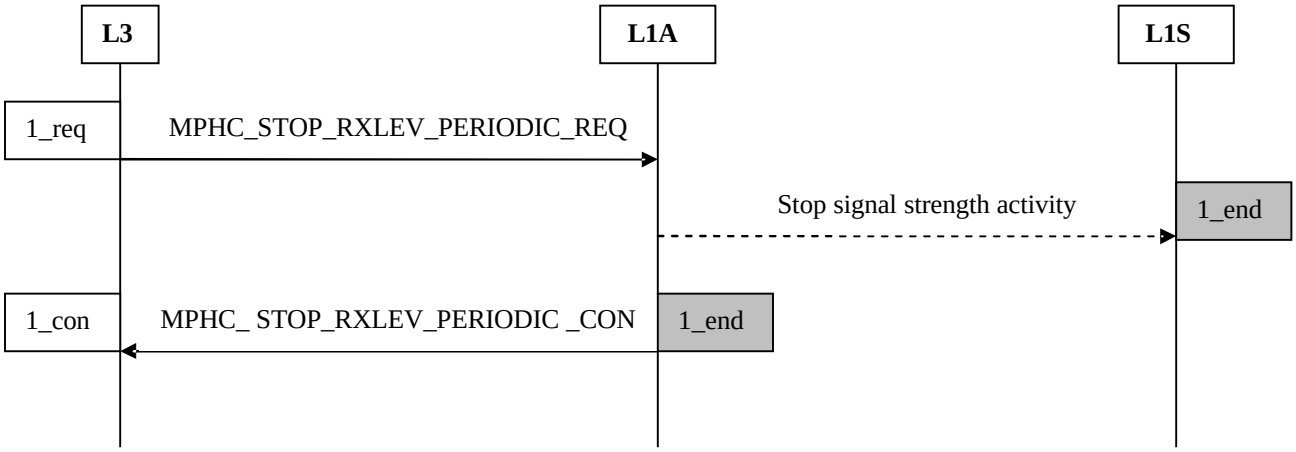


Figure 21: Periodically report cell signal strength, STOP macro

The above figure represents a generic "STOP macro" which can be inserted at any place in Figure 20. The periodic measurement process is stopped immediately.

MPHC_RXLEV_PERIODIC_REQ

T_MPHC_RXLEV_PERIODIC_REQ message type

- **chan_list**
- **num_of_chans**
- **ba_id**
- **next_radio_freq_measured**

chan_list (structure)
An array containing a BA list of up to 33 carrier radio frequencies (1 serving plus up to 32 neighbour cells).

A[32+1] (33 x UWORD16)
Radio frequency channel number.

num_of_chans (UWORD8)
The number of valid carrier numbers contained in the BA list (chan_number) array (1 to 33).

ba_id (UWORD8)
ba_id is an identifier for BA_LIST. A different ba_id must be provided for each list.
Value range 0 to 255.

next_radio_freq_measured (UWORD8)
Indicates the first position in the specified chan_list where the monitoring shall start (range 0-32 such that next_radio_freq_measured = 0 means the first result is expected for A[0]).

Comment: **The serving cell carrier is used for BA list measurements in order to keep track of its input level during dedicated mode such that the correct AGC setting is applied when switching back from dedicated to idle mode (without cell selection phase). As the BA list set in idle mode is reused in dedicated mode unless a MPHC_UPDATE_BA_LIST is posted, the serving cell carrier has to be contained in the BA list in idle mode**.

MPHC_RXLEV_PERIODIC_IND

T_MPHC_RXLEV_PERIODIC_IND message type

- **A[8]**
- **nbr_of_carriers**

- **s_rxlev**
- **ba_id**

A[8] (table of structure)
An array containing up to 8 neighbor cell carrier power strength levels measured during a PCH block. The array is in the same order as the BA list given in the MPHC_BA_LIST request message. Each element of the table follows the following strucutre:

radio_freq (UWORD16)
Radio frequency channel number.

rxlev (WORD8)
Store into which the result of a power measurements is accumulated.
The signal strength units used are based on those described in GSM 05.08 (ETS 300 911) section 8.1.4. The measurements continue above 63 and below 0:

signal_strength -1= -112dBm to -111dBm
signal_strength 0 = -111dBm to -110dBm
signal_strength 1 = -110dBm to -109dBm
(…)
signal_strength 63 = -48dBm to -47dBm
signal_strength 64 = -47dBm to -46dBm

nbr_of_carriers (UWORD8)
The number of cell carriers measured during the PCH block.

s_rxlev (WORD8)
The serving cell received power level (average of the 4 PCH bursts read during the particular PCH block).

ba_id (UWORD8)
Identifier corresponding to the list measured and reported in this message (seeMPHC_RXLEV_PERIODIC_REQ request message ba_id). Value range 0 to 255.

10. Power Measurement Request

This request is used to perform:

- Cell selection full power measurements
- PLMN search

Notes:

Direction	Message name	Type
L3->L1	**MPHC_RXLEV_REQ**	**T_MPHC_RXLEV_REQ**
L3<-L1	**MPHC_RXLEV_IND**	**T_MPHC_RXLEV_REQ**
L3->L1	**MPHC_STOP_RXLEV_REQ**	**Trigger**
L3<-L1	**MPHC_STOP_RXLEV_CON**	**Trigger**

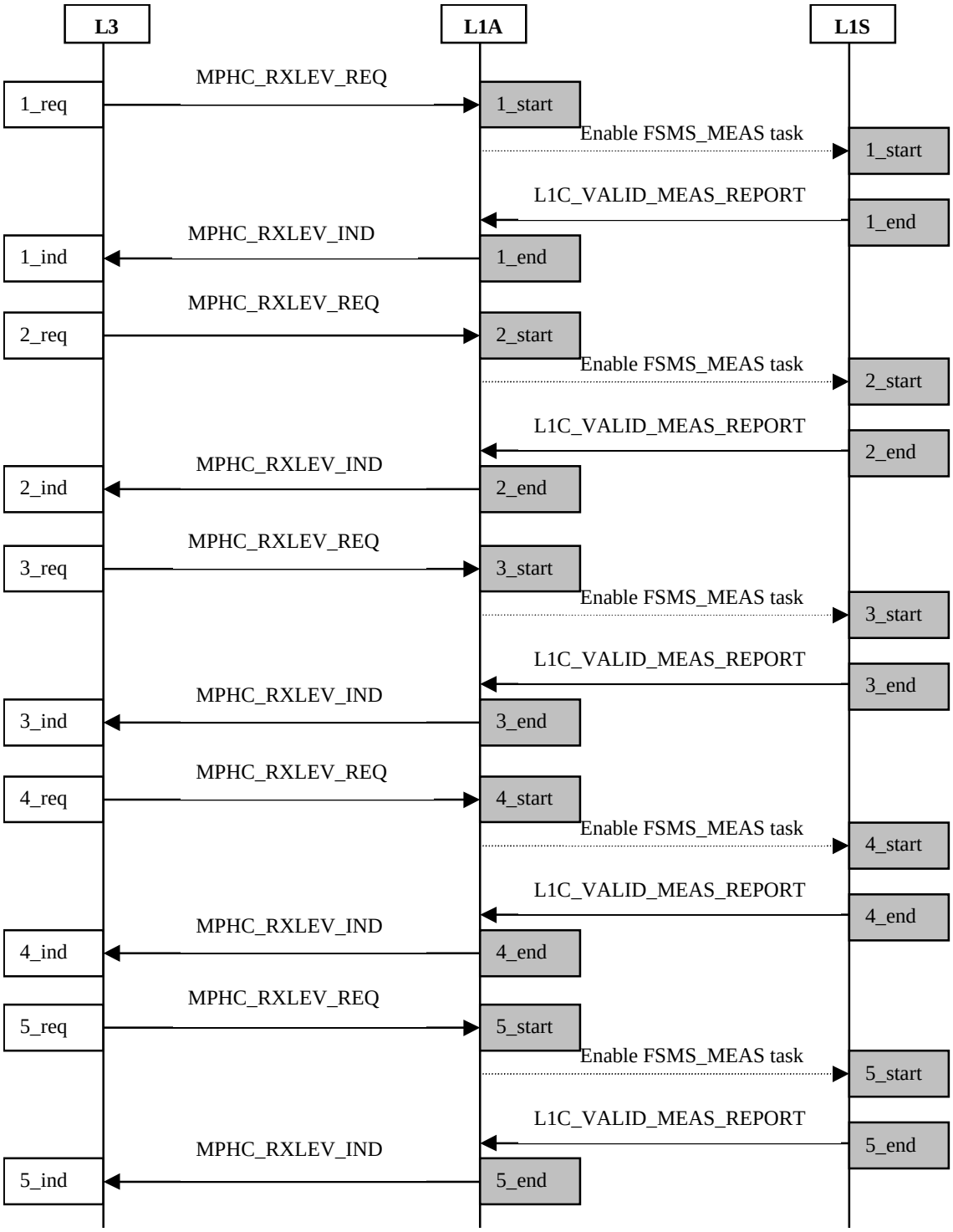


Figure 22: Power Measurement, normal flow

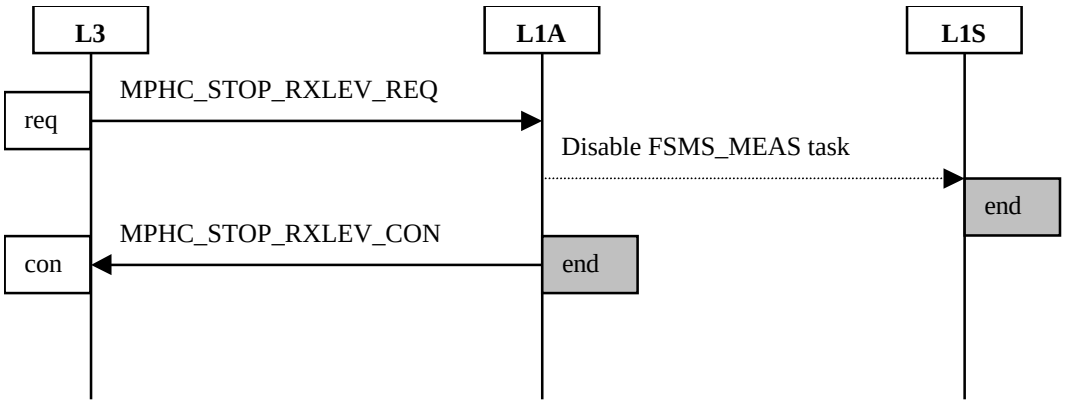


Figure 23: Power Measurement, STOP macro

The above figure represents a generic “STOP macro” which can be inserted at any place in the Figure 22. The running measurement process is stopped immediately.

10.1.MPHC_RXLEV_REQ / IND

T_MPHC_RXLEV_REQ message type

- **power_array_size**
- **power_array**

power_array_size (UWORD16)
Size of power_array specified as the number of entries in the array. This will often be the full list of radio frequencies used but can be in the range 1 – 548 (for dual band E-GSM/DCS).

power_array[] (table of structure)
A terminated array containing a list of carriers to be monitored and fields into which the results are accumulated.

radio_freq (UWORD16)
Radio frequency channel number.

accum_power_result (WORD16)
Store into which the result of a power measurements is accumulated.
The signal strength units used are based on those described in GSM 05.08 (ETS 300 911) section 8.1.4. The measurements continue above 63 and below 0:
(…)
signal_strength -1=-112dBm to -111dBm
signal_strength 0 =-111dBm to -110dBm
signal_strength 1 =-110dBm to -109dBm
(…)
signal_strength 63=-48dBm to -47dBm
signal_strength 64=-47dBm to -46dBm

Note1: To avoid dynamic allocation/de-allocation of the structure T_MPHC_RXLEV_REQ, L3 can, instead of usual partition pool usage, provide L1 with a pointer to an internal L3 table. This is achieved by setting **SigP** pointer to this table when building the MPHC_RXLEV_REQ message. During its measurement acquisition process, L1 will fill up this table by accumulating the signal strength within the **accum_power_result** fields. Those fields must be reset by L3 before sending the 1st request message to L1.

Note2: From the fact that various measurements can be performed in 1 TDMA (up to 4 in current implementation, more with next RF generation), it becomes a L3 responsibility to perform 5 sessions of measurement on the full list in a range of [3, 5s]. Full list measurement timing is reduced in L1 in order to fall on sleep mode as fast as possible. Full list measurement process is activated only when a MPHC_RXLEV_REQ is received. L3 has to manage the time between two requests.

11. Configure and Read Cell Broadcast Channel

Notes:

- Provided schedule is relative to the next "8*MF51" boundary.
- CBCH scheduling is described in GSM04.11 / GSM04.12.
- Normal CBCH and Extended CBCH are 2 independent processes that may run in parallel.

Direction	Message name	Type
L3->L1	**MPHC_CONFIG_CBCH_REQ**	**T_MPHC_CONFIG_CBCH_REQ**
L3->L1	**MPHC_CBCH_SCHEDULE_REQ**	**T_MPHC_CBCH_SCHEDULE_REQ**
L3->L1	**MPHC_CBCH_INFO_REQ**	**T_MPHC_CBCH_INFO_REQ**
L3->L1	**MPHC_CBCH_UPDATE_REQ**	**T_MPHC_CBCH_UPDATE_REQ**
L3->L1	**MPHC_STOP_CBCH_REQ**	**T_MPHC_STOP_CBCH_REQ**
L3<-L1	**MPHC_STOP_CBCH_CON**	**trigger**
L3<-L1	**MPHC_DATA_IND**	**T_MPHC_DATA_IND**

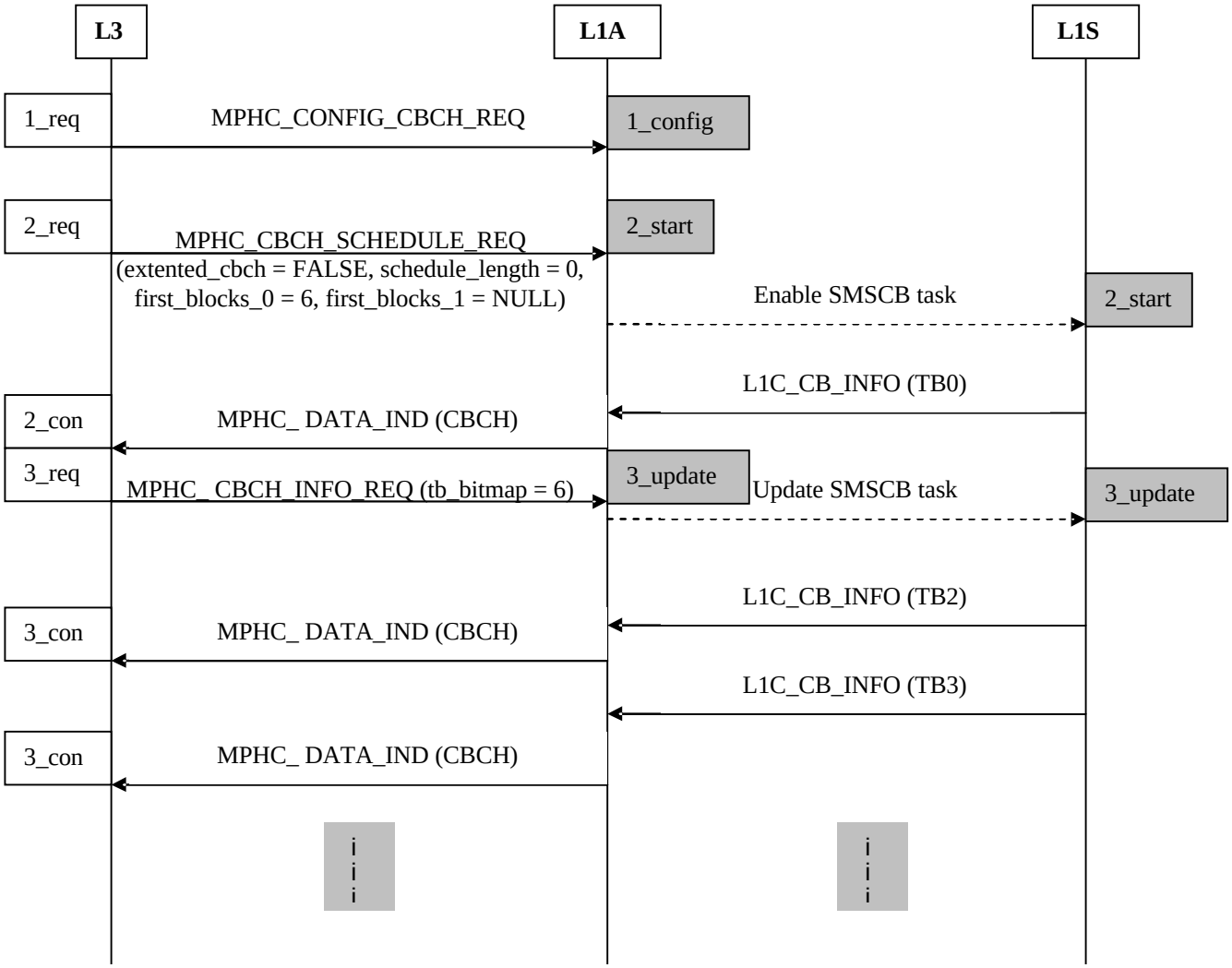


Figure 24: Configure and read cell broadcast channel

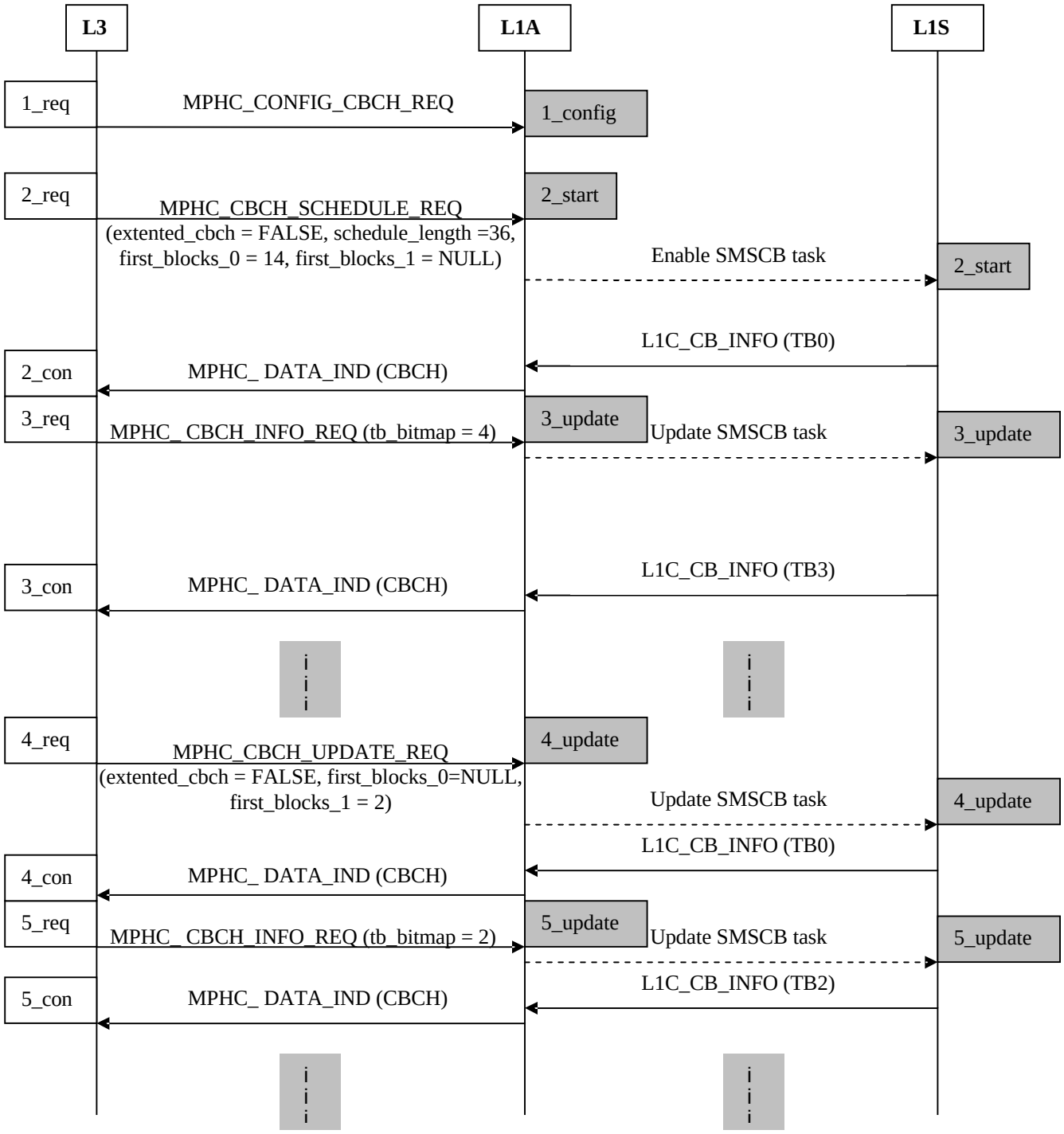


Figure 25: Configure and read cell broadcast channel

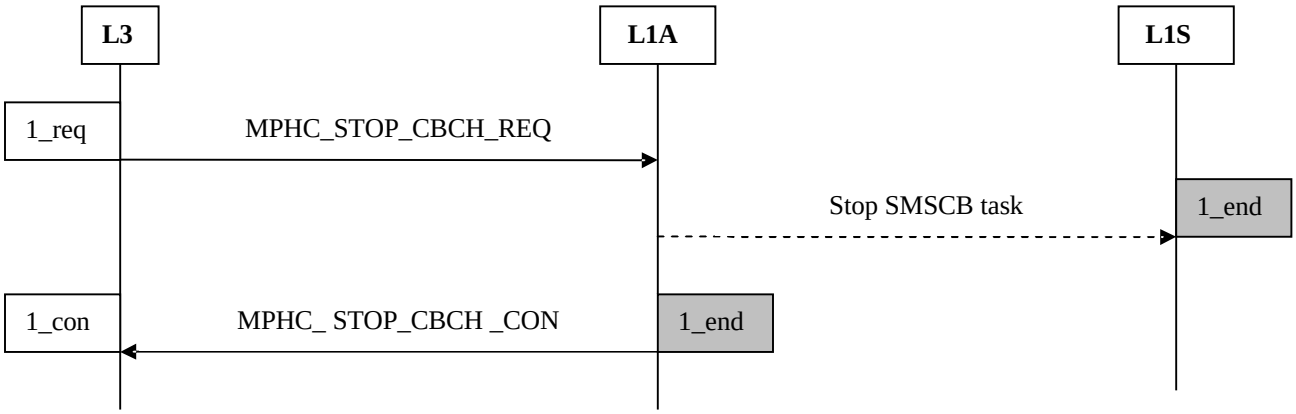


Figure 26: Read Cell Broadcast Channel, STOP macro

The above figure represents a generic "STOP macro" which can be inserted at any place in the Figure 25. The SMS process is stopped immediately.

11.1.MPHC_CONFIG_CBCH_REQ

T_MPHC_CONFIG_CBCH_REQ message structure:			
T_CHANNEL_DESCRIPTION **cbch_desc**			
	(BOOL) **cbch_desc.chan_sel.h** Hopping or Static RF channel (0 = Static, 1 = Hopping).		
	(union)	(UWORD16)	**cbch_desc.chan_sel.rf_channel.single_rf.radio_freq**
		(UWORD8)	**cbch_desc.chan_sel.rf_channel.hopping_rf.maio**
		(UWORD8)	**cbch_desc.chan_sel.rf_channel.hopping_rf.hsn**
	(UWORD8) **cbch_desc.channel_type** 3 = SDCCH/4, 4 = SDCCH/8.		
	(UWORD8) **cbch_desc.subchannel** Not applicable		
	(UWORD8) **cbch_desc.timeslot_no** Timeslot on which to configure the new channel (0-3)		
	(UWORD8) **cbch_desc.tsc** Training Sequence Code to select transmit midamble		
T_MOBILE_ALLOCATION **cbch_freq_list**			
	(UWORD16) **cbch_freq_list.rf_chan_cnt** Number of channels in MA list		
	(UWORD16) **cbch_freq_list.rf_chan_no.A[64]** MA List		

11.2.MPHC_CBCH_SCHEDULE_REQ

Notes:

- "schedule_length=0" is used to specify "No schedule present". In such a case, L1 starts reading continuously TB0 for normal CBCH or TB4 for Extended CBCH.
- MPHC_CBCH_SCHEDULE_REQ is used to provide L1 with a new CBCH scheduling which will start from the next "8*MF51" boundary.

T_MPHC_CBCH_SCHEDULE_REQ message type

- **extended_cbch**
- **schedule_length**
- **first_block_0**
- **fisrt_block_1**

extended_cbch (BOOL)
Indication as to which cell broadcast channel the request refers.
TRUE – request is for the extended cell broadcast channel
FALSE – request is for the normal cell broadcast channel

schedule_length (UWORD8)
Indicates the length of the schedule period described in the request
Range 0 – 48
Value 0 means no schedule provided. L1 is due to read continuously TB0 for normal CBCH or TB4 for Extended CBCH

first_blocks_0 (UWORD32)
Bitmap defining a set of blocks to be read during the schedule period. The bits represent the blocks as follows:

bit	MSB	30	29	28	.....	.....	3	2	1	LSB
block	32	31	30	29	.....	.....	4	3	2	1

first_blocks_1 (UWORD16)
Bitmap defining a set of blocks to be read during the schedule period. The bits represent the blocks as follows:

bit	MSB	14	13	12	.....	.....	3	2	1	LSB
block	48	47	46	45	.....	.....	36	35	34	33

11.3.MPHC_CBCH_INFO_REQ

Notes:

- Request to read some CBCH blocks after a CBCH/TB0 (or CBCH/TB4 for extended CBCH) has been read. This message is posted very quickly after TB0 (or TB4) reading, it will therefore never provoque any collision with an ongoing L1S CBCH reading.
- This message can be used to update on fly L1 with new tb_bitmap which then replaces the current one. In such a case, 0 in the bitmap means abort any pending reading of TB1/2/3 or TB5/6/7.
- a previous "schedule_length=0" is used to specify "No schedule present". In such a case, L1 starts reading continuously TB0 for normal CBCH or TB4 for Extended CBCH.

T_MPHC_CBCH_INFO_REQ message type

- **tb_bitmap**

tb_bitmap (UWORD8)
Indicates the TB (multi-frame-51 number) to be used to retrieved CBCH block.

bit	7(MSB)	6	5	4	3	2	1	0(LSB)
TB	-	-	7	6	5	3	2	1

A "1" means: request to read CBCH from this TB.

11.4. MPHC_CBCH_UPDATE_REQ

Notes:

- MPHC_CBCH_UPDATE_REQ is used to provide L1 with an update of the current scheduling without reconsidering the relative starting time of the ongoing scheduling.

T_MPHC_CBCH_UPDATE_REQ message type

- **extended_cbch**
- **first_block_0**
- **fisrt_block_1**

extended_cbch (BOOL)
Indication as to which cell broadcast channel the request refers.
TRUE – request is for the extended cell broadcast channel
FALSE – request is for the normal cell broadcast channel

first_blocks_0 (UWORD32)
Bitmap defining a set of blocks to be read during the schedule period. The bits represent the blocks as follows:

bit	MSB	30	29	28	.....	.....	3	2	1	LSB
block	32	31	30	29	.....	.....	4	3	2	1

first_blocks_1 (UWORD16)
Bitmap defining a set of blocks to be read during the schedule period. The bits represent the blocks as follows:

bit	MSB	14	13	12	.....	.....	3	2	1	LSB
block	48	47	46	45	.....	.....	36	35	34	33

11.5.MPHC_STOP_CBCH_REQ

T_MPHC_STOP_CBCH_REQ message type

- **normal_cbch**
- **extended_cbch**

normal_cbch (BOOL)
TRUE - stop normal cell broadcast decoding
FALSE - no effect on normal cell broadcast decoding

extended_cbch (BOOL)
TRUE - stop extended cell broadcast decoding
FALSE - no effect on extended cell broadcast decoding

12. New Serving Cell Selection

Notes:

Direction	Message name	Type
L3->L1	**MPHC_NEW_SCELL_REQ**	**T_MPHC_NEW_SCELL_REQ**
L3<-L1	**MPHC_NEW_SCELL_CON**	**trigger**

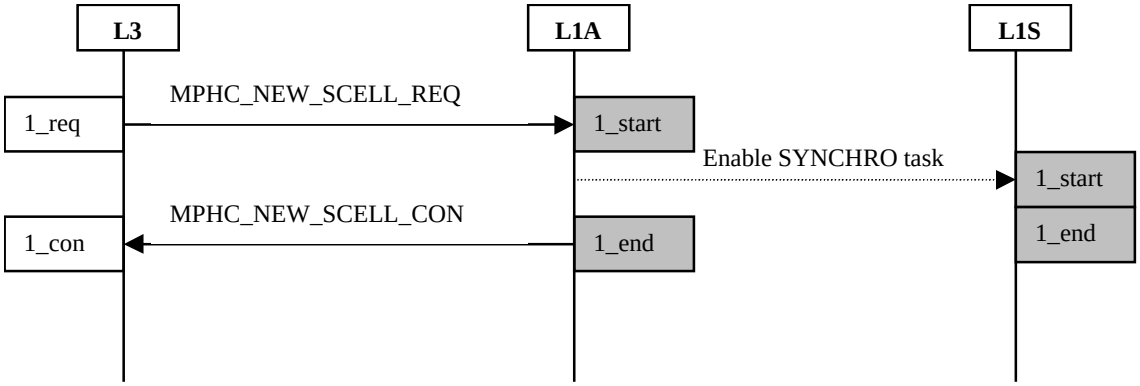


Figure 27: New Serving Cell Selection

12.1.MPHC_NEW_SCELL_REQ

T_MPHC_NEW_SCELL_REQ message type

- **radio_freq**
- **fn_offset**
- **time_alignment**
- **bsic**

radio_freq (UWORD16)
Radio frequency channel number of the required carrier.

fn_offset (UWORD32)
Difference in absolute frame numbers between the serving cell and the non-serving cell.
fn_offset is in the range 0 - 2715647 (0 to (max GSM frames – 1))

time_alignment (UWORD32)
Difference in quarter bits between the first bit in a frame of the serving cell and the first bit in the next frame of the non-serving cell.
time_alignment is in the range 0 - 4999.

bsic (UWORD8)
Base station identification code.

13. Connection Establishment

Direction	Message name	Type
L3->L1	**MPHC_RA_REQ**	**T_MPHC_RA_REQ**
L3<-L1	**MPHC_RA_CON**	**T_MPHC_RA_CON**
L3->L1	**MPHC_STOP_RA_REQ**	**trigger**
L3<-L1	**MPHC_STOP_RA_CON**	**trigger**

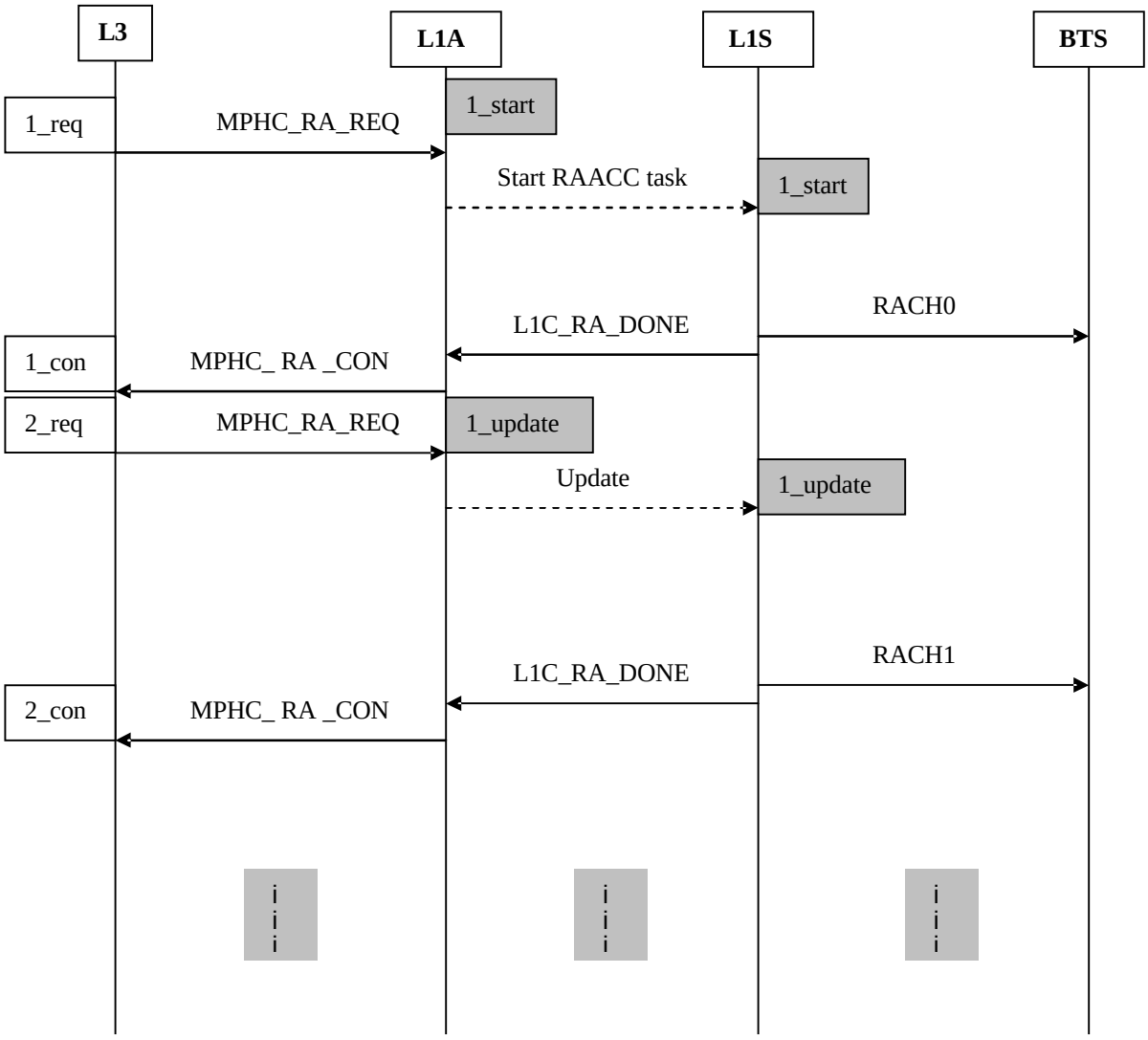


Figure 28: Connection Establishment, normal flow

L3 starts the connection establishment process by sending a first MPHC_RA_REQ message in which a delay is specified. L1 uses this delay as described in 04.08. On each update, L1 counter is loaded with the new delay.

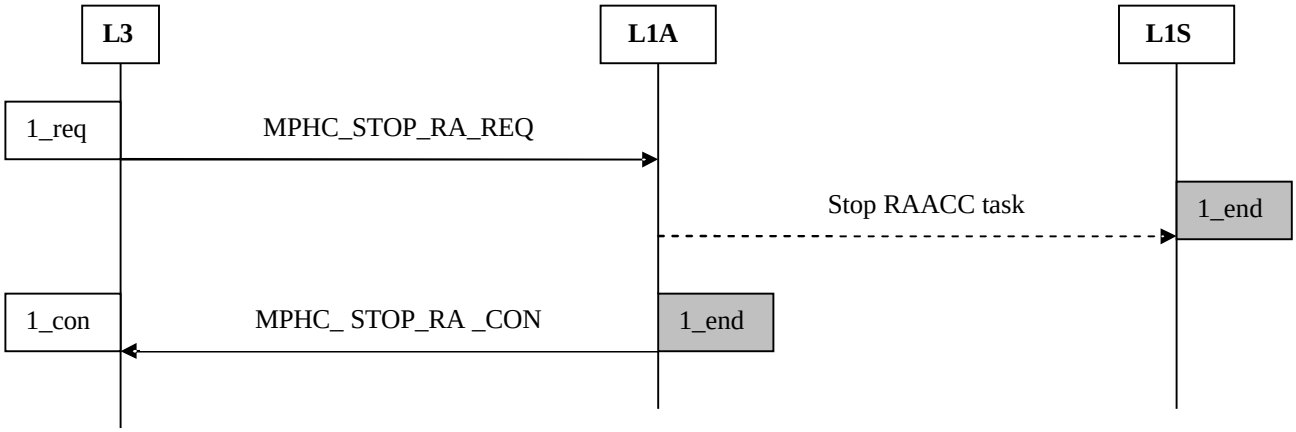


Figure 29: Connection Establishment, STOP macro

The above figure represents a generic "STOP macro" which can be inserted at any place in the Figure 28. Sending of RA bursts process, is stopped immediately.

Note: Stop must be sent to abort L1 process after:

a) receipt of an Immediat Assignment message or
b) receipt of an Immediat Assignment Reject message or
c) when the maximum number of RA bursts have been sent.

13.1.MPHC_RA_REQ

T_MPHC_RA_REQ message type

- **txpwr**
- **rand**
- **channel_request**
- **powerclass_band1**
- **powerclass_band2**

txpwr (UWORD8)
Indicates the power level which the mobile should use for transmission on the RACH. This is derived by layer 3 from information broadcast on the BCCH (broadcast in Cell Selection Parameters see GSM 04.08 section 10.5.2.4).
txpwr is coded as the binary representation of the power control level as defined in GSM 05.05 section 4.1 and is in the range 0 - 31.

rand (UWORD8)
Indicates the delay after which layer 1 sends each Random Access burst. The delay is specified as a number of empty slots belonging to the mobiles RACH between transmissions as defined in GSM 05.02 clause 7 and GSM 04.08 section 3.3.1.1.2. Layer 1 should start the first delay on the reception of the message from layer 3. Subsequent delays refer to the time since the last sent random access burst.

channel_request (UWORD8)
Data byte of the RA request.

powerclass_band1 (UWORD8)
GSM power class of the MS.

powerclass_band2 (UWORD8)
DCS power class of the MS.

13.2.MPHC_RA_CON

T_MPHC_RA_CON message type

- **fn**
- **channel_request**

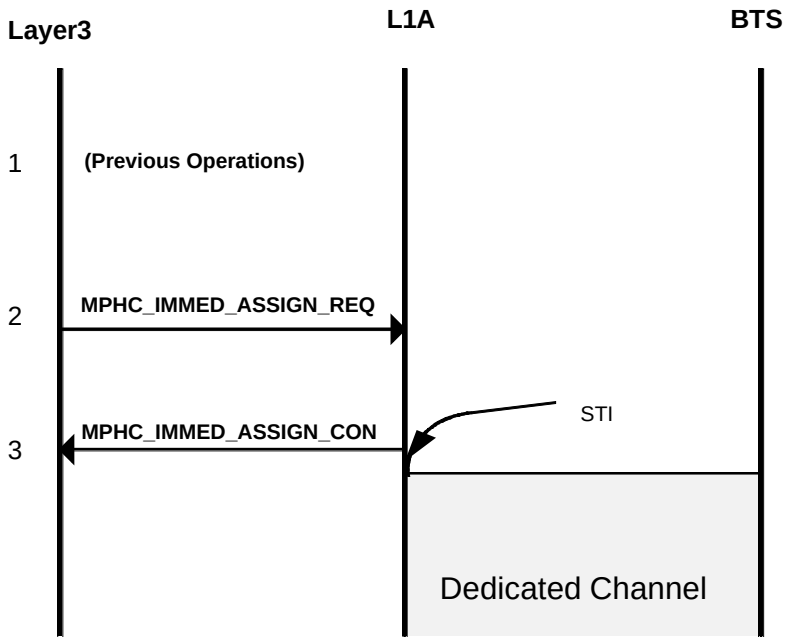
fn (UWORD32)
Full frame number of transmit block.
fn is in the range 0 – 2715647.

channel_request (UWORD8)
Data byte of the RA request.

14. Dedicated Immediate Assignment Interface

Immediate Assignment procedure (start of dedicated mode) is depicted below. This procedure will be carried out on the receipt of an IMMEDIATE ASSIGNMENT message from the network.

Immediate Assignment sequence



14.1. Prerequisites

It is a prerequisite that the MS has previously been in IDLE mode and has obtained the relevant Connection Establishment and Dedicated mode data.

14.2. Figure comments

1. Connection Establishment procedures result in an IMMEDIATE ASSIGNMENT message received by L3 (RRM).
2. L3 sets up the new channel description and starts dedicated mode by posting MPHC_IMMED_ASSIGN_REQ to the L1. L1A and L1S act upon this message to activate a new logical channel. A start time (STI) may be defined at some point in the future.

3. When the new logical channel activation is programmed, MPHC_IMMED_ASSIGN_CON is emitted by L1.

14.3. Message Structures

MPHC_IMMED_ASSIGN_REQ

T_MPHC_IMMED_ASSIGN_REQ message structure:		
T_CHANNEL_DESCRIPTION **channel_desc**		
	(BOOL) **channel_desc.chan_sel.h** Hopping or Static RF channel (0 = Static, 1 = Hopping).	
	(union)	(UWORD16) **channel_desc.chan_sel.rf_channel.single_rf.radio_freq**
		(UWORD8) **channel_desc.chan_sel.rf_channel.hopping_rf.maio**
		(UWORD8) **channel_desc.chan_sel.rf_channel.hopping_rf.hsn**
	(UWORD8) **channel_desc.channel_type** 0 = Invalid Channel, 1 = TCH/F, 2 = TCH/H, 3 = SDCCH/4, 4 = SDCCH/8.	
	(UWORD8) **channel_desc.subchannel** Subchannel for SDCCH/4 (0-3), SDCCH/8 (0-7) and TCH/H (0,1)	
	(UWORD8) **channel_desc.timeslot_no** Timeslot on which to configure the new channel (0-7)	
	(UWORD8) **channel_desc.tsc** Training Sequence Code to select transmit midamble	
(UWORD8) **timing_advance** Initial TA to use on new channel. Range: [0…63] bits		
T_MOBILE_ALLOCATION **frequency_list**		
	(UWORD16) **frequency_list.rf_chan_cnt** Number of channels in MA list	
	(UWORD16) **frequency_list.rf_chan_no.A[64]** MA List	
T_STARTING_TIME **starting_time**		
	(BOOL) **starting_time.start_time_present** Indicates if a starting time is specified for the new channel	
	(UWORD8) **starting_time.start_time.n32** Start frame number (T1' component)	
	(UWORD8) **starting_time.start_time.n51** Start frame number (T3 component)	
	(UWORD8) **starting_time.start_time.n26** Start frame number (T2 component)	
T_MOBILE_ALLOCATION **frequency_list_bef_sti**		

	(UWORD16) **frequency_list_bef_sti.rf_chan_cnt** Number of channels in MA list which is to be used before the start time.
	(UWORD16) **frequency_list_bef_sti.rf_chan_no.A[64]** MA list to be used before the start time.
(UWORD8) **maio_bef_sti** Mobile Allocation Index Offset to be used before the start time.	
(BOOL) **dtx_allowed** Indicates whether the MS will use uplink DTX or not (0 - MS must not use DTX, 1 - MS must use DTX).	
T_BCCH_LIST **bcch_allocation** **NOT USED IN CURRENT L1 IMPLEMENTATION**	
	(UWORD16) **bcch_allocation.num_of_chans** number of bcch in the list
	(UWORD16) **bcch_allocation.num_of_chans.A[31+1]** BCCH list
(UWORD8) **ba_id** **NOT USED IN CURRENT L1 IMPLEMENTATION**	
(BOOL) **pwrc** flag used to reject power measurements made on the serving cell when **radio_freq** is the beacon (see GSM05.08)	

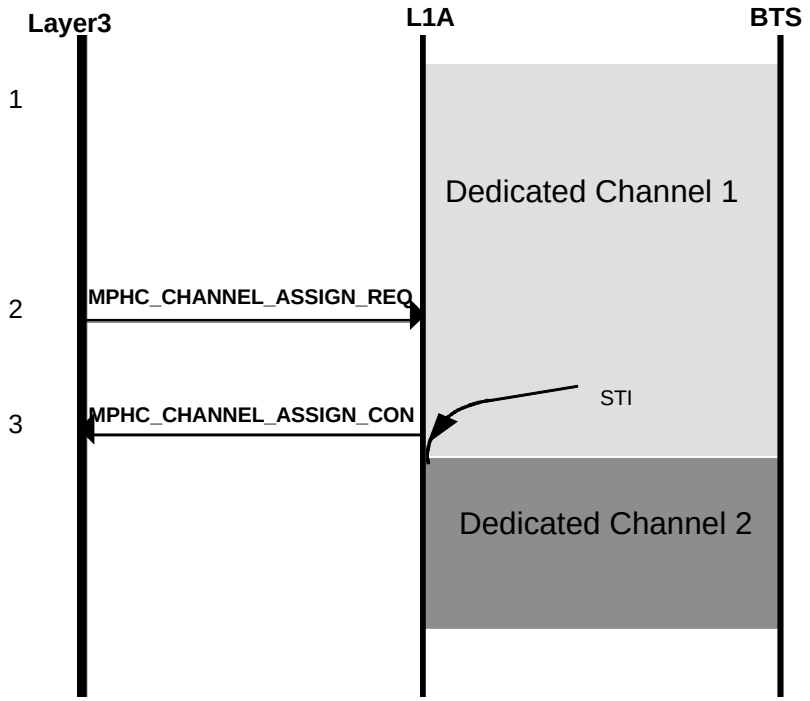
MPHC_IMMED_ASSIGN_CON

This message is a trigger without parameters.

15. Dedicated Channel Assignment Interface

The Channel Assignment procedure (physical and/or logical channel change during dedicated mode) is depicted below. This procedure will be carried out on the receipt of an ASSIGNMENT COMMAND message from the network.

Channel Assignment sequence



15.1. Prerequisites

It's a prerequisite that the MS is already on some dedicated channel.

15.2. Figure comments

1. MS is already operating on some dedicated channel.

2. L3 sets up the new channel description and posts MPHC_CHANNEL_ASSIGN_REQ to the L1. L1A and L1S act upon this message to deactivate the previous channel, and activate the new channel. A start time (STI) may be defined at some point in the future.
3. When the new channel is activation is programmed MPHC_CHANNEL_ASSIGN_CON is emitted by L1.

NOTE:

In case of AMR feature:

- All the abnormal case (c.f. 4.18 [1] 3.4.3.3, 3.4.6.1.3) and concerning the assignement, messages must be managed directly by the protocol stack. It means that the protocol stack checks the validity of the multirate parameters before to send it to the Layer1. In case of theses parameters are incorrect, the protocol stack doesn't send this new AMR configuration to the L1.
- The protocol stack must fill the all the multirate configuration parameters only if the channel mode is TCH/AFS or TCH/AHS.
- The AMR parameters are valid and exist only if channel_mode = TCH/AHS or TCH/AFS and if the DSP code support the AMR (code v34xx)).

15.3. Message Structures

MPHC_CHANNEL_ASSIGN_REQ

T_MPHC_CHANNEL_ASSIGN_REQ message structure:		
T_CHANNEL_DESCRIPTION **channel_desc_1**		
	(BOOL) **channel_desc_1.chan_sel.h** Hopping or Static RF channel (0 = Static, 1 = Hopping).	
	(union)	(UWORD16) **channel_desc_1.chan_sel.rf_channel.single_rf.radio_freq** Absolute Radio Frequency Channel Number (0 to 124, 975 to 1023)
		(UWORD8) **channel_desc_1.chan_sel.rf_channel.hopping_rf.maio** Mobile Allocation Index Offset
		(UWORD8) **channel_desc_1.chan_sel.rf_channel.hopping_rf.hsn** Hopping Sequence Number
	(UWORD8) **channel_desc_1.channel_type** 0 = Invalid Channel, 1 = TCH/F, 2 = TCH/H, 3 = SDCCH/4, 4 = SDCCH/8.	
	(UWORD8) **channel_desc_1.subchannel** Subchannel for SDCCH/4 (0-3), SDCCH/8 (0-7) and TCH/H (0,1)	
	(UWORD8) **channel_desc_1.timeslot_no** Timeslot on which to configure the new channel (0-7)	
	(UWORD8) **channel_desc_1.tsc** Training Sequence Code to select transmit midamble	
(UWORD8) **channel_mode_1** 0 = Signalling Only, 1 = TCH/FS, 2 = TCH/HS, 3 = TCH/96, 4 = TCH/F48, 5 = TCH/H48, 6 = TCH/F24, 7 = TCH/H24, 8 = TCH/EFR, 9 = TCH/144, 10 = TCH/AHS, 11 = TCH/AFS.		
(UWORD8) **txpwr** Transmit power		
T_MOBILE_ALLOCATION **frequency_list**		
	(UWORD16) **frequency_list.rf_chan_cnt** Number of channels in MA list	
	(UWORD16) **frequency_list.rf_chan_no.A[64]** MA List	
T_STARTING_TIME **starting_time**		
	(BOOL) **starting_time.start_time_present** Indicates if a starting time is specified for the new channel	

	(UWORD8) **starting_time.start_time.n32** Start frame number (T1' component)	
	(UWORD8) **starting_time.start_time.n51** Start frame number (T3 component)	
	(UWORD8) **starting_time.start_time.n26** Start frame number (T2 component)	
T_CHANNEL_DESCRIPTION **channel_desc2**		
	(BOOL) **channel_desc2.chan_sel.h** Hopping or Static RF channel (0 = Static, 1 = Hopping).	
	(union)	(UWORD16) **channel_desc_2.chan_sel.rf_channel.single_rf.radio_freq** Absolute Radio Frequency Channel Number (0 to 124, 975 to 1023)
		(UWORD8) **channel_desc_2.chan_sel.rf_channel.hopping_rf.maio** Mobile Allocation Index Offset
		(UWORD8) **channel_desc_2.chan_sel.rf_channel.hopping_rf.hsn** Hopping Sequence Number
	(UWORD8) **channel_desc_2.channel_type** 0 = Invalid Channel, 1 = TCH/F, 2 = TCH/H, 3 = SDCCH/4, 4 = SDCCH/8.	
	(UWORD8) **channel_desc_2.subchannel** Subchannel for SDCCH/4 (0-3), SDCCH/8 (0-7) and TCH/H (0,1)	
	(UWORD8) **channel_desc_2.timeslot_no** Timeslot on which to configure the new channel (0-7)	
	(UWORD8) **channel_desc_2.tsc** Training Sequence Code to select transmit midamble	
(UWORD8) **channel_mode_2** 0 = Signalling Only, 1 = TCH/FS, 2 = TCH/HS, 3 = TCH/96, 4 = TCH/F48, 5 = TCH/H48, 6 = TCH/F24, 7 = TCH/H24, 8 = TCH/EFR, 9 = TCH/144, 10 = TCH/AHS, 11 = TCH/AFS.		
T_MOBILE_ALLOCATION **frequency_list_bef_sti**		
	(UWORD16) **frequency_list_bef_sti.rf_chan_cnt** Number of channels in MA list which is to be used before the start time.	
	(UWORD16) **frequency_list_bef_sti.rf_chan_no.A[64]** MA list to be used before the start time.	
T_CHANNEL_DESCRIPTION **channel_desc_1_bef_sti**		
	(BOOL) **channel_desc_1_bef_sti.chan_sel.h** Hopping or Static RF channel (0 = Static, 1 = Hopping).	
	(union)	(UWORD16)**channel_desc_1_bef_sti.chan_sel.rf_channel.single_rf.radio_freq** Absolute Radio Frequency Channel Number (0 to 124, 975 to 1023)

		(UWORD8)**channel_desc_1_bef_sti.chan_sel.rf_channel.hopping_rf.maio** Mobile Allocation Index Offset
		(UWORD8)**channel_desc_1_bef_sti.chan_sel.rf_channel.hopping_rf.hsn** Hopping Sequence Number
	(UWORD8) **channel_desc_1_bef_sti.channel_type** 0 = Invalid Channel, 1 = TCH/F, 2 = TCH/H, 3 = SDCCH/4, 4 = SDCCH/8.	
	(UWORD8) **channel_desc_1_bef_sti.subchannel** Subchannel for SDCCH/4 (0-3), SDCCH/8 (0-7) and TCH/H (0,1)	
	(UWORD8) **channel_desc_1_bef_sti.timeslot_no** Timeslot on which to configure the new channel (0-7)	
	(UWORD8) **channel_desc_1_bef_sti.tsc** Training Sequence Code to select transmit midamble	
T_CHANNEL_DESCRIPTION **channel_desc_2_bef_sti**		
	(BOOL) **channel_desc_2_bef_sti.chan_sel.h** Hopping or Static RF channel (0 = Static, 1 = Hopping).	
	(union)	(UWORD16)**channel_desc_2_bef_sti.chan_sel.rf_channel.single_rf.radio_freq** Absolute Radio Frequency Channel Number (0 to 124, 975 to 1023)
		(UWORD8)**channel_desc_2_bef_sti.chan_sel.rf_channel.hopping_rf.maio** Mobile Allocation Index Offset
		(UWORD8)**channel_desc_2_bef_sti.chan_sel.rf_channel.hopping_rf.hsn** Hopping Sequence Number
	(UWORD8) **channel_desc_2_bef_sti.channel_type** 0 = Invalid Channel, 1 = TCH/F, 2 = TCH/H, 3 = SDCCH/4, 4 = SDCCH/8.	
	(UWORD8) **channel_desc_2_bef_sti.subchannel** Subchannel for SDCCH/4 (0-3), SDCCH/8 (0-7) and TCH/H (0,1)	
	(UWORD8) **channel_desc_2_bef_sti.timeslot_no** Timeslot on which to configure the new channel (0-7)	
	(UWORD8) **channel_desc_2_bef_sti.tsc** Training Sequence Code to select transmit midamble	
(UWORD8) **cipher_mode** 0 = No Cyphering, !0 = Cyphering ON.		
(UWORD8) **a5_algorithm** 0 = A5/1, 1 = A5/2, 2 = A5/3, 3 = A5/4, 4 = A5/5, 5 = A5/6, 6 = A5/7		
(UWORD8) **cipher_key.A[8]** ciphering key.		
(BOOL) **dtx_allowed**		

Indicates whether the MS will use uplink DTX or not (0 - MS must not use DTX, 1 - MS must use DTX).

T_AMR_CONFIGURATION **amr_configuration (these following parameters are valid and exist only if channel_mode = TCH/AHS or TCH/AFS and if the DSP code support the AMR (code v34xx)).**

(BOOL) **amr_configuration.noise_suppression_bit**

For instance the role of this flag is unknown but the possible values are:

Value	Legend
FALSE	Noise Suppression can be used (default)
TRUE	Noise Suppression shall be turned off

(BOOL) **amr_configuration.initial_codec_mode_indicator**

Indicates the rule to choose the initial codec mode.

initial_codec_mode_indicator = FALSE: the initial codec mode is defined by the implicit rule provided in GSM 05.09 (3.4.3).

initial_codec_mode_indicator = TRUE: the initial codec mode is defined by the Start Mode field (i.e. value initial_codec_mode).

Value	Legend
FALSE	The initial codec mode is defined by the implicit rule provided in GSM 05.09 (3.4.3).
TRUE	The initial codec mode is defined by the Start Mode field (i.e. value initial_codec_mode).

(UWORD8) **amr_configuration.initial_codec_mode**

Indicates which codec mode are used in the initialise phase in case of *initial_codec_mode_indicator = TRUE*. The possible value are defined in the GSM5.09 section 3.4.1 and 5.03 section 3.10:

Identifier	Value	Legend
CODEC_MODE_1	0	Represents the lowest codec mode (lowest bit-rate) of the ACS.
CODEC_MODE_2	1	Represents the second lowest mode, if the ACS includes more than one mode.
CODEC_MODE_3	2	Represents the third lowest mode, if the ACS includes more than two modes.
CODEC_MODE_4	3	Represents the highest mode, if the ACS includes four modes.

(UWORD8) **amr_configuration.active_codec_set**

Indicates which voice codec will be used. This value is a bit field, if a bit corresponding to a voice codec is equal to 1, this codec is used in this AMR configuration otherwise is not used. The correspondence between the bit number and the voice codec is the following:

Bit number	Vocoder
8 (MSB)	12,2 kbit/s codec rate.
7	10,2 kbit/s codec rate.
6	7,95 kbit/s codec rate.
5	7,40 kbit/s codec rate.
4	6,70 kbit/s codec rate.
3	5,90 kbit/s codec rate.
2	5,15 kbit/s codec rate.

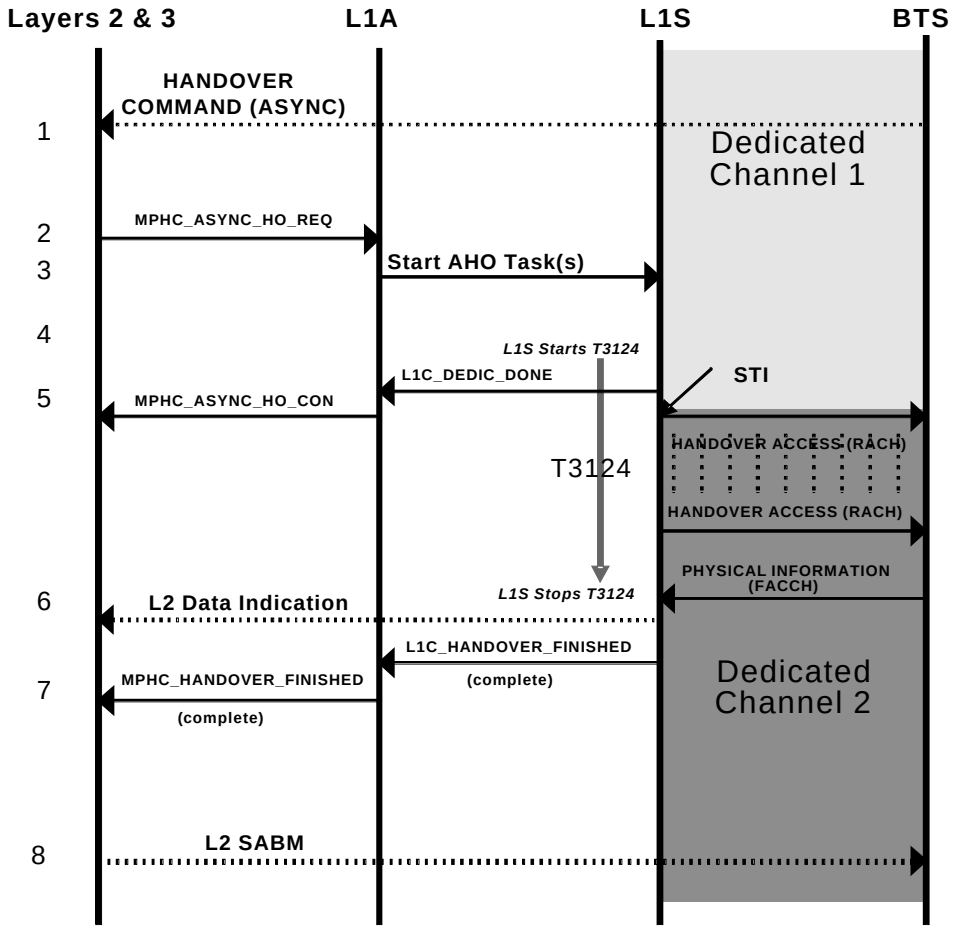
1 (LSB)	4,75 kbit/s codec rate.
(UWORD8) **amr_configuration.threshold[3]** Indicates the AMR lower thresholds values for switching between the voice codec mode j and j-1. The range of each threshold is from 0 to 63 and the number of threshold is from 0 to 3.	
(UWORD8) **amr_configuration.hysteresis[3]** Indicates the AMR hysteresis values to obtain the higher thresholds for switching between the voice codec mode j and j+1. The range of each hysteresis is from 0 to 15 and the number of hysteresis is from 1 to 3.	

MPHC_CHANNEL_ASSIGN_CON

This message is a trigger without parameters.

16. Asynchronous Handover Procedure (Successful)

Asynchronous Handover sequence



16.1. Prerequisites

Its a prerequisite that the MS is already on some dedicated channel.

16.2. Figure comments

1. The Network commands an asynchronous handover on the downlink main DCCH.
2. L3 posts an MPHC_ASYNC_HO_REQ message to L1.
3. L1A starts L1S tasks to perform the handover operation. L1S waits for the starting time (if specified), otherwise it starts the handover immediately. Once the L1S programs the events to be done after STI, it immediately sends L1C_DEDIC_DONE to L1A. In turn L1A creates an MPHC_ASYNC_HO_CON message and posts it to L3. Thus HO confirmation is sent before the STI itself.
4. L1S deactivates any previously active dedicated channels, starts timer T3124 and repeatedly sends the HANDOVER ACCESS message in an access burst on the main DCCH of the new logical channel. L1S also activates normal downlink reception on the new channel and waits for a FACCH indication (PHYSICAL INFORMATION) from the network.
5. Upon successful reception of PHYSICAL INFORMATION on the downlink FACCH, L1S extracts the timing advance information for the new channel and stops T3124. The PHYSICAL INFORMATION frame is also conveyed to the L2 via the DCCH/FACCH mechanism described elsewhere in this document. L1S constructs a L1C_HANDOVER_FINISHED with *cause* 'complete' and sends it to L1A.
6. L1A constructs MPHC_HANDOVER_FINISHED with *cause* 'complete' and posts it to L3.
7. Layer 2 establishment can be started on the new channel.

Notes:

The connection failure timer T3124 is maintained in L1S.

L1S waits for the reception of PHYSICAL INFORMATION, then directly extracts the timing advance for the new channel.

Since L3 maintains the serving/Neighbour time difference data base, L3 is able to fill the OTD (Observed Time Difference) field in the HANDOVER COMPLETE message sent to the BTS on FACCH/UL.

In case of AMR feature:

- All the abnormal case (c.f. 4.18 [1] 3.4.3.3, 3.4.6.1.3) and concerning the handover messages must be managed directly by the protocol stack. It means that the protocol stack checks the validity of the multirate parameters before to send it to the Layer1. In case of theses parameters are incorrect, the protocol stack doesn't send this new AMR configuration to the L1.
- If a handover message is received by the protocol stack and only the multirate channel mode is specified (c.f. 4.18 [1], 3.4.4.1, 9.1.15.13), the previous multirate configuration must be sent again to the L1 with the handover message.
- The protocol stack must fill the all the multirate configuration parameters only if the channel mode is TCH/AFS or TCH/AHS.
- The AMR parameters are valid and exist only if channel_mode = TCH/AHS or TCH/AFS and if the DSP code support the AMR (code v34xx)).

16.3. Message Structures

MPHC_ASYNC_HO_REQ

T_MPHC_ASYNC_HO_REQ message structure:		
T_HO_PARAMS **handover_command**		
	(UWORD8) **cell_description.ncc**	
	(UWORD8) **cell_description.bcc**	
	(UWORD16) **cell_description.bcch_carrier**	
	(BOOL) **channel_desc_1.chan_sel.h** Hopping or Static RF channel (0 = Static, 1 = Hopping).	
	(union)	(UWORD16) **channel_desc_1.chan_sel.rf_channel.single_rf.radio_freq** Absolute Radio Frequency Channel Number (0 to 124, 975 to 1023)
		(UWORD8) **channel_desc_1.chan_sel.rf_channel.hopping_rf.maio** Mobile Allocation Index Offset
		(UWORD8) **channel_desc_1.chan_sel.rf_channel.hopping_rf.hsn** Hopping Sequence Number
	(UWORD8) **channel_desc_1.channel_type** 0 = Invalid Channel, 1 = TCH/F, 2 = TCH/H, 3 = SDCCH/4, 4 = SDCCH/8.	
	(UWORD8) **channel_desc_1.subchannel** Subchannel for SDCCH/4 (0-3), SDCCH/8 (0-7) and TCH/H (0,1)	
	(UWORD8) **channel_desc_1.timeslot_no** Timeslot on which to configure the new channel (0-7)	
	(UWORD8) **channel_desc_1.tsc** Training Sequence Code to select transmit midamble	
	(UWORD8) **channel_mode_1** 0 = Signalling Only, 1 = TCH/FS, 2 = TCH/HS, 3 = TCH/96, 4 = TCH/F48, 5 = TCH/H48, 6 = TCH/F24, 7 = TCH/H24, 8 = TCH/EFR, 9 = TCH/144, 10 = TCH/AHS, 11 = TCH/AFS.	
	(BOOL) **starting_time.start_time_present** Indicates if a starting time is specified for the new channel	
	(UWORD8) **starting_time.start_time.n32** Start frame number (T1' component)	
	(UWORD8) **starting_time.start_time.n51** Start frame number (T3 component)	
	(UWORD8) **starting_time.start_time.n26** Start frame number (T2 component)	
	(UWORD8) **ho_acc** Handover RACH data.	

(UWORD8) **txpwr** Transmit power	
(UWORD8) **report_time_diff** RTD (see GSM05.10).	
(UWORD16) **frequency_list.rf_chan_cnt** Number of channels in MA list	
(UWORD16) **frequency_list.rf_chan_no.A[64]** MA List	
(BOOL) **channel_desc2.chan_sel.h** Hopping or Static RF channel (0 = Static, 1 = Hopping).	
(union)	(UWORD16) **channel_desc_2.chan_sel.rf_channel.single_rf.radio_freq** Absolute Radio Frequency Channel Number (0 to 124, 975 to 1023)
	(UWORD8) **channel_desc_2.chan_sel.rf_channel.hopping_rf.maio** Mobile Allocation Index Offset
	(UWORD8) **channel_desc_2.chan_sel.rf_channel.hopping_rf.hsn** Hopping Sequence Number
(UWORD8) **channel_desc_2.channel_type** 0 = Invalid Channel, 1 = TCH/F, 2 = TCH/H, 3 = SDCCH/4, 4 = SDCCH/8.	
(UWORD8) **channel_desc_2.subchannel** Subchannel for SDCCH/4 (0-3), SDCCH/8 (0-7) and TCH/H (0,1)	
(UWORD8) **channel_desc_2.timeslot_no** Timeslot on which to configure the new channel (0-7)	
(UWORD8) **channel_desc_2.tsc** Training Sequence Code to select transmit midamble	
(UWORD8) **channel_mode_2** 0 = Signalling Only, 1 = TCH/FS, 2 = TCH/HS, 3 = TCH/96, 4 = TCH/F48, 5 = TCH/H48, 6 = TCH/F24, 7 = TCH/H24, 8 = TCH/EFR, 9 = TCH/144, 10 = TCH/AHS, 11 = TCH/AFS.	
(UWORD16) **frequency_list_bef_sti.rf_chan_cnt** Number of channels in MA list which is to be used before the start time.	
(UWORD16) **frequency_list_bef_sti.rf_chan_no.A[64]** MA list to be used before the start time.	
(BOOL) **channel_desc_1_bef_sti.chan_sel.h** Hopping or Static RF channel (0 = Static, 1 = Hopping).	
(union)	(UWORD16)**channel_desc_1_bef_sti.chan_sel.rf_channel.single_rf.radio_freq** Absolute Radio Frequency Channel Number (0 to 124, 975 to 1023)
	(UWORD8)**channel_desc_1_bef_sti.chan_sel.rf_channel.hopping_rf.maio** Mobile Allocation Index Offset

		(UWORD8)**channel_desc_1_bef_sti.chan_sel.rf_channel.hopping_rf.hsn** Hopping Sequence Number
	(UWORD8) **channel_desc_1_bef_sti.channel_type** 0 = Invalid Channel, 1 = TCH/F, 2 = TCH/H, 3 = SDCCH/4, 4 = SDCCH/8.	
	(UWORD8) **channel_desc_1_bef_sti.subchannel** Subchannel for SDCCH/4 (0-3), SDCCH/8 (0-7) and TCH/H (0,1)	
	(UWORD8) **channel_desc_1_bef_sti.timeslot_no** Timeslot on which to configure the new channel (0-7)	
	(UWORD8) **channel_desc_1_bef_sti.tsc** Training Sequence Code to select transmit midamble	
	(BOOL) **channel_desc_2_bef_sti.chan_sel.h** Hopping or Static RF channel (0 = Static, 1 = Hopping).	
	(union)	(UWORD16)**channel_desc_2_bef_sti.chan_sel.rf_channel.single_rf.radio_freq** Absolute Radio Frequency Channel Number (0 to 124, 975 to 1023)
		(UWORD8)**channel_desc_2_bef_sti.chan_sel.rf_channel.hopping_rf.maio** Mobile Allocation Index Offset
		(UWORD8)**channel_desc_2_bef_sti.chan_sel.rf_channel.hopping_rf.hsn** Hopping Sequence Number
	(UWORD8) **channel_desc_2_bef_sti.channel_type** 0 = Invalid Channel, 1 = TCH/F, 2 = TCH/H, 3 = SDCCH/4, 4 = SDCCH/8.	
	(UWORD8) **channel_desc_2_bef_sti.subchannel** Subchannel for SDCCH/4 (0-3), SDCCH/8 (0-7) and TCH/H (0,1)	
	(UWORD8) **channel_desc_2_bef_sti.timeslot_no** Timeslot on which to configure the new channel (0-7)	
	(UWORD8) **channel_desc_2_bef_sti.tsc** Training Sequence Code to select transmit midamble	
	(UWORD8) **cipher_mode** 0 = No Cyphering, !0 = Cyphering ON.	
	(UWORD8) **a5_algorithm** 0 = A5/1, 1 = A5/2, 2 = A5/3, 3 = A5/4, 4 = A5/5, 5 = A5/6, 6 = A5/7	
(UWORD32) **fn_offset** Difference in frame number between serving and handover destination cell (0 to MAX_FN-1).		
(UWORD32) **time_alignmnt** Difference in QBO between serving and handover destination cell (0 to 5000)		
(UWORD8) **cipher_key.A[8]** ciphering key.		

T_AMR_CONFIGURATION **amr_configuration (these following parameters are valid and exist only if channel_mode = TCH/AHS or TCH/AFS and if the DSP code support the AMR (code v34xx)).**

(BOOL) **amr_configuration.noise_suppression_bit**

For instance the role of this flag is unknown but the possible values are:

Value	Legend
FALSE	Noise Suppression can be used (default)
TRUE	Noise Suppression shall be turned off

(BOOL) **amr_configuration.initial_codec_mode_indicator**

Indicates the rule to choose the initial codec mode.

initial_codec_mode_indicator = FALSE: the initial codec mode is defined by the implicit rule provided in GSM 05.09 (3.4.3).

initial_codec_mode_indicator = TRUE: the initial codec mode is defined by the Start Mode field (i.e. value initial_codec_mode).

Value	Legend
FALSE	The initial codec mode is defined by the implicit rule provided in GSM 05.09 (3.4.3).
TRUE	The initial codec mode is defined by the Start Mode field (i.e. value initial_codec_mode).

(UWORD8) **amr_configuration.initial_codec_mode**

Indicates which codec mode are used in the initialise phase in case of *initial_codec_mode_indicator = TRUE*. The possible value are defined in the GSM5.09 section 3.4.1 and 5.03 section 3.10:

Identifier	Value	Legend
CODEC_MODE_1	0	Represents the lowest codec mode (lowest bit-rate) of the ACS.
CODEC_MODE_2	1	Represents the second lowest mode, if the ACS includes more than one mode.
CODEC_MODE_3	2	Represents the third lowest mode, if the ACS includes more than two modes.
CODEC_MODE_4	3	Represents the highest mode, if the ACS includes four modes.

(UWORD8) **amr_configuration.active_codec_set**

Indicates which voice codec will be used. This value is a bit field, if a bit corresponding to a voice codec is equal to 1, this codec is used in this AMR configuration otherwise is not used. The correspondence between the bit number and the voice codec is the following:

Bit number	Vocoder
8 (MSB)	12,2 kbit/s codec rate.
7	10,2 kbit/s codec rate.
6	7,95 kbit/s codec rate.
5	7,40 kbit/s codec rate.
4	6,70 kbit/s codec rate.
3	5,90 kbit/s codec rate.
2	5,15 kbit/s codec rate.
1 (LSB)	4,75 kbit/s codec rate.

(UWORD8) **amr_configuration.threshold[3]** Indicates the AMR lower thresholds values for switching between the voice codec mode j and j-1. The range of each threshold is from 0 to 63 and the number of threshold is from 0 to 3.
(UWORD8) **amr_configuration.hysteresis[3]** Indicates the AMR hysteresis values to obtain the higher thresholds for switching between the voice codec mode j and j+1. The range of each hysteresis is from 0 to 15 and the number of hysteresis is from 1 to 3.

MPHC_ASYNC_HO_CON

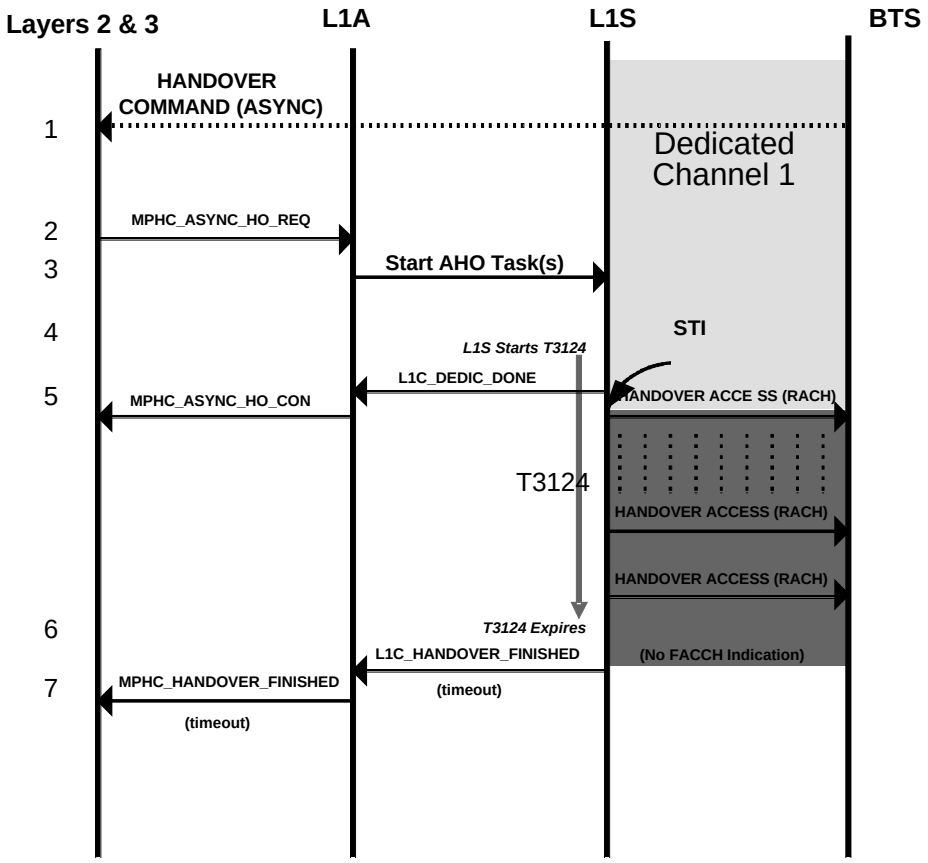
This message is a trigger without parameters.

MPHC_HANDOVER_FINISHED

T_MPHC_HANDOVER_FINISHED message structure:
(UWORD8) **cause** Indication of handover success. 0 - complete, 1 - timeout

17. Asynchronous Handover Procedure (Failure)

Asynchronous Handover Failure sequence



17.1. Prerequisites

Its a prerequisite that the MS is already on some dedicated channel.

17.2. Figure comments

1. The Network commands an asynchronous handover on the downlink main DCCH.
2. L3 posts an MPHC_ASYNC_HO_REQ message, which is received by L1A.
3. L1A starts L1S tasks to perform the handover operation. L1S waits for the starting time (if specified), otherwise it starts the handover immediately. Once the L1S programs the events to be done after STI, it immediately sends L1C_DEDIC_DONE to L1A. In turn L1A creates an MPHC_ASYNC_HO_CON message and posts it to L3. Thus HO confirmation is sent before the STI itself.
4. L1S deactivates any previously active dedicated channels, starts timer T3124 and repeatedly sends the HANDOVER ACCESS message in an access burst on the main DCCH of the new logical channel. L1S also activates normal downlink reception on the new channel and waits for a FACCH indication (PHYSICAL INFORMATION) from the network.
5. If T3124 expires before PHYSICAL INFORMATION is received, L1S deactivates all channels, constructs a L1C_HANDOVER_FINISHED with *cause* 'timeout' and sends it to L1A.
6. L1A constructs MPHC_HANDOVER_FINISHED with *cause* 'timeout' and sends it to L3.

Notes

The connection failure timer T3124 is maintained in L1S.

L1S waits for the reception of PHYSICAL INFORMATION, then directly extracts the timing advance for the new channel.

17.3. Message Structures

MPHC_ASYNC_HO_REQ

Defined previously.

MPHC_ASYNC_HO_CON

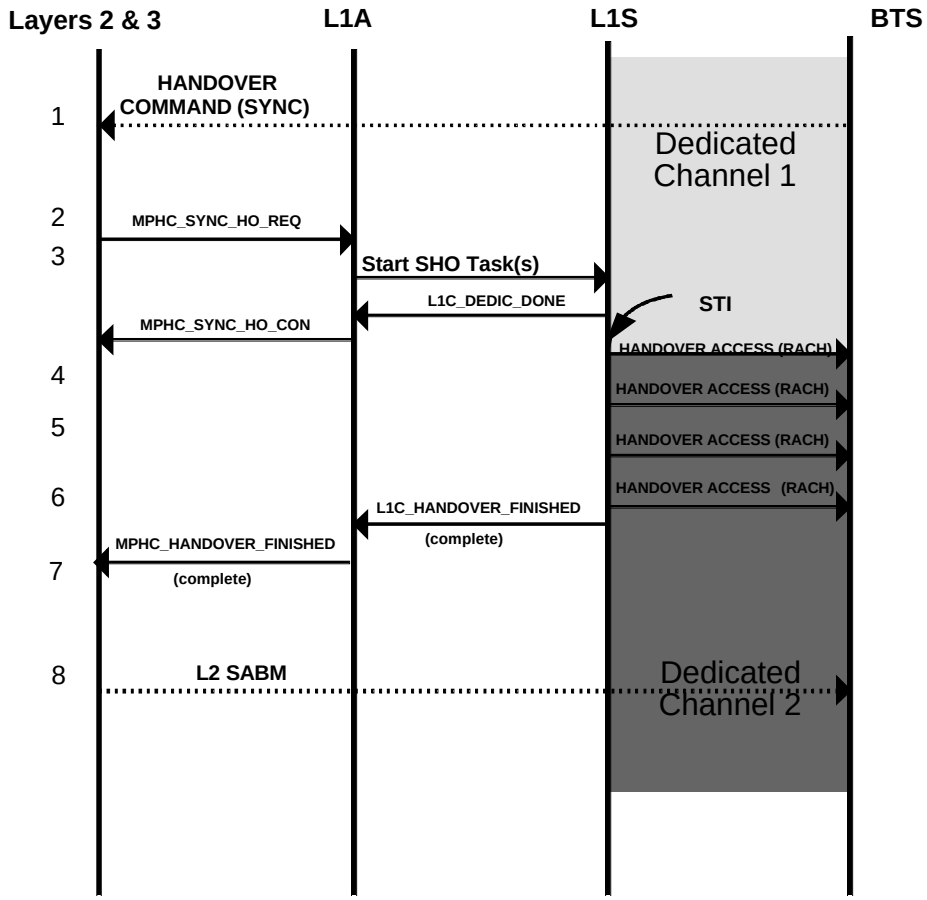
Defined previously.

MPHC_HANDOVER_FINISHED

Defined previously.

18. Synchronous Handover Procedure (Successful)

Synchronous Handover sequence



18.1. Prerequisites

Its a prerequisite that the MS is already on some dedicated channel.

18.2. Figure comments

1. The Network commands a synchronous handover on the downlink main DCCH.
2. L3 posts an MPHC_SYNC_HO_REQ message to L1.
3. L1A starts L1S tasks to perform the handover operation. L1S waits for the starting time (if specified), otherwise it starts the handover immediately. Once the L1S programs the events to be done after STI, it immediately sends L1C_DEDIC_DONE to L1A. In turn L1A creates an MPHC_SYNC_HO_CON message and posts it to L3. Thus HO confirmation is sent before the STI itself.
4. L1S deactivates any previously active dedicated channels, and sends four consecutive HANDOVER ACCESS messages in access bursts on the main DCCH of the new logical channel.
5. After sending four access bursts, L1S activates the new logical channel in uplink. L1S constructs a L1C_HANDOVER_FINISHED with *cause* 'complete' and sends it to L1A.
6. L1A constructs MPHC_HANDOVER_FINISHED and sends it to L3.
7. Layer 2 establishment can be started on the new channel.

NOTE:

In case of AMR feature:

- All the abnormal case (c.f. 4.18 [1] 3.4.3.3, 3.4.6.1.3) and concerning the handover messages must be managed directly by the protocol stack. It means that the protocol stack checks the validity of the multirate parameters before to send it to the Layer1. In case of theses parameters are incorrect, the protocol stack doesn't send this new AMR configuration to the L1.
- If a handover message is received by the protocol stack and only the multirate channel mode is specified (c.f. 4.18 [1], 3.4.4.1, 9.1.15.13), the previous multirate configuration must be sent again to the L1 with the handover message.
- The protocol stack must fill the all the multirate configuration parameters only if the channel mode is TCH/AFS or TCH/AHS.
- The AMR parameters are valid and exist only if channel_mode = TCH/AHS or TCH/AFS and if the DSP code support the AMR (code v34xx)).

18.3. Message Structures

MPHC_SYNC_HO_REQ

T_MPHC_SYNC_HO_REQ message structure (delta specification) :
All fields as per MPHC_ASYNC_HO_REQ, plus:
(BOOL) **nci** Normal Cell Indication 0 = Out of range TA is ignored. 1 = Out of range TA shall trigger a handover faillure procedure.

Note: **nci** is placed before T_AMR_CONFIGURATION in the structure

MPHC_SYNC_HO_CON

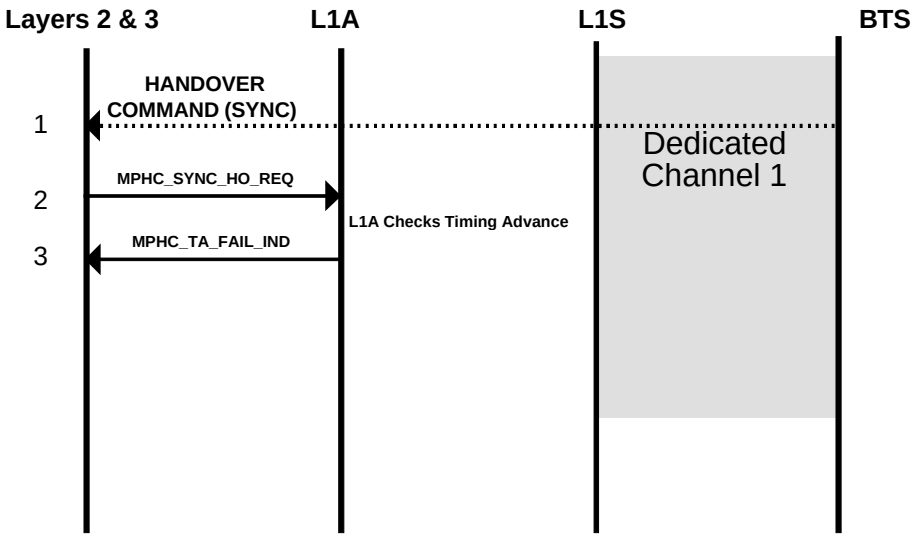
This message is a trigger without parameters.

MPHC_HANDOVER_FINISHED

Defined previously.

19. Synchronous Handover Procedure (Failure - TA out of range)

Synchronous Handover Failure sequence



19.1. Prerequisites

Its a prerequisite that the MS is already on some dedicated channel.

19.2. Figure comments

1. The Network commands a synchronous handover on the downlink main DCCH.
2. L3 posts an MPHC_SYNC_HO_REQ message, which is received by L1A.
3. L1A detects the timing advance is out of range (and that the handover cell does not support out of range timing advance). L1A sends MPHC_TA_FAIL_IND to L3.

Notes

The TA calculation is made immediately upon receiving the handover request. In

the unlikely event that the real TA may change while waiting for starting time L1 has no way of receiving the updated TA information - therefore there is no benefit in waiting for starting time before posting MPHC_TA_FAIL_IND. In this failure case L3 does not initiate Return Handover.

19.3. Message Structures

MPHC_SYNC_HO_REQ

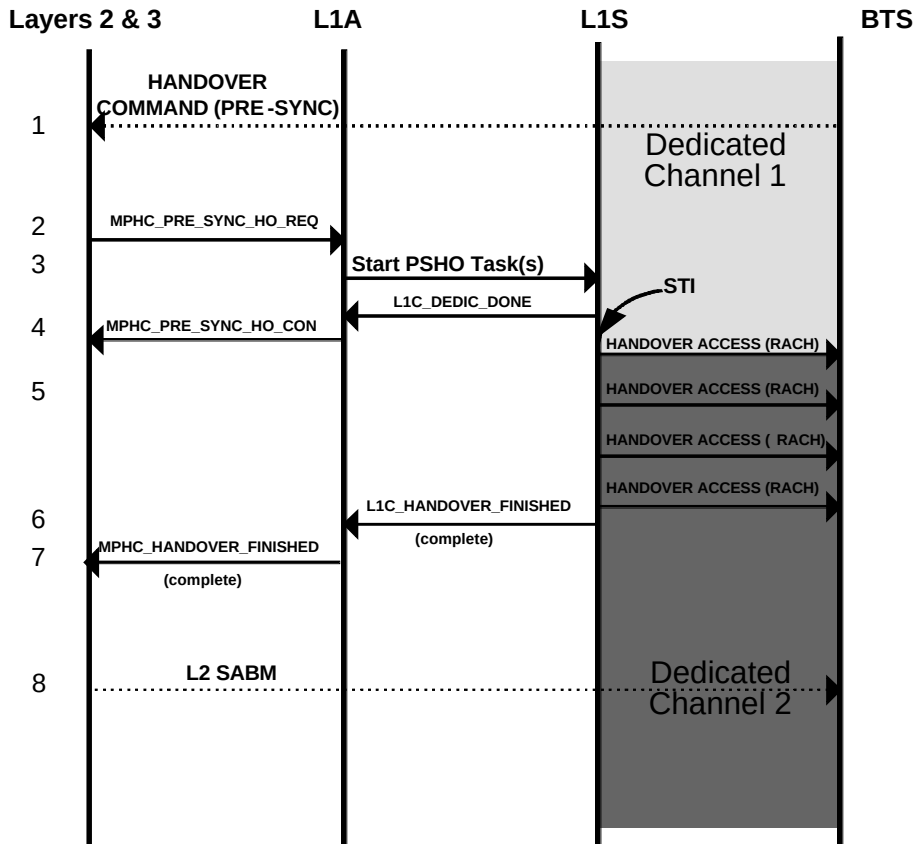
Defined previously.

MPHC_TA_FAIL_IND

This message is a trigger without parameters.

20. Pre-Synchronous Handover Procedure (Successful)

Pre-Synchronous Handover sequence



20.1.rerequisites

Its a prerequisite that the MS is already on some dedicated channel.

20.2. Figure comments

1. The Network commands a pre-synchronous handover on the downlink main DCCH.
2. L3 posts an MPHC_PRE_SYNC_HO_REQ message, which is received by L1A.
3. L1A starts L1S tasks to perform the handover operation. L1S waits for the starting time (if specified), otherwise it starts the handover immediately. Once the L1S programs the events to be done after STI, it immediately sends L1C_DEDIC_DONE to L1A. In turn L1A creates an MPHC_PRE_SYNC_HO_CON message and posts it to L3. Thus HO confirmation is sent before the STI itself.
4. L1S deactivates any previously active dedicated channels, and sends four consecutive HANDOVER ACCESS messages in access bursts on the main DCCH of the new logical channel.
5. After sending four access bursts, L1S activates the new logical channel in uplink. L1S constructs a L1C_HANDOVER_FINISHED with *cause* 'complete' and sends it to L1A.
6. L1A constructs MPHC_HANDOVER_FINISHED with *cause* 'complete' and sends it to L3.
7. Layer 2 establishment can be started on the new channel.

NOTE:

In case of AMR feature:

- All the abnormal case (c.f. 4.18 [1] 3.4.3.3, 3.4.6.1.3) and concerning the handover messages must be managed directly by the protocol stack. It means that the protocol stack checks the validity of the multirate parameters before to send it to the Layer1. In case of theses parameters are incorrect, the protocol stack doesn't send this new AMR configuration to the L1.
- If a handover message is received by the protocol stack and only the multirate channel mode is specified (c.f. 4.18 [1], 3.4.4.1, 9.1.15.13), the previous multirate configuration must be sent again to the L1 with the handover message.
- The protocol stack must fill the all the multirate configuration parameters only if the channel mode is TCH/AFS or TCH/AHS.
- The AMR parameters are valid and exist only if channel_mode = TCH/AHS or TCH/AFS and if the DSP code support the AMR (code v34xx)).

20.3. Message Structures

MPHC_PRE_SYNC_HO_REQ

T_MPHC_PRE_SYNC_HO_REQ message structure (delta specification) :
All fields as per MPHC_SYNC_HO_REQ, plus:
(BOOL) **timing_advance_valid** 0 = TA is not valid. 1 = TA is valid.
(UWORD8) **timing_advance** Specify the timing advance value. Range: [0…63] bits

Note: **timing_advance_valid** and **timing_advance** are placed before T_AMR_CONFIGURATION in the structure

MPHC_PRE_SYNC_HO_CON

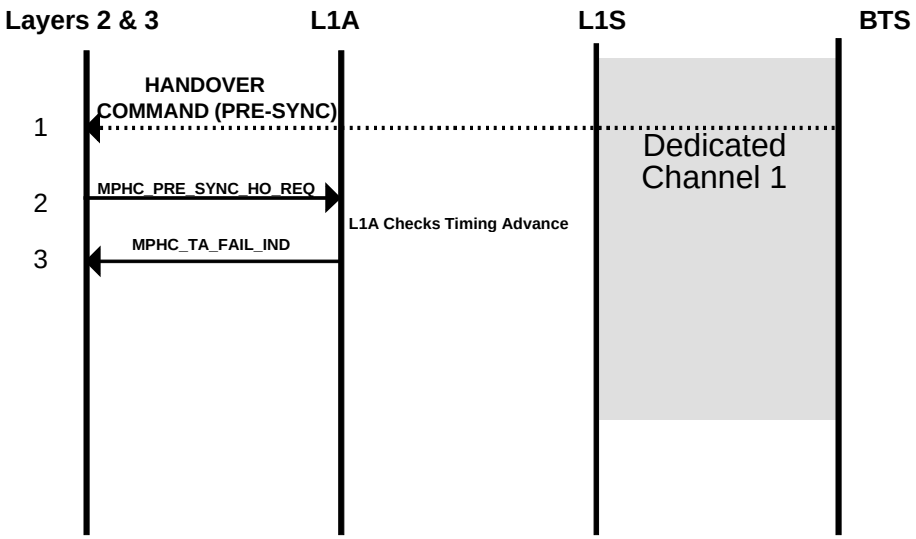
This message is a trigger without parameters.

MPHC_HANDOVER_FINISHED

Defined previously.

21. Pre-Synchronous Handover Procedure (Failure - TA out of range)

Pre-Synchronous Handover Failure sequence



Notes

The notes regarding immediate posting of MPHC_TA_FAIL_IND for Synchronous Handover applies for Pre-Synchronous Handover. In this failure case L3 does not initiate Return Handover.

21.1. Prerequisites

Its a prerequisite that the MS is already on some dedicated channel.

21.2. Figure comments

1. The Network commands a synchronous handover on the downlink main DCCH.
2. L3 posts an MPHC_PRE_SYNC_HO_REQ message, which is received by L1A.

3. L1A detects the timing advance is out of range (and that the handover cell does not support out of range timing advance). L1A sends MPHC_TA_FAIL_IND to L3.

21.3. Message Structures

MPHC_PRE_SYNC_HO_REQ

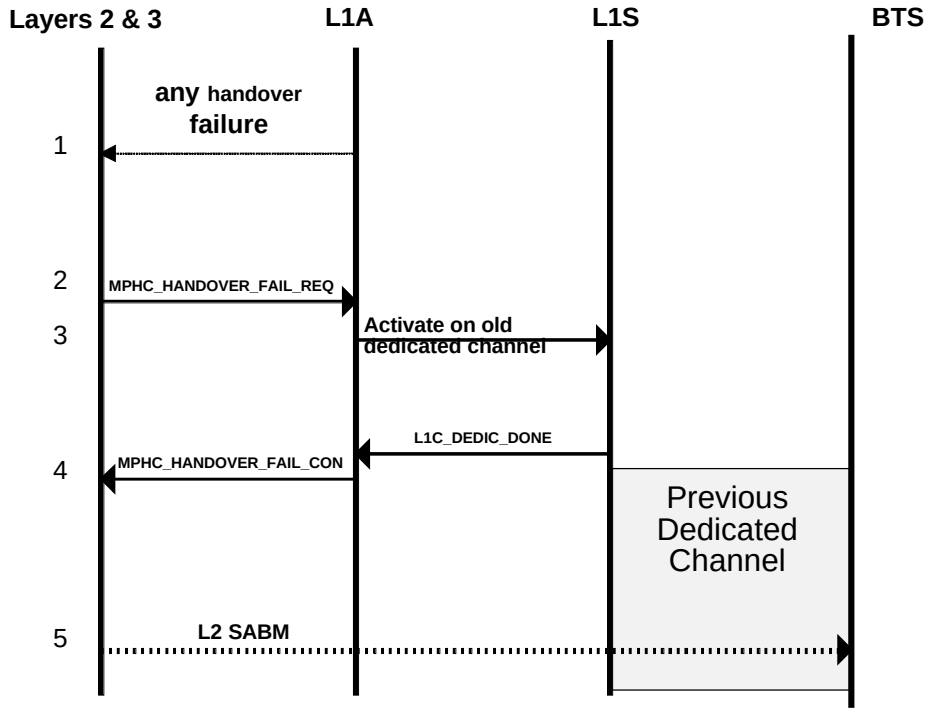
Defined previously.

MPHC_TA_FAIL_IND

This message is a trigger without parameters.

22. Handover Failure Recovery (all cases)

Handover Failure Recovery sequence



22.1. Prerequisites

Its a prerequisite that the MS is already on some dedicated channel.

22.2. Figure comments

1. A handover failure results in a Handover failure message from from L1, or a failure being detected in L2.
2. L3 posts MPHC_HANDOVER_FAIL_REQ to L1A, which requests L1 to reactive the old dedicated channel.
3. L1 switches the active and free dedicated sets and reactivates the old channel.

4. When the previous channel reactivation is programmed L1S posts L1C_DEDIC_DONE to L1A. L1A posts MPHC_HANDOVER_FAIL_CON to L3.
5. Layer 2 reconnection can be started on the new (old) channel.

22.3. Message Structures

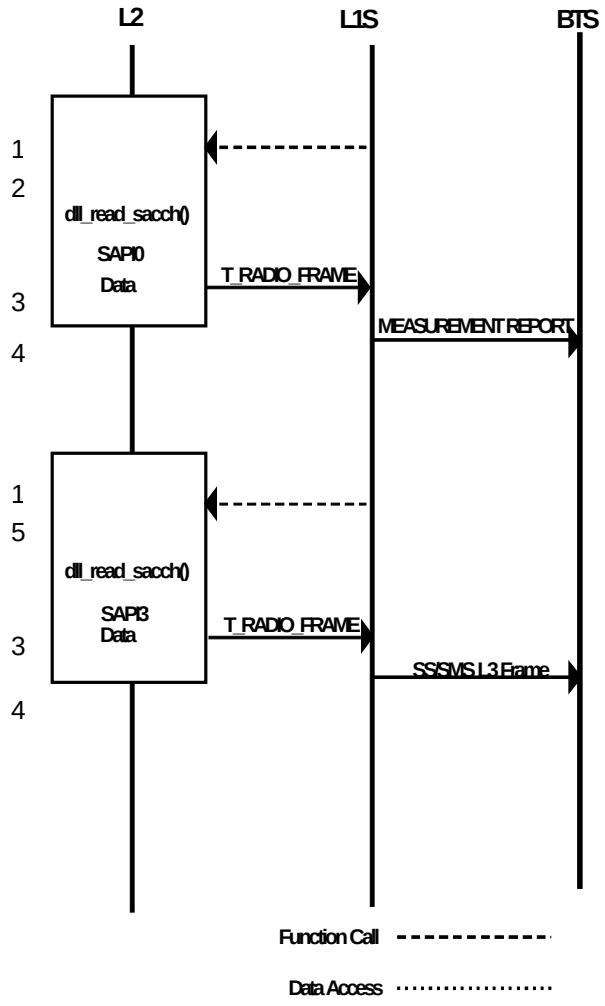
MPHC_HANDOVER_FAIL_REQ

This message is a trigger without parameters.

23. SACCH Uplink Interface

The SACCH uplink procedure (on all associated DCCHs) is depicted below. L2/L1 interface is represented by a function ***dll_read_sacch()*** which must be implemented in the L2. Two cases are shown below; SAPI 0 data transfer (applicable to SDCCH and TCH contexts) and SAPI 3 data transfer (TCH only). ***dll_read_sacch()*** must perform the SAPI0/SAPI3 multiplexing in the case of both services being required at the same time on TCH.

SACCH uplink sequence



Notes:

dll_read_sacch() seamlessly arbitrates between SAPI 0 and SAPI 3 data according to the rules described in GSM Ph2+ 04.05.

If SAPI 0 data is unavailable L1 is to supply the dummy SACCH data with the invalid flag set within it. This is to be done within L2.

send_l2_sacch_frame() returns a pointer to a 21 UWORD8 data block which is owned by L2 and

No parameter needs to be passed to *dll_read_sacch()* as presently this function is always called in SIG_ONLY_MODE. However in the current implementation, ***channel_mode*** is passed as parameter.

23.1. Prerequisites

Its a prerequisite that the MS is already on some dedicated channel.

23.2. Figure comments

1. In order to collect SACCH uplink data, L1S calls *dll_read_sacch()* in L2.
2. *dll_read_sacch()* examines the prepared data structures in the L2, and finds SAPI 0 data ready for transmission.
3. *dll_read_sacch* maps the SACCH data into a T_RADIO_FRAME structure and returns a pointer to it.
4. L1S appends the layer 1 header and transmits the SACCH block.
5. *dll_read_sacch()* examines the prepared data structures in the L2, and finds SAPI 3 data ready for transmission.

23.3. L2 Function prototype

The L2 function will comply with the following prototype,

```
T_RADIO_FRAME   *dll_read_sacch (UWORD8 chn_mode);
```

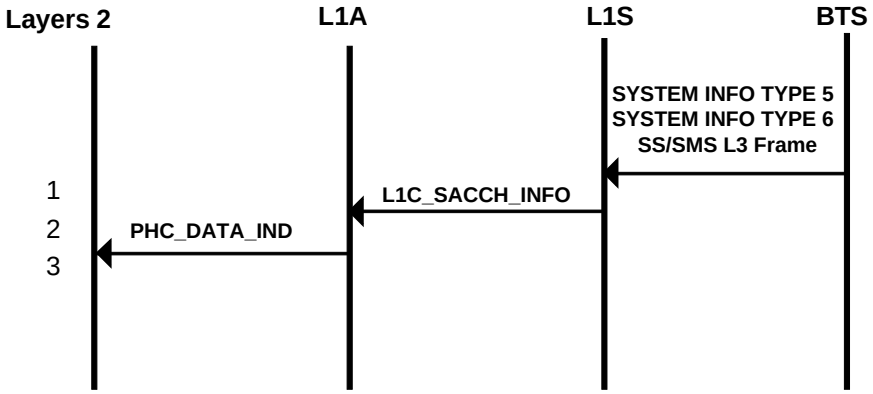
with

```
typedef struct
{
  UWORD8 A[22+1];
}
T_RADIO_FRAME;
```

24. SACCH Downlink Interface

The SACCH downlink procedure (on all associated DCCHs) is depicted below. Two cases are shown below; SAPI 0 data transfer (applicable to SDCCH and TCH contexts) and SAPI 3 data transfer (TCH only).

SACCH downlink sequence



Notes

L1C_SACCH_INFO should be posted whenever there is a downlink SACCH data opportunity, whatever the dedicated channel type, even if there is no data received or a Bad Frame Indication. L1 does not maintain the S counter, this is achieved in L3.

While only 21 UWORD8s of the SACCH block are used (UWORD8 1 and 2 are the L1 header) it is expected that the full SACCH block will be present in the PH_DATA_IND structure. There is limited value in adjusting L1S to only post 21.

24.1. Prerequisites

Its a prerequisite that the MS is already on some dedicated channel.

24.2. Figure comments

1. Incident SACCH frames arrive at L1S.
2. L1S sends L1C_SACCH_INFO indication to L1A.
3. L1A prepares a PH_DATA_IND message and sends it to L2.

24.3. Message Structures

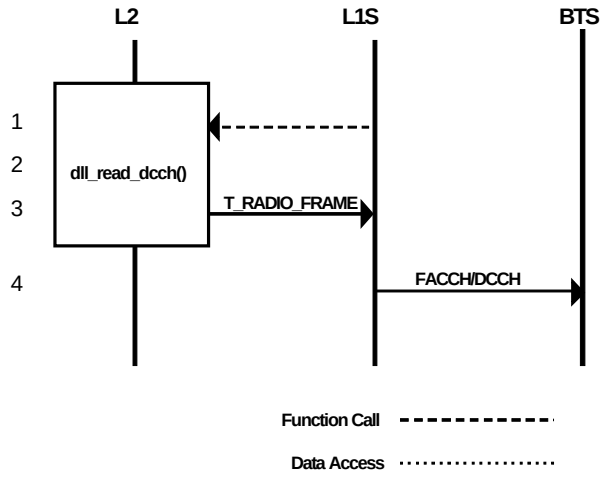
PH_DATA_IND

T_PH_DATA_IND message structure		
UWORD16	**rf_chan_num**	
Channel (radio_freq) that block was read on.		
UWORD8	**l2_channel_type**	
Indicates the MF51 block type read (enum type), maps to l1s msg signal code:		
1	(L2_CHANNEL_SACCH)	indicates SACCH block
2	(L2_CHANNEL_SDCCH)	indicates SDCCH block
3	(L2_CHANNEL_FACCH_H)	indicates half rate FACCH
4	(L2_CHANNEL_FACCH_F)	indicates full rate FACCH
5	(L2_CHANNEL_CCCH)	indicates CCCH block
6	(L2_CHANNEL_NBCCH)	indicates normal BCCH
7	(L2_CHANNEL_PCH)	indicates PCH block
8	(L2_CHANNEL_EPCH)	indicates extended PCH block
9	(L2_CHANNEL_CBCH)	indicates CBCH block
10	(L2_CHANNEL_EBCCH)	indicates extended BCCH block
UWORD8	**error_cause**	
Indicates if l2_frame data is invalid: !0 == invalid block info read 0 == valid block info read		
UWORD8	**l2_frame.A[0]**	
Layer 1 header, ordered MS power level. Valid if error_cause == 0 [GSM 04.04 section 7]		
UWORD8	**l2_frame.A[1]**	
Layer 1 header, Timing Advance command. Valid if error_cause == 0 [GSM 04.04 section 7]		
UWORD8	**l2_frame.A[2]..[22]**	
Decoded data (21 bytes). Valid if error_cause == 0 [GSM 04.04 section 7]		
UWORD8	**bsic**	
Base Station Identification Code [GSM 05.08 ver 4.15.0 section 9]		
UWORD8	**tc**	
tc = (FN div 51) % 8 - i.e: indicates which MF51 the block belongs to (e.g: for BCCH sys info).		

25. FACCH/DCCH Uplink Interface

The FACCH/DCCH uplink procedure is depicted below. L2/L1 interface is represented by a function ***dll_read_dcch()*** which must be implemented in the L2.

FACCH/DCCH uplink sequence



Notes:

dll_read_dcch() is L1S's interface for both uplink DCCH and FACCH (hereafter collectively termed DCCH).

dll_read_dcch() takes a parameter *chan_mode*. If this is set to SIG_ONLY mode, then layer 2 fill frames will be returned if no data is waiting to be transmitted (i.e for SDCCH and TCH/SigOnly). Otherwise, if no data is waiting, NULL_PTR will be returned (for TCH/F when no FACCH signalling is required).

dll_read_dcch() should be called whenever there is an uplink data opportunity, whatever the dedicated channel type (i.e. before SDCCH blocks and FACCH interleaving boundaries).

25.1. Prerequisites

Its a prerequisite that the MS is already on some dedicated channel.

25.2. Figure comments

1. In order to collect DCCH uplink data for sending, L1S calls *dll_read_dcch()* in L2.
2. Either data, a layer 2 fill frame, or NULL_PTR is returned, depending on the conditions described above in the notes.
3. *dll_read_dcch* maps the DCCH data (if available) into a T_RADIO_FRAME structure and returns a pointer (either data or NULL_PTR).
4. If applicable, L1S transmits the DCCH block according to the current channel type.

25.3. L2 Function prototype

The L2 function will comply with the following prototype,

```
T_RADIO_FRAME   *dll_read_dcch(UWORD8 chn_mode);
```

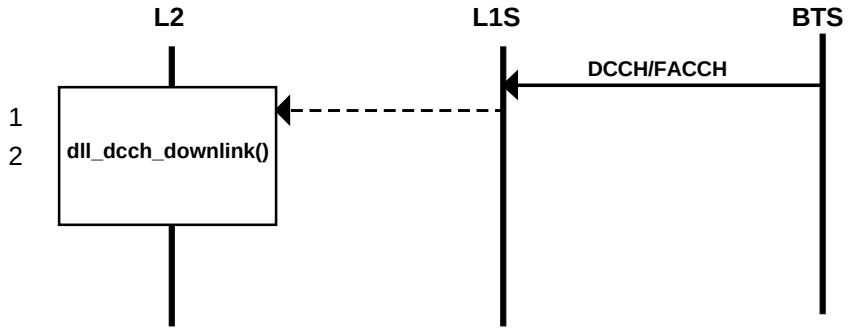
with

```
typedef struct
{
  UWORD8 A[22+1];
}
T_RADIO_FRAME;
```

26. FACCH/DCCH Downlink Interface

The FACCH downlink procedure (on all associated DCCHs) is depicted below. L2/L1 interface is represented by a function ***dll_dcch_downlink()*** which must be implemented in the L2.

FACCH/DCCH downlink sequence



Notes:

The above describes the interface for both downlink DCCH and FACCH (hereafter collectively termed DCCH).

dll_dcch_downlink(data_ptr, valid_flag) should be called whenever there is a downlink data opportunity, whatever the dedicated channel type (i.e. after SDCCH blocks and FACCH deinterleaving boundaries), even if there is no data received or a Bad Frame Indication.

2 parameters are passed with via *dll_dcch_downlink(data_ptr, valid_flag)* to the L2 function. To avoid unnecessary copying of the DCCH data the *data_ptr* parameter points directly to the DSP memory. It is up to L1 to clear the DSP memory after a read.

If no data is available L1 calls *dll_dcch_downlink(NULL, 0)*.

26.1. Prerequisites

Its a prerequisite that the MS is already on some dedicated channel.

26.2. Figure comments

1. Incident DCCH frames arrive at L1S.
2. L1S calls *dll_dcch_downlink()* in L2, with a pointer to the received frame (or NULL_PTR if no data or BFI).

26.3. L2 Function prototype

The L2 function will comply with the following prototype,

*void dll_dcch_downlink (API*info_address, UWORD8 valid_flag);*

with

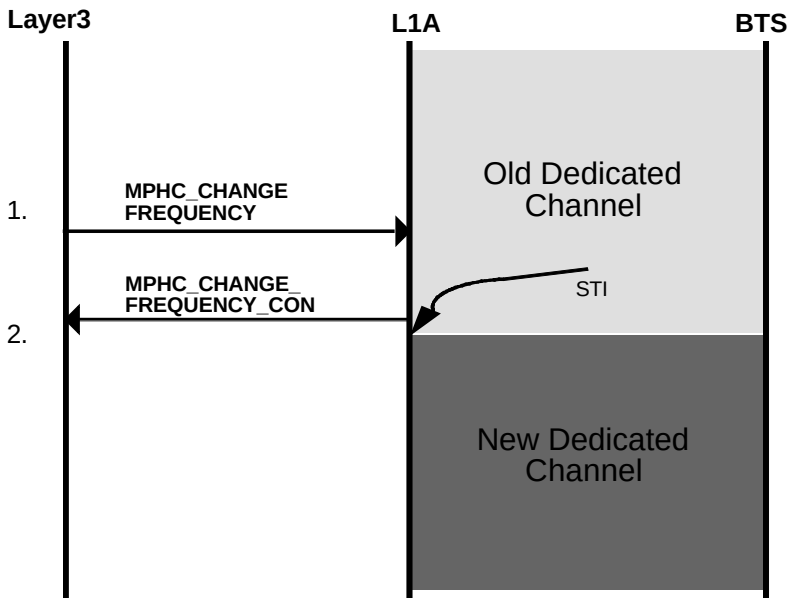
```
#if CHIPSET == 0
        typedef UWORD32  API;
#endif

#if CHIPSET == 2
        typedef UWORD16  API;
#endif
```

A new optional interface, where the current L1 frame number is passed in this function call to the protocol stack is also available via the compilation option SEND_FN_TO_L2_IN_DCCH. The new prototype with this option is:

*void dll_dcch_downlink (API*info_address, UWORD8 valid_flag, UWORD32 frame_number);*

27. Dedicated Frequency Redefinition Interface

Frequency Redefinition sequence



27.1. Prerequisites

Its a prerequisite that the MS is already on some dedicated channel.

27.2. Figure comments

1. L3 posts the MPHC_CHANGE_FREQUENCY to start the redefinition.
2. The old dedicated channel is maintained until the starting time is reached.The MPHC_CHANGE_FREQUENCY_CON message is posted when the new dedicated channel is programmed by L1s.

Notes

Initially it was thought that there would be no difficulties if L1 posted the confirm message before the starting time was reached. This is because the confirm is used by L3 to determine the appropriate frequency for the MS's dedicated

channel in the event of a failed handover attempt occurring between the CHANGE_FREQ_REQ and the starting time. The method that L1 uses to store the before and after time data for a dedicated channel ensures that the correct frequency will be returned to in all cases. However, the above solution does not account for a failed assignment at the L2 level, therefore the solution is to post the confirm message immediately after the starting time.

27.3. Message Structures

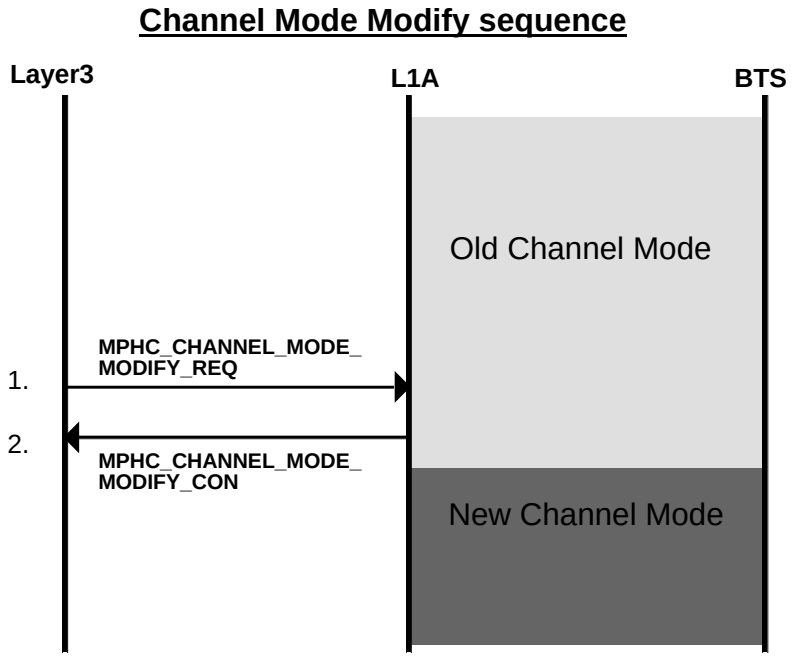
MPHC_CHANGE_FREQUENCY

T_ MPHC_CHANGE_FREQUENCY message structure:		
T_CHANNEL_DESCRIPTION **channel_desc**		
	(BOOL) **channel_desc.chan_sel.h** Hopping or Static RF channel (0 = Static, 1 = Hopping).	
	(union)	(UWORD16) **channel_desc.chan_sel.rf_channel.single_rf.radio_freq** Absolute Radio Frequency Channel Number (0 to 124, 975 to 1023)
		(UWORD8) **channel_desc.chan_sel.rf_channel.hopping_rf.maio** Mobile Allocation Index Offset
		(UWORD8) **channel_desc.chan_sel.rf_channel.hopping_rf.hsn** Hopping Sequence Number
	(UWORD8) **channel_desc.channel_type** 0 = Invalid Channel, 1 = TCH/F, 2 = TCH/H, 3 = SDCCH/4, 4 = SDCCH/8.	
	(UWORD8) **channel_desc.subchannel** Subchannel for SDCCH/4 (0-3), SDCCH/8 (0-7) and TCH/H (0,1)	
	(UWORD8) **channel_desc.timeslot_no** Timeslot on which to configure the new channel (0-7)	
	(UWORD8) **channel_desc.tsc** Training Sequence Code to select transmit midamble	
T_MOBILE_ALLOCATION **frequency_list**		
	(UWORD16) **frequency_list.rf_chan_cnt** Number of channels in MA list	
	(UWORD16) **frequency_list.rf_chan_no.A[64]** MA List	
T_STARTING_TIME **starting_time**		
	(BOOL) **starting_time.start_time_present** Indicates if a starting time is specified for the new channel	
	(UWORD8) **starting_time.start_time.n32** Start frame number (T1' component)	
	(UWORD8) **starting_time.start_time.n51** Start frame number (T3 component)	
	(UWORD8) **starting_time.start_time.n26** Start frame number (T2 component)	

MPHC_CHANGE_FREQUENCY_CON

This message is a trigger without parameters.

28. Dedicated Channel Mode Modify Interface



28.1. Prerequisites

Its a prerequisite that the MS is already on some dedicated channel.

28.2. Figure comments

1. L3 posts the MPHC_CHANNEL_MODE_MODIFY_REQ to start the channel mode modify.
2. Immediately on receiving the request the channel mode is modified and the confirm message MPHC_CHANNEL_MODE_MODIFY_CON is sent to L3

NOTE:

In case of AMR feature:

- All the abnormal case (c.f. 4.18 [1] 3.4.3.3, 3.4.6.1.3) and concerning the mode modify messages must be managed directly by the protocol stack. It means that the protocol stack checks the

validity of the multirate parameters before to send it to the Layer1. In case of theses parameters are incorrect, the protocol stack doesn't send this new AMR configuration to the L1.

- The protocol stack must fill the all the multirate configuration parameters only if the channel mode is TCH/AFS or TCH/AHS.
- The AMR parameters are valid and exist only if channel_mode = TCH/AHS or TCH/AFS and if the DSP code support the AMR (code v34xx)).

28.3. Message Structures

MPHC_CHANNEL_MODE_MODIFY_REQ

T_ MPHC_CHANNEL_MODE_MODIFY_REQ message structure:

(UWORD8) **subchannel**

0 = mode change is for subchannel 0.

1 = mode change is for subchannel 1.

(UWORD8) **channel_mode**

0 = Signalling Only, 1 = TCH/FS, 2 = TCH/HS, 3 = TCH/96, 4 = TCH/F48, 5 = TCH/H48,
6 = TCH/F24, 7 = TCH/H24, 8 = TCH/EFR, 9 = TCH/144, 10 = TCH/AHS, 11 = TCH/AFS.

T_AMR_CONFIGURATION **amr_configuration (these following parameters are valid and exist only if channel_mode = TCH/AHS or TCH/AFS and if the DSP code support the AMR (code v34xx)).**

(BOOL) **amr_configuration.noise_suppression_bit**

For instance the role of this flag is unknown but the possible values are:

Value	Legend
FALSE	Noise Suppression can be used (default)
TRUE	Noise Suppression shall be turned off

(BOOL) **amr_configuration.initial_codec_mode_indicator**

Indicates the rule to choose the initial codec mode.

initial_codec_mode_indicator = FALSE: the initial codec mode is defined by the implicit rule provided in GSM 05.09 (3.4.3).

initial_codec_mode_indicator = TRUE: the initial codec mode is defined by the Start Mode field (i.e. value initial_codec_mode).

Value	Legend
FALSE	The initial codec mode is defined by the implicit rule provided in GSM 05.09 (3.4.3).
TRUE	The initial codec mode is defined by the Start Mode field (i.e. value initial_codec_mode).

(UWORD8) **amr_configuration.initial_codec_mode**

Indicates which codec mode are used in the initialise phase in case of *initial_codec_mode_indicator = TRUE*. The possible value are defined in the GSM5.09 section 3.4.1 and 5.03 section 3.10:

Identifier	Value	Legend
CODEC_MODE_1	0	Represents the lowest codec mode (lowest bit-rate) of the ACS.

CODEC_MODE_2	1	Represents the second lowest mode, if the ACS includes more than one mode.
CODEC_MODE_3	2	Represents the third lowest mode, if the ACS includes more than two modes.
CODEC_MODE_4	3	Represents the highest mode, if the ACS includes four modes.

(UWORD8) **amr_configuration.active_codec_set**

Indicates which voice codec will be used. This value is a bit field, if a bit corresponding to a voice codec is equal to 1, this codec is used in this AMR configuration otherwise is not used. The correspondence between the bit number and the voice codec is the following:

Bit number	Vocoder
8 (MSB)	12,2 kbit/s codec rate.
7	10,2 kbit/s codec rate.
6	7,95 kbit/s codec rate.
5	7,40 kbit/s codec rate.
4	6,70 kbit/s codec rate.
3	5,90 kbit/s codec rate.
2	5,15 kbit/s codec rate.
1 (LSB)	4,75 kbit/s codec rate.

(UWORD8) **amr_configuration.threshold[3]**

Indicates the AMR lower thresholds values for switching between the voice codec mode j and j-1. The range of each threshold is from 0 to 63 and the number of threshold is from 0 to 3.

(UWORD8) **amr_configuration.hysteresis[3]**

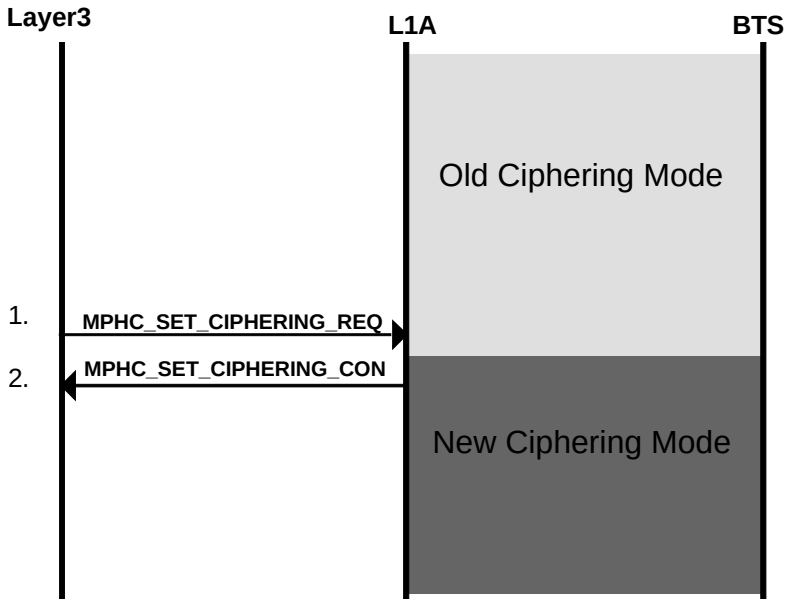
Indicates the AMR hysteresis values to obtain the higher thresholds for switching between the voice codec mode j and j+1. The range of each hysteresis is from 0 to 15 and the number of hysteresis is from 1 to 3.

MPHC_CHANNEL_MODE_MODIFY_CON

This message is a trigger without parameters.

29. Dedicated Set Ciphering Interface

Set Ciphering sequence



29.1. Prerequisites

Its a prerequisite that the MS is already on some dedicated channel.

29.2. Figure comments

1. L3 posts the MPHC_SET_CIPHERING_REQ to start the ciphering on current channel.
2. Immediately on receiving the request the ciphering mode is modified and the confirm message MPHC_SET_CIPHERING_CON is sent to L3.

29.3. Message Structures

MPHC_SET_CIPHERING_REQ

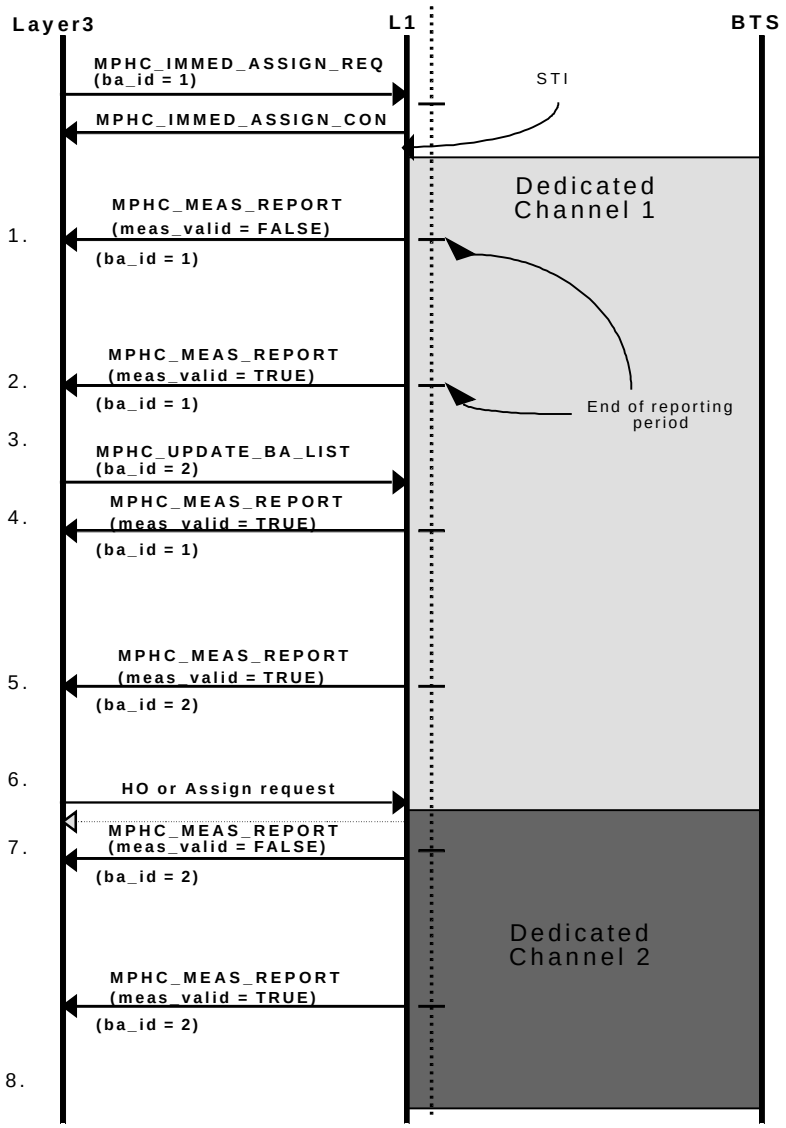
T_ MPHC_SET_CIPHERING_REQ message structure:
(UWORD8) **cyper_mode** 0 = No Cyphering, !0 = Cyphering ON.
(UWORD8) **a5_algorithm** 0 = A5/1, 1 = A5/2, 2 = A5/3, 3 = A5/4, 4 = A5/5, 5 = A5/6, 6 = A5/7
(UWORD8) **new_ciph_param.A[8]** ciphering key.

MPHC_SET_CIPHERING_CON

This message is a trigger without parameters.

30. Dedicated Power Measurement Interface

Power Measurement sequence



30.1. Prerequisites

Its a prerequisite that the MS is already on some dedicated channel.

30.2. Figure comments

1. While on a Dedicated channel L1 is reporting the power measurements of the neighbour cells every 104 frames (102 for SDCCH channel). At the end of the reporting period MPHC_MEAS_REPORT is posted to L3 with a flag to indicate whether the data is valid i.e., whether a full set of measurement data was obtained. The first report after the immediate assign will be invalid.
2. The second MPHC_MEAS_REPORT message after an immediate assignment should be valid, if it is the *meas_valid* flag will be set to TRUE.
3. If the BA list is to be updated MPHC_UPDATE_BA_LIST which contains the full BA list (33 cells) and associated parameters is posted to L1. The measurement process will download the new BA list at the end of the reporting period (The immediate BA list download mechanism in dedicated mode is now obsolete).
4. The MPHC_MEAS_REPORT message after an update BA list instruction should be valid, if it is the *meas_valid* flag will be set to TRUE. (The Immediate BA list download mechanism in dedicated mode is now obsolete).
5. After any handover request or a channel assign the measurement report will be interrupted. Therefore the *meas_valid* flag is set to FALSE. nb. In the diagram the confirm messages have been only lightly drawn in.
6. The first MPHC_MEAS_REPORT message after a handover or channel assign will be invalid.
7. The second MPHC_MEAS_REPORT message after a handover or channel assign should be valid.

Notes1

It is within the GSM specs to return an invalid report if a handover or assignment has been actioned. It is unclear what is required if the BA list is updated. The approach adopted is to keep measuring on the old BA list until the end of the reporting period. And start the new reporting period with the new BA list. (The immediate BA list download Mechanism in dedicated mode is now Obsolete).

If a MPHC_STOP_DEDICATED(_REQ) is received and there is a pending new list not yet downloaded, the new BA list is downloaded before going to Idle.

If a MPHC_CHANNEL_RELEASE is received the process is aborted and no result message is posted.

Notes2

For Automatic Gain Control Algorithm reason, the Serving Cell Beacon radio

frequency value must be always included in the BA list (cf. Problem Report PB512.doc TI_22)

30.3. Message Structures

MPHC_MEAS_REPORT

T_MPHC_MEAS_REPORT message structure:	
(BOOL) **dtx_used** Uplink DTX has been used on this measurement period. TRUE for used, FALSE for not.	
(BOOL) **meas_valid** The measurements are valid, i.e., a full set of measurements were made.	
(WORD16) **rxlev_full_acc** Accumulated RX power level over all serving cell frames. Compensated (i.e., for gain, freq, temp) but not averaged or limited.	
(UWORD8) **rxlev_full_nbr_meas** Number of rxlev_full measurements taken on serving cell.	
(WORD16) **rxlev_sub_acc** Accumulated RX power level over serving cell frames that will always be received even in DTX mode. Compensated (i.e., for gain, freq, temp) but not averaged or limited.	
(UWORD8) **rxlev_sub_nbr_meas** Number of rxlev_sub measurements taken on serving cell.	
(UWORD16) **rxqual_full_acc_errors** Number of bit errors over all serving cell frames in the reporting period.	
(UWORD16) **rxqual_full_nbr_bits** Number of bits examined for errors on serving cell.	
(UWORD16) **rxqual_sub_acc_errors** Number of bit errors over serving cell frames that will always be received even in DTX mode.	
(UWORD16) **rxqual_sub_nbr_bits** Number of bits examined for errors over serving cell frames that will always be received even in DTX mode.	
(UWORD8) **no_of_ncell_meas** Number of neighbour cells that have had measurements made on them.	
T5_NCELL_MEAS **ncell_meas**	
	(UWORD16) **ncell_meas.A[33].bcch_freq** Neighbour cell carrier frequency (radio_freq).
	(WORD16) **ncell_meas.A[33].rxlev_acc** Accumulated RX power level on the neighbour cell. Compensating but not averaged or limited.
	(UWORD8) **ncell_meas.A[33].rxlev_nbr_meas** Number of measurements made on the neighbour cell.

(UWORD8) **ba_id** Message sequence number of this request message. Value range 0 to 255.
(UWORD8) **timing_advance** Timing advance value currently used by L1. For TEST MOBILE only. Range: [0…63] bits
(UWORD8) **txpwr_used** TX power level currently used by L1. For TEST MOBILE only.
RESERVED FOR TRACE / DEBUG
(UWORD8) **facch_dl_count**
(UWORD8) **facch_ul_count**

MPHC_UPDATE_BA_LIST

T_MPHC_UPDATE_BA_LIST message structure:
(UWORD8) **num_of_chans** Number of valid carrier numbers contained in the BA list (chan_number) array (1 to 33)
(UWORD16) **chan_list.A[32+1]** An array containing a BA list of up to 33 carrier **radio_freqs** (nb. this is 1 more than is required as the serving cell may only be included in the list if it is requested as one of the 32 neighbour cells, unlike Idle mode which will always have the serving cell in its BA list).
(BOOL) **pwrc** Power control indicator.
(BOOL) **dtx_allowed** Indicates whether the MS will use uplink DTX or not (0 - MS must not use DTX, 1 - MS must use DTX).
(UWORD8) **ba_id** Message sequence number of this request message. Value range 0 to 255.

Notes:

L3 expects the *rx_lev* and *rx_qual* measurements to contain both the accumulated value and the number of measurements made. The averaging process is done in L3.

In Idle mode the BA list is constructed with 33 carrier radio_freqs. Previously in Dedicated there was only the facility to describe (in MPHC_UPDATE_BA_LIST) and receive measurements (in MPHC_MEAS_REPORT) from 32 neighbour cells - this was updated to match IDLE in the way that is shown above. Subsequently it was decided that the serving cell would only ever be part of the BA list if specifically requested by the network - therefore the 32 carrier radio_freq was sufficient.

31. Stop Dedicated Mode

Any dedicated mode can be stopped by L3 sending a stop message.
This stop message is the same for any dedicated channel (SDCCH or TCH).

Actually because of a rework in L1 and protocol stack, there are 2 different interfaces to stop the dedicated mode.

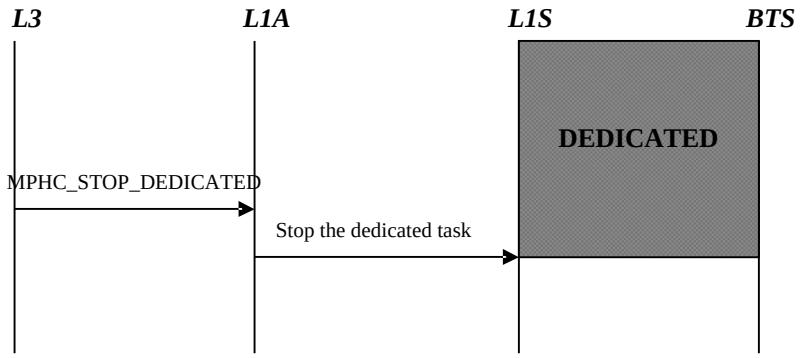
- The first one is used in old L1 code up to releases TCS_2.1.1_L1_1444_0 and TCS_3.1_L1_1442_0.

L3->L1	**MPHC_STOP_DEDICATED**	**Trigger**

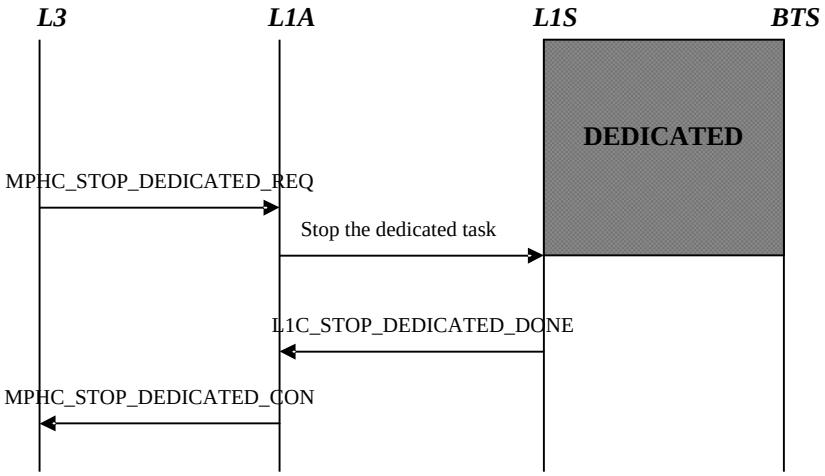
- The new interface to stop the dedicated mode is used for L1 releases from releases TCS_2.1.1_L1_1444_0, TCS_3.1_L1_1442_0 and TCS3.2_L1_1441_0.

L3->L1	**MPHC_STOP_DEDICATED_REQ**	**Trigger**
L3<-L1	**MPHC_STOP_DEDICATED_CON**	**Trigger**

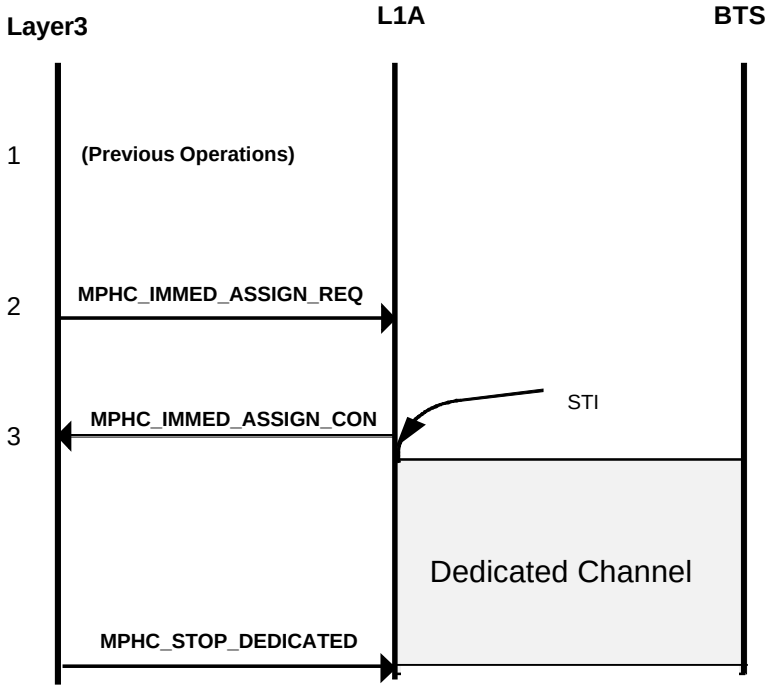
Stop sequence for old interface:



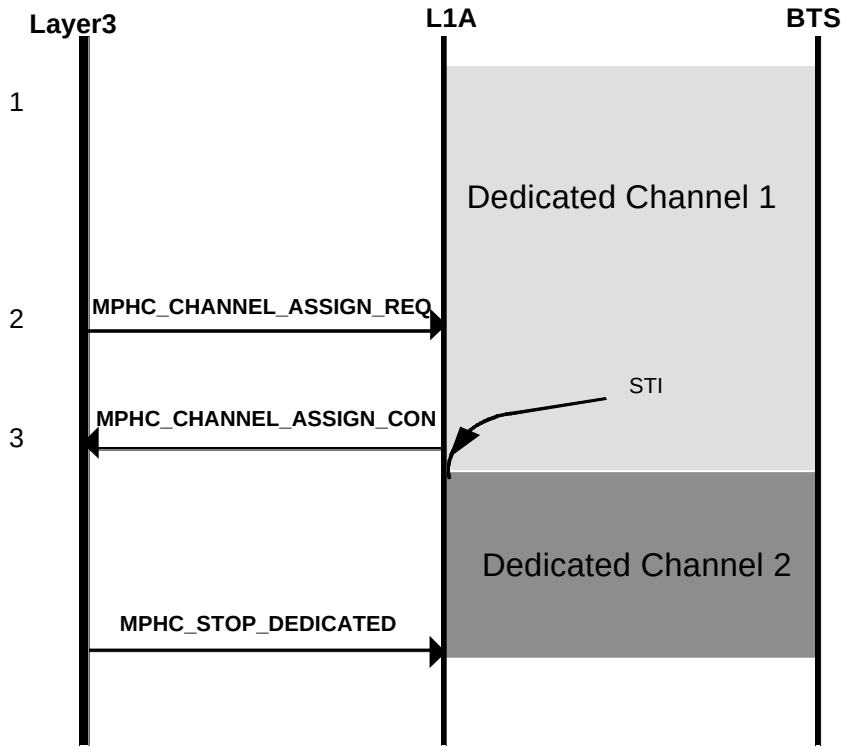
Stop sequence for the new interface:



Below are 2 examples of dedicated stop with the old interface:



Stopping Dedicated Mode (Signaling only)



Stopping Dedicated Mode (Speech)

With the new interface, the L3 should wait for the MPHC_STOP_DEDICATED_CON coming from L1, before starting any idle activity and so before sending the message MPHC_START_CCCH_REQ to L1.

31.1. Message Structures

MPHC_STOP_DEDICATED:

This is a trigger message that does not contain parameters. L3 sends this message in order to request L1 to stop dedicated mode of any type.

MPHC_STOP_DEDICATED_REQ:

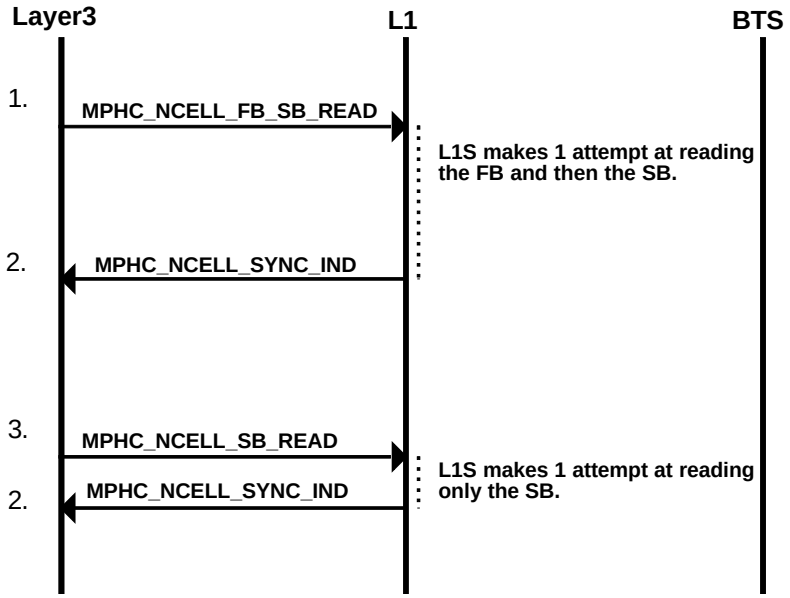
This is a trigger message that does not contain parameters. L3 sends this message in order to request L1 to stop dedicated mode of any type.

MPHC_STOP_DEDICATED_CON:

This is a trigger message that does not contain parameters. L1 sends this message in order to confirm to L3 that dedicated mode has been stopped.

32. Dedicated NCell Sync Interface (up to 6 neighbor cells)

Neighbour Cell Sync sequence



32.1. Prerequisites

It's a prerequisite that the MS is already on some dedicated channel.

32.2. Figure comments

1. L3 starts the Frequency and Sync burst detection procedure with the MPHC_NCELL_FB_SB_READ. No timing information is available. L1 starts the process to first acquire the frequency burst of the neighbour cell, then receive the sync burst.
2. MPHC_NCELL_SYNC_IND returns the sync information to L3. If detection failed the *sb_flag* is set to FALSE.
3. If the neighbour cell has previously been acquired and it's timing information is known only the sync burst needs to be re-acquired (BSIC confirmation). This can be done by L3 posting the MPHC_NCELL_SB_READ. L1 starts the process that makes a single attemp to read the sync burst.

Notes

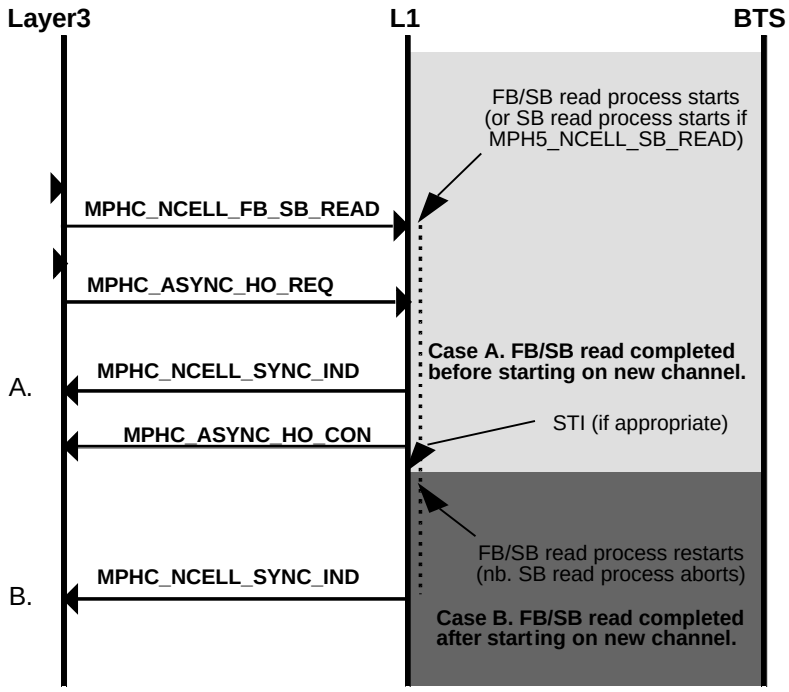
If during the FB/SB or SB read process the BA list is updated the sync attempt will be completed.

If during the FB/SB or SB read process a handover attempt is made the synchronisation process will continue until the new dedicated channel is activated (either immediately or at the starting time). If the sync attempt has successfully completed before the new dedicated channel is activated the MPHC_NCELL_SYNC_IND will be posted as normal (See Case A.).

If during the FB/SB read process the handover attempt is confirmed (i.e., MS is using the new channel) the FB/SB read process is stopped and restarted (See Case B.).

Process allowed in Packet Transfer mode and aborted with the end of the mode.

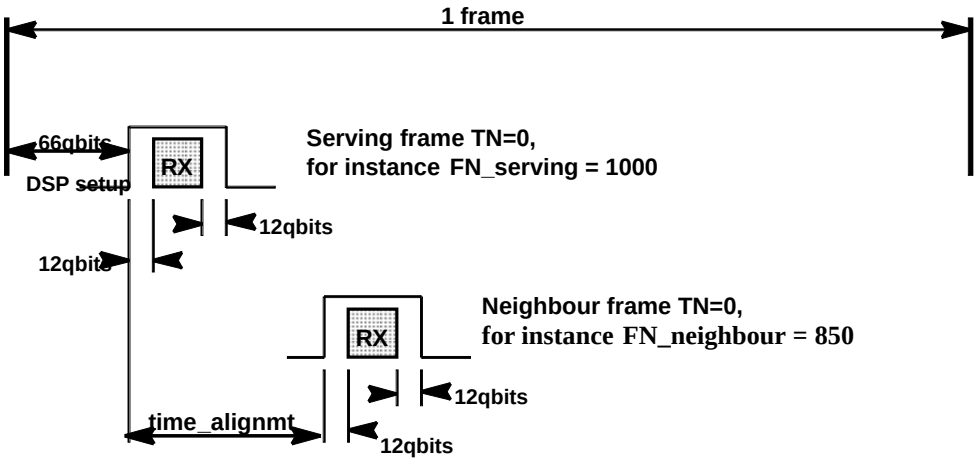
Neighbour Cell Sync sequence (with Handover)



If during the SB read process the handover attempt is confirmed (i.e., MS is using the new channel) the FB/SB read process is stopped and the invalid MPHC_NCELL_SYNC_IND message is posted. This is done because the inter-cell timing is no longer valid between the new dedicated channel and the neighbour cell.

If during a FB/SB or SB read process a MPHC_STOP_DEDICATED(_REQ) is received the process is aborted and no result message is posted.

Timing between Serving Cell and Neighbour Cell (neighbour cell leads serving cell)



Ø ***fn_offset*** is the offset in frames between the serving cell TS0 and the neighbour cell TS0 (see figure above and example below). It is always positive within the range 0 (cells are aligned) to MAX_FN-1.
In the above example we have,

fn_offset = (FN_neighbour – FN_serving) % MAX_FN
= (850 – 1000) % 2715648 = 2715498

Ø ***time_alignment*** is the fine quarter bit alignment within the frame between the serving cell TS0 and the neighbour cell TS0 (see figure above). It is always within the range 0 (cells are exactly aligned) to 4999.

32.3. Message Structures

MPHC_NCELL_FB_SB_READ

T_MPHC_NCELL_FB_SB_READ message structure
(UWORD16) **radio_freq** carrier frequency of FB/SB to be read (range 1 to 124).

MPHC_NCELL_SB_READ

T_MPHC_NCELL_SB_READ message structure
(UWORD8) **sb_flag** TRUE if SB found, else FALSE. (Not used)
(UWORD16) **radio_freq** carrier frequency of BCCH to be read (range 1 to 124).
(UWORD8) **bsic** Base Station Identification Code range: 0 to 63. [GSM 05.08 ver 4.15.0 section 9]
(UWORD32) **fn_offset** Frame offset from current serving cell
(UWORD32) **time_alignmt** Time difference in qbits between serving cell frame (TS=0) and neighbour cell frame (TS=0)

MPHC_NCELL_SYNC_IND

See paragraphe 7.3 for the definition of the structure T_MPHC_NCELL_SYNC_IND.

33. Dedicated NCell Sync Interface (up to 12 neighbor cells)

33.1. Prerequisites

The new interface is enabled by **compilation switch L1_12NEIGH** (default value is 0 except for D-sample).

The new interface supports same set of messages than the Idle mode and supports also the old interface (MPHC_NCELL_FB_SB_READ, MPHC_NCELL_SB_READ)[1]. A neighbor list has been created (max. 12 neighbor cells) that allows to receive up to 12 instances of neighbor cells monitoring requests. The MPHC_LIST_NCELL_SYNC_REQ message allows to request a list of neighbor cells monitoring (up to 12 with **eotd** field = FALSE).

- **All Rules for Ncell BA list monitoring in Idle mode apply**
- **All current transition rules of NCELL monitoring apply**.

Note [1] : Old interface is maintained but it should be removed in future.

Direction	Message name	Type
L3->L1	**MPHC_NCELL_SYNC_REQ**	**T_MPHC_NCELL_SYNC_REQ**
L3<-L1	**MPHC_NCELL_SYNC_IND**	**T_MPHC_NCELL_SYNC_IND**
L3->L1	**MPHC_STOP_NCELL_SYNC_REQ**	**T_MPHC_STOP_NCELL_SYNC_REQ**
L3<-L1	**MPHC_STOP_NCELL_SYNC_CON**	**Trigger**
L3->L1	**MPHC_NCELL_LIST_SYNC_REQ**	**T_MPHC_NCELL_LIST_SYNC_REQ**
L3->L1	**MPHC_NCELL_FB_SB_READ**	**T_MPHC_NCELL_FB_SB_READ**
L3->L1	**MPHC_NCELL_SB_READ**	**T_MPHC_NCELL_SB_READ**

Table 1: New interface for Ncell monitoring in Dedicated mode

Notes for neighbor cells BA list monitoring

- If during neighbor cell monitoring the BA list is updated the sync attempt will be completed (cf. Document [1]).
- If a handover attempt is made the monitoring process is stopped by L3 before sending the handover request (the inter-cell timing is no longer valid between the new dedicated channel and the neighbor cell). When the handover attempt is confirmed (i.e., MS is using the new channel) the neighbor cell monitoring is restarted by .
- If there is a channel change OR a new TBF assignment the monitoring are suspended by L1 and the timing information of all neighbor cells (FB/SB reading or SB confirmation) is updated. Then all

neigbor cell monitoring are restarted (Figure 1) by L1 with FB/SB reading restarted from FB reading.

- If the dedicated channel is released or all TBF are released the process is aborted as specified by transition rules (see Document [3]), no result message is posted and it must be restarted by L3

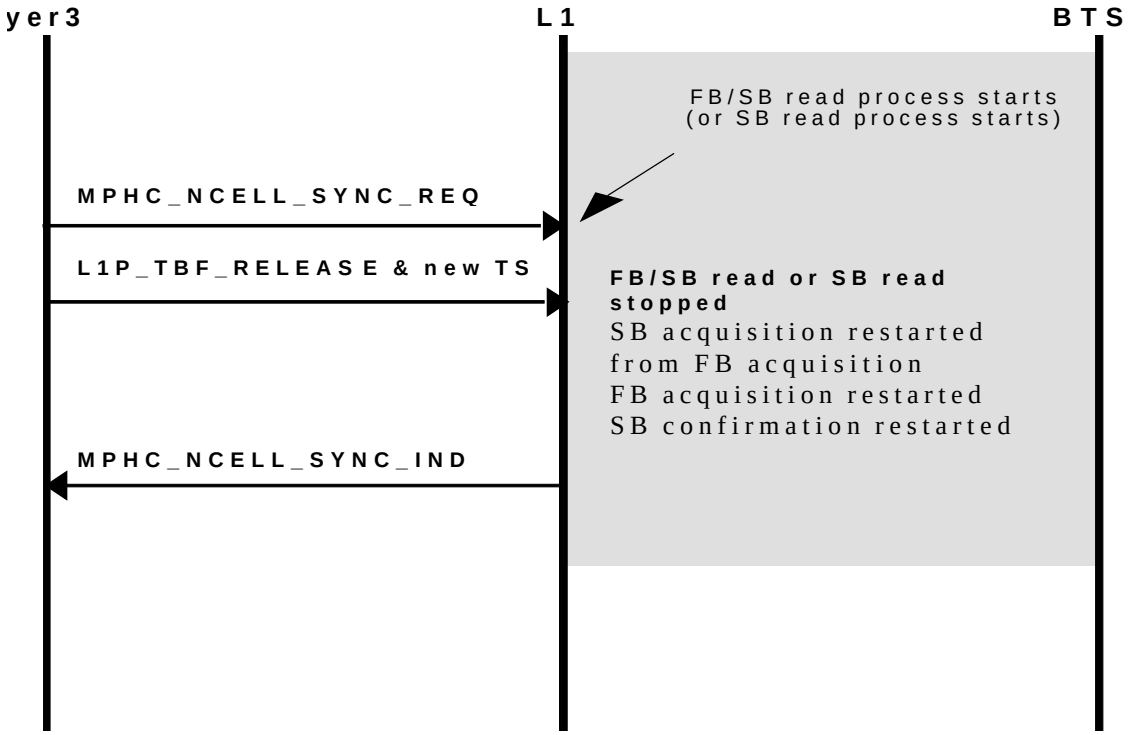


Figure 30: Ncell monitoring in Packet transfert with Handover

Notes for EOTD monitoring

- The EOTD interface is similar to the EOTD in Idle mode (i.e. MPHC_NCELL_LIST_SYNC_REQ message (up to 12 cells) with **eotd=TRUE, timing_validity=2,** no neighbor cell list update, stop only allowed for the whole list.
- All current transition rules of EOTD monitoring in Idle mode apply.
- If during Ncell monitoring the BA list is updated the sync attempt will be completed (cf. Document [1]).
- If a handover attempt is made the EOTD monitoring process is stopped by L3 before sending the handover request (the inter-cell timing is no longer valid between the new dedicated channel and the neighbor cell). When the handover attempt is confirmed (i.e., MS is using the new channel) the EOTD monitoring is restarted by L3
- If there is a channel change OR a new TBF assignment the EOTD monitoring are stopped by L1 and

the timing information of all neighbor cells (FB/SB reading or SB confirmation) is updated. Then all EOTD monitoring are restarted (Figure 1) by L1 with FB/SB reading restarted from FB reading. If EOTD process was in waiting of 1st Serving Cell SB, EOTD monitoring is restarted on 1st Serving Cell SB.

- If the dedicated channel is released or all TBF are released the process is aborted as specified by transition rules and no result message is posted. EOTD monitoring must be restarted by L3

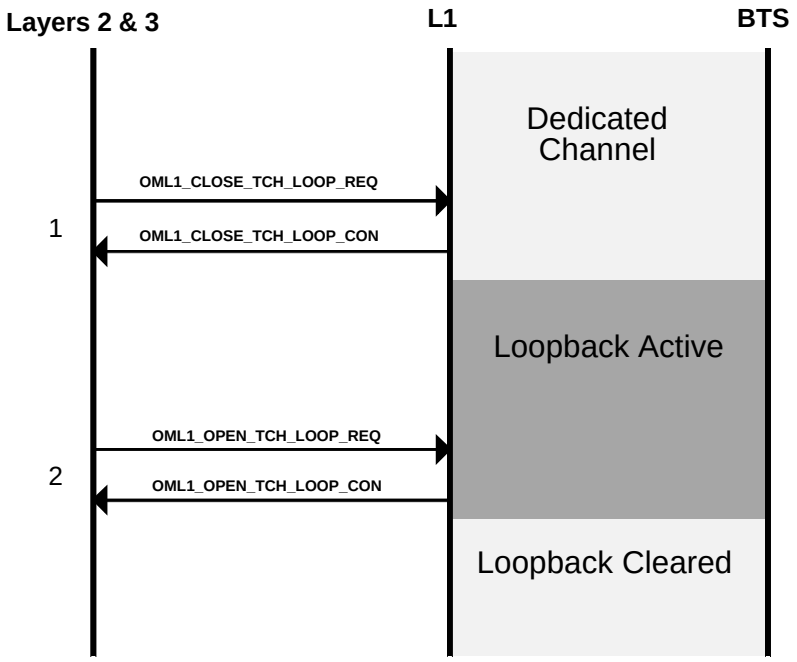
33.2. Messages structures

MPHC_NCELL_SYNC_REQ,MPHC_NCELL_SYNC_IND, MPHC_NCELL_LIST_SYNC_REQ

See paragraph 7.3 for the definition of the structure T_MPHC_NCELL_SYNC_REQ, T_MPHC_NCELL_LIST_ SYNC_REQ, T_MPHC_NCELL_SYNC_IND

34. TCH Loopback

TCH Loopback sequence



34.1. Prerequisites

Its a prerequisite that the MS is already on some dedicated channel.

34.2. Figure comments

1. L3 posts OML1_CLOSE_TCH_LOOP_REQ to L1. Layer 1 does the programming for closing of TCH loop, then posts OML1_CLOSE_TCH_LOOP_CON to L3.
2. When the loop back needs to be cleared, L3 OML1_OPEN_TCH_LOOP_REQ to L1. Layer 1 does the programming for opening the TCH loop, then posts OML1_OPEN_TCH_LOOP_CON to L3.

34.3. Message Structures

OML1_CLOSE_TCH_LOOP_REQ

T_OML1_CLOSE_TCH_LOOP_REQ message structure
(UWORD8) **sub_channel**
(UWORD8) **frame_erasure** 0 = loop A, 1 = loop B, 2 = loop C, 3 = loop D, 4 = loop E, 5 = loop F, 6 = loop I

OML1_CLOSE_TCH_LOOP_CON

This message is a trigger without parameters.

OML1_OPEN_TCH_LOOP_REQ

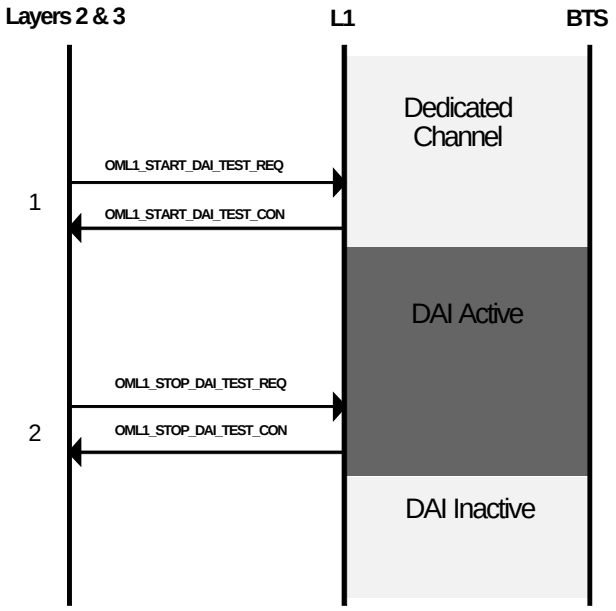
This message is a trigger without parameters.

OML1_OPEN_TCH_LOOP_CON

This message is a trigger without parameters.

35. DAI Testing

DAI Testing sequence



35.1. Prerequisites

It's a prerequisite that the MS is already on some dedicated channel.

35.2. Figure comments

1. L3 posts OML1_START_DAI_TEST_REQ to L1. Layer 1 does the programming for activation of the Digital Audio Interface, then posts OML1_START_DAI_TEST_CON to L3.
2. When the DAI needs the be stopped, L3 posts OML1_STOP_DAI_TEST_REQ to L1. Layer 1 does the programming for deactivation of the DAI, then posts OML1_STOP_DAI_TEST_CON to L3.

35.3. Message Structures

OML1_START_DAI_TEST_REQ

T_OML1_START_DAI_TEST_REQ message structure
(UWORD8) **tested_device**
0 = no test
1 = speech decoder
2 = speech encoder
3 = no test
4= Acoustic devices

OML1_START_DAI_TEST_CON

This message is a trigger without parameters.

OML1_STOP_DAI_TEST_REQ

This message is a trigger without parameters.

OML1_STOP_DAI_TEST_CON

This message is a trigger without parameters.

36.

Hardware and DSP tests

Notes:

- Single shot process (not continuous).
- Can be used for asking DSP and MCU software versions.

Direction	Message name	Type
L3->L1	**TST_TEST_HW_REQ**	**Trigger**
L3<-L1	**TST_TEST_HW_CON**	**T_ TST_TEST_HW_CON**

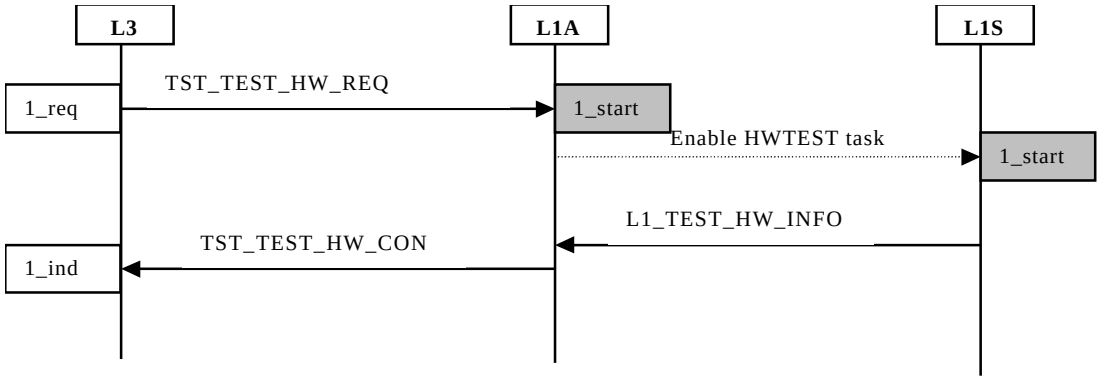


Figure 32: Hardware and DSP software versions reading, normal flow

36.1.TST_TEST_HW_CON

T_ TST_TEST_HW_CON message type

- **dsp_code_version**
- **dsp_checksum**
- **dsp_patch_version**
- **mcu_alr_version**
- **mcu_gprs_version**
- **mcu_tm_version**

dsp_code_version (UWORD16)
Version number of the DSP code.

dsp_checksum (UWORD16)
Checksum of the DSP software (code + patch)

dsp_patch_version (UWORD16)
Version number of the DSP patch code.

mcu_alr_version (UWORD16)
Version number of the MCU ALR code.

mcu_gprs_version (UWORD16)
Version number of the MCU GPRS code. (0 in case of no GPRS code)

mcu_tm_version (UWORD16)
Version number of the Testmode

37. Custom Functions

This chapter contains the description of the custom functions included within L1 software. These functions are customer dependent since they usually depend on RF implementation.

37.1. Transmit Power Management

The transmit power level management is a complex task handled within the physical layer. This task makes use of the TPU, the VEGA (TCM4400) and the MCU L1 software which is the master of the process.

L1 software provides 2 functional modes selected via configuration using the configurable structure l1_config. The structure element l1_config.tx_pwr_code can be initialized to zero or to one. This initialization corresponds to the following modes:

l1_config.tx_pwr_code = 0, Fixed Transmit power and RAMP shapes.
This mode allows to integrate and test the MS system with a fixed transmit power level and a fixed ramp up/down set, whatever the GSM **TXPWR** parameter setting. This mode is to be used for system integration test only, it cannot be used for “real life” MS implementation.

l1_config.tx_pwr_code = 1, GSM controlled Transmit power with according RAMP shapes.
This mode is the one used to fully comply with GSM requirements. To this avail, all components must be tuned: the TPU drivers, the GSM transmit power regulation algorithm, the APC (Adaptive Power Control, part of VEGA) and the RAMP UP/DOWN shapes.

37.1.1. Fixed Transmit power and RAMP shapes

In this mode (l1_config.tx_pwr_code=*0),* any transmission (Uplink) is made using a fixed transmit power level value and a fixed ramp up/down shapes set. The GSM transmit power regulation algorithm implemented within L1 has no effect. There is no special requirement put on the TPU drivers.

The fixed transmit power level value is taken from the file ***"l1_rf#.h"*** (file which contains all RF dependant definitions), it is labeled **"FIXED_TXPWR".** This is directly the value stored by L1 in the APC register within VEGA (see TCM4400 data sheet for more details on the format of this data). The fixed RAMP UP/DOWN shape set is taken from the table "ramp_tables" from the file ***"l1_rf#.c"***. These ramps are stored in VEGA on the first Uplink transmission, which is made on RACH.

37.1.2. GSM controlled Transmit power with according RAMP shapes

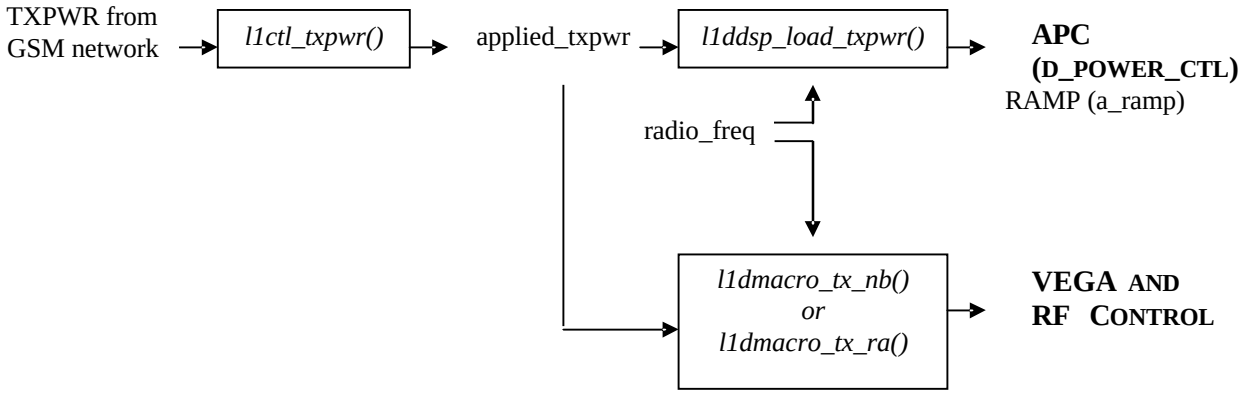
In this mode *(l1_config.tx_pwr_code=1),* the goal is to setup a transmit power management system which follows the GSM requirements in time domain (burst level and shape), and frequency domain (modulation spectrum and switching transients spectrum).

After several RF analysis, we have built a L1 skeleton which contains a general approach to cope with all identified non linearity taking place in the RF transmit power path. Let us list these non linearity:

The GSM parameter ***TXPWR*** is translated into an APC command. The PA response to this APC command is non linear ("TXPWR TO APC" translation is not linear) and is frequency dependant.

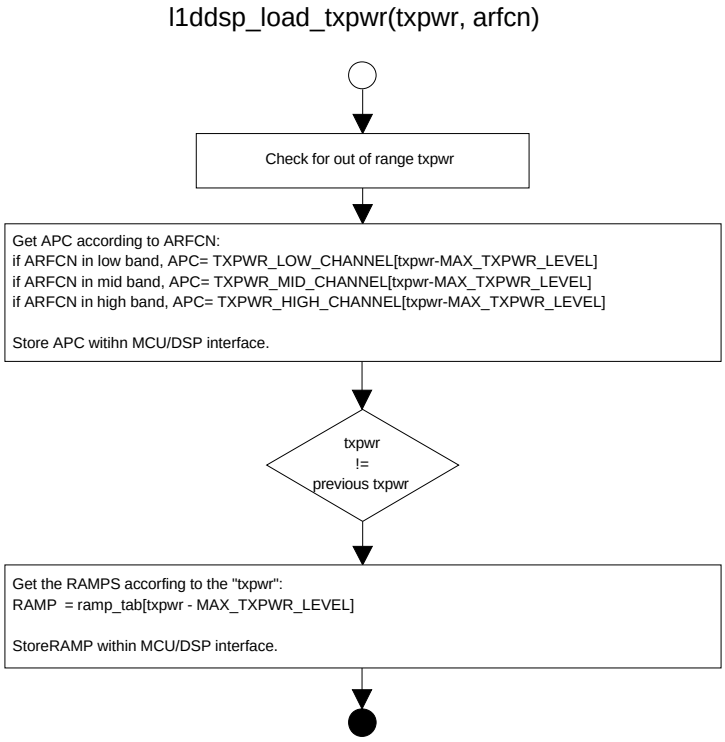
The **RAMP UP** and **RAMP DOWN** shapes cannot be unique. It seems that there is a strong relationship between the APC value and the RAMP UP and RAMP DOWN shapes which satisfy the GSM constraints.

The Power Amplifier (PA) switch-on timing (part of uplink TPU scenario) has a big impact on the burst shape and therefore on the output spectrum.



Here is given a brief description of the MCU L1 software which handles the transmit power management. This chain contains some provisions to cope with the non linearity we have listed above.

The control function *"l1ctl_txpwr()"*strictly implements the MS power control strategy according to GSM 05.08, chapter 4.3. This function output the TXPWR value to be effectively applied (called "applied_txpwr") to the uplink signal at any time. The driver function *"l1ddsp_load_txpwr()"* uses it together with the radio_freq and outputs the APC and the RAMPS. Here below is given a description of this function.



The table "apc_bands[]" allow to build a model of the PA response for as many frequency sub-domains as needed. This table is typically built using a custom calibration process, in our generic layer1, the table contains constants defined in the file "l1_rf#.c".

The 15 tables "ramp_tables[0..14]" allow to build a model of the burst shape for each possible TXPWR (15 possible values from 5 to 19). In other words, each

possible TXPWR is associated with a ramp set. These tables are typically built using a custom calibration process, in our generic layer1, the tables are constants defined in the file “l1_rf#.c”.

The tpu driver functions “l1dmacro_tx_nb()” and “l1dmacro_tx_ra()” are called when building the TPU scenario for uplink. These two functions are by definition (since part of the TPU drivers) customer/RF dependent. To allow the customer to implement easily the body of these functions, we pass the “applied_txpwr” and the radio_freq as parameter. These parameters are the one identified as potentially impacting the uplink tpu scenario body. It is possible for instance to use these parameters to select the correct timing for PA switch-on.

37.2. Auxiliary conversions

TI GSM system offer includes some auxiliary conversion capabilities. These conversions are supported within Analog Base Band by the so called *10-bit "monitoring A/D converter"* (see Analog Base Band data sheet for more information). This auxiliary A/D converter allows reading up to 9 analog inputs to the OMEGA used generally for battery, temperature and accessories monitoring.

The MCU layer 1 software integrates a mechanism to read the 9 auxiliary analog inputs to the Analog Base Band.

In the following of this document we introduce the different parts described here below. The separation between the parts (functional split) is conceptual. Different design choices can be made grouping some of these parts in the same entity without changing anything to the concept described in this document.

ADC manager

This is a task (or part of an existing task) in charge of requesting ADC monitoring to L1 on a regular basis. This task must either be aware of the MS state (Cell Selection / Idle / Dedicated / Packet Transfer) or implement a "watchdog like" software timer in order to be able to adapt the request periodicity and qualifier (inside / outside TX burst) to the phase and address the side effect cases introduced by the system.

SPI manager

This is a task in charge of collecting all request of SPI access from any task in the system. Its role is then to prioritize the requests and serve them one after the other.

L1

Layer1 software. This task is in charge of programming the ADC reading TPU scenario according to the request received from the ADC manager.

TPU

Time Processor Unit. This part is in charge of executing the ADC reading TPU scenario programmed by L1S. Basically, this scenario consists in triggering the

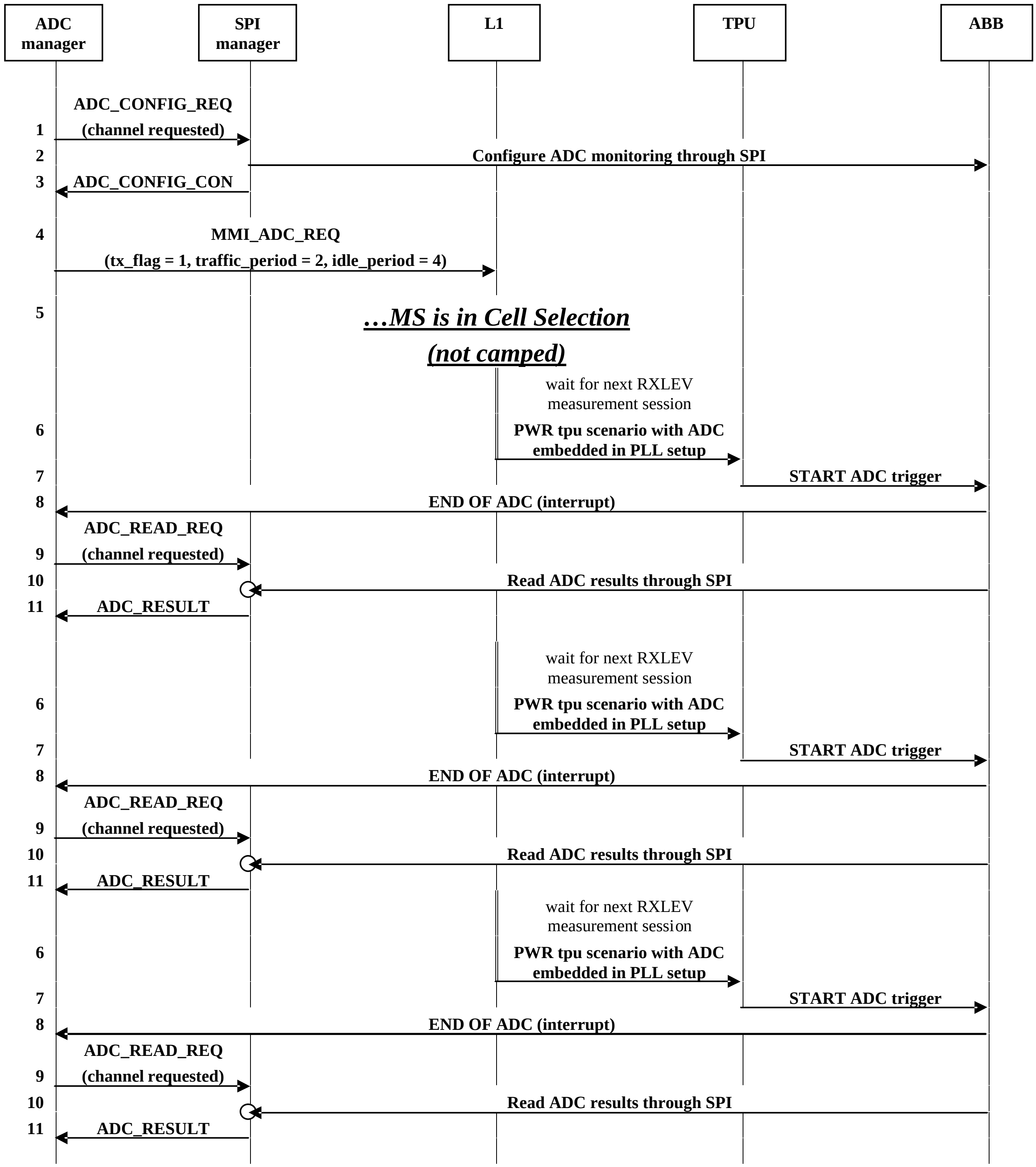
Analog Base-Band (ABB) to start the ADC conversion sequence.

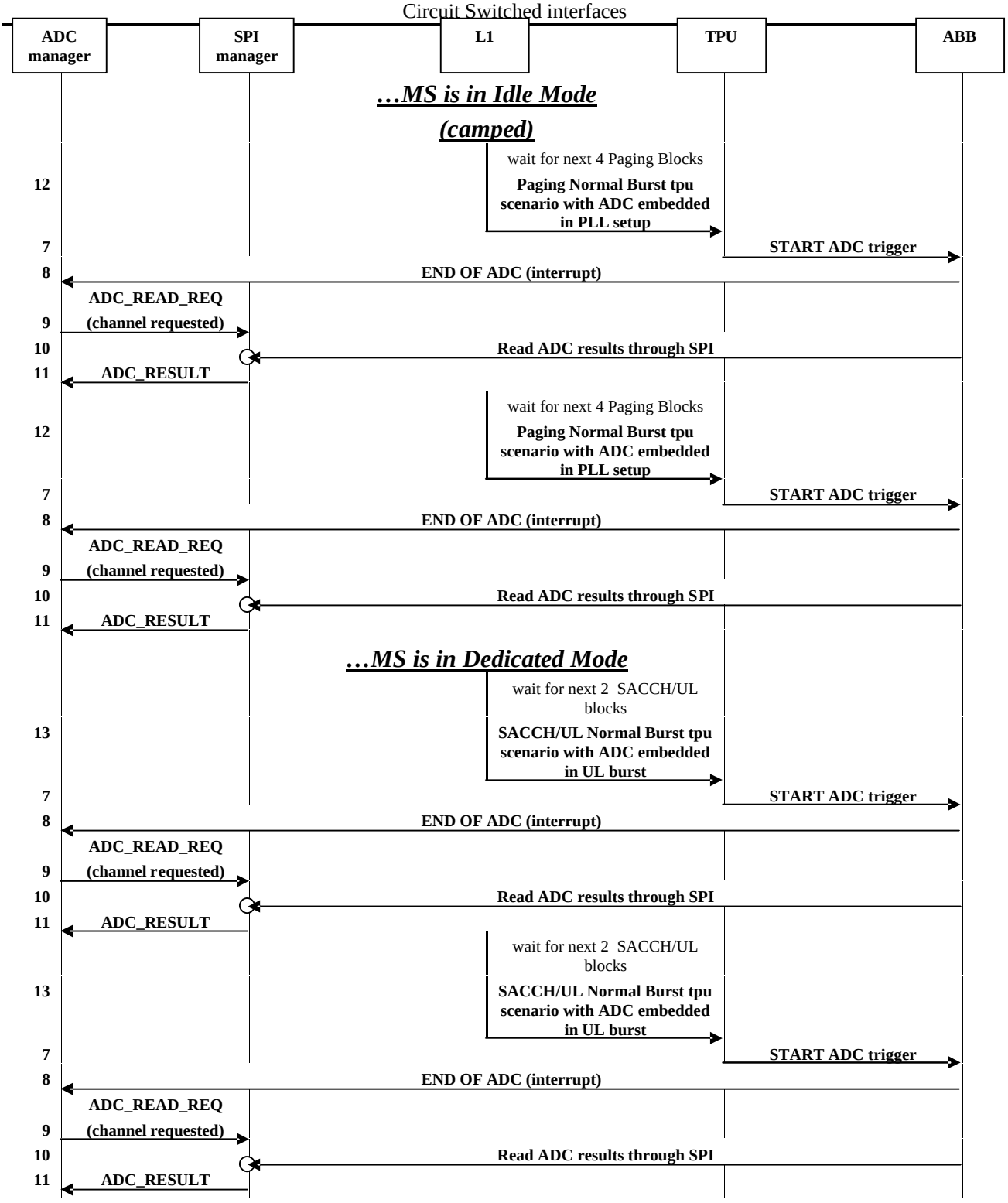
ABB

Analog Base-Band. This part performs the ADC conversion sequence and generates an Interrupt to the MCU when all conversions have been completed.

Notes:

Direction	Message name	Type
L3->L1	**MMI_ADC_REQ**	**T_MMI_ADC_REQ**
L3<-SPI	**CST_ADC_RESULT**	**T_CST_ADC_RESULT**
L3->L1	**MMI_STOP_ADC_REQ**	**Trigger**
L3<-L1	**MMI_STOP_ADC_CON**	**Trigger**





1. ADC manager requests SPI manager to configure the ADC monitoring for a given channel set. (This phase can be done once at system initialisation).
2. SPI manager configures the ADC monitoring through SPI within the Analog Base-Band. (This phase can be done once at system initialisation).
3. SPI manager confirms the ADC monitoring configuration when the configuration is complete. (This phase can be done once at system initialisation).
4. **ADC manager requests L1 to perform the ADC monitoring with parameters "tx_flag=1", "traffic_period=2" (ADC performed within TX burst when L1 is in dedicated mode), within a TX burst. This is achieved by sending a message MMI_ADC_REQ to L1A, with parameter inside_tx=1.**
5. MS is in Cell Selection mode (MS not camped).
6. **L1 waits for next Cell Selection Measurement Session (MPHC_RXLEV_REQ), and then programs the ADC reading within the first Power Measurement activity of that session.**
7. TPU executes the ADC reading scenario. It starts the ADC conversion sequence.
8. ABB generates an interrupt when the ADC conversion sequence is complete.
9. ADC manager posts a request to the SPI manager to read the ADC results from ABB.
10. SPI manager reads the ADC results from ABB through the SPI.
11. SPI manager indicates the ADC result the ADC manager.
12. **L1 waits for next 4 Paging blocks, and then programs the ADC reading within the first Noraml Burst activity of the 4th one (according to "idle_period" parameter).**
13. **L1 waits for next 2 SACCH/UL blocks, and then programs the ADC reading within the Normal Burst activity of the 2nd one (according to "traffic_period" parameter).**

37.2.1. ADC execution algorithm within L1

IF (ADC requested)
- **IF** (Dedicated mode)
  - **IF** (tx_flag == 1) // inside TX
    - **IF** (traffic_period == 0) // Single shoot
      Trigger ADC within **next** SACCH/**UL** burst (1).
    - **ELSE** // Periodic
      Trigger ADC within **each** SACCH/**UL** burst block according to "traffic_period" periodicity (2).
  - **ELSE** // outside TX
    - **IF** (traffic_period == 0) // Single shoot
      Trigger ADC within **next** SACCH/**DL** (3).
    - **ELSE** // Periodic
      Trigger ADC within **each** SACCH/**DL** block according to "traffic_period" periodicity (4).
  - **ENDIF**
- **ELIF** (Transfer mode)
  - **IF** (tx_flag == 1) // inside TX
    - **IF** (traffic_period == 0) // Single shoot
      Trigger ADC within **next** PTCCH/**UL** burst (9).
    - **ELSE** // Periodic
      Trigger ADC within **each** PTCCH/**UL** burst block according to "traffic_period" periodicity (10).
  - **ELSE** // outside TX
    - **IF** (traffic_period == 0) // Single shoot
      Trigger ADC within **next** PDTCH/**DL** (11).
    - **ELSE** // Periodic
      Trigger ADC within **each** PDTCH/**DL** block according to "traffic_period" periodicity (12).
- **ENDIF**
- **ELIF** (Idle mode ou Packet Idle mode)
  - **IF** (idle_period == 0) // Single shoot
    Trigger ADC within **next** (Packet) Normal Paging block according to "idle_period" periodicity (5).
  - **ELSE** // Periodic
    Trigger ADC within **each** (Packet) Normal Paging block according to "idle_period" periodicity (6).
  - **ENDIF**
- **ELIF** (Connection Establishment mode)
  - **IF** (tx_flag == 1) // inside TX

Trigger ADC within **each** (packet) RACH burst (1).

ELSE

IF (idle_period == 0) // Single shoot

Trigger ADC within **next** (Packet) Normal Paging block according to "idle_period" periodicity 13).

ELSE // Periodic

Trigger ADC within **each** (Packet) Normal Paging block according to "idle_period" periodicity (14).

ENDIF

ENDIF

ELIF(Cell Selection)

IF (idle_period == 0) // Single shoot

Trigger ADC within **next** Measurement Session (7).

ELSE // Periodic

Trigger ADC within **each** Measurement Session (8).

ENDIF

ELSE // no GSM mode

IF (idle_period == 0) // Single shoot

Trigger ADC on the **next** frame (15).

ELSE // Periodic

Trigger ADC on the **each** frame according to "idle_period" periodicity (16).

ENDIF

ENDIF

ENDIF

(1) ADC must be executed within a TX burst not subject to DTX. L1 use SACCH/UL, maximum latency is 26 frames. Single shoot.

(2) ADC must be executed within a TX burst not subject to DTX. L1 use SACCH/UL, maximum latency is 104 frames. Periodicity is given by "traffic_period".

(3) ADC must be executed outside TX burst. L1 use SACCH/DL, maximum latency is 26 frames. Single shoot.

(4) ADC must be executed outside TX burst. L1 use SACCH/DL, maximum latency is 104 frames. Periodicity is given by "traffic_period".

(5) ADC must be executed within Normal Paging activity. Maximum latency is 9x51=459 frames in Idle and 3328 frames within Packet Normal Paging. Single shoot.

(6) ADC must be executed within Normal Paging activity. Maximum latency is 9x51=459 frames in Idle and 3328 frames within Packet Normal Paging. Periodicity is given by "idle_period".

(7) ADC must be executed within next Measurement Session. Maximum latency depends on Protocol stack cell selection strategy. Single shoot.

(8) ADC must be executed within next Measurement Session. Maximum latency and persiodicity depends on Protocol stack cell selection strategy.

(9) ADC must be executed within a TX burst not subject to DTX. L1 use PTCCH/UL, maximum latency is 416 frames. Single shoot.

(10) ADC must be executed within a TX burst not subject to DTX. L1 use PTCCH /UL, maximum latency is 416 frames. Periodicity is given by "traffic_period".

(11) ADC must be executed outside TX burst. L1 use PDTCH/DL, maximum latency is 2 frames. Single shoot.

(12) ADC must be executed outside TX burst. L1 use PDTCH /DL, maximum latency is 104 frames. Periodicity is given by "traffic_period".

(13) ADC must be executed within TX burst. L1 use RACH, maximum latency depends on Protocol stack. Single shoot.

(14) ADC must be executed within TX burst. L1 use RACH, maximum latency depends on Protocol stack. Periodicity is given by "traffic_period".

(15) ADC must be executed on the next. Single shoot.

(16) ADC must be executed on the next frame according to the "Idle position" periodicity. Maximum latency is 102 frames. Periodicity is given by "idle_period".

37.2.2. MMI_ADC_REQ

T_MMI_ADC_REQ message structure

UWORD8 tx_flag

Specifies where the ADC conversion must take place during Dedicated or Packet Transfer mode or Connection Establishment mode.

0 outside a TX burst.
1 Inside a TX burst.

UWORD8 traffic_period

This parameter specifies the period to which ADC conversion sequence must be triggered by L1 during Dedicated or Packet Transfer mode (uplink is active).

Range 0 to 127.

Note: "0" means no periodicity, L1 triggers the ADC conversion in the next coming activity and stops the activities (1) (2) and (3) (single shoot).
Note: ADC conversion associated with "idle_period" is still running.

(1) In Dedicated mode, the ADC conversion will be triggered every "traffic_period" multi-frame 104, in the SACCH activity.

(2) In Connection Establishment mode with tx_flag enabled, the ADC conversion will be triggered every (packet) RACH activities (the periodicity is meaningless).

(3) In Packet Transfer mode, the ADC conversion will be triggered every "traffic_period" multi-frame 104, in the PDTCH activity when tx_flag is disabled otherwise in PTCCH activity.

UWORD8 idle_period

This parameter specifies the period to which ADC conversion sequence must be triggered by L1 during Idle mode This parameter is valid in Idle mode (MS camped).

Range 0 to 127.

Note 1: "0" means no periodicity, L1 triggers the ADC conversion in the next coming DL activity and stops the activities (1) (2) (3) and (4) (single shoot).
In No GSM mode (Cf. the section "Network Synchronization Lost"), the ADC conversion is performed in the current frame

Note 2: ADC conversion associated with "traffic_period" is still running.

(1) In Idle mode and in Connection Establishment mode with tx_flag disabled, the ADC conversion will be triggered every "idle_period" paging period (51 frames times BS_PA_MFRMS), in the Normal Paging activity.
In Connection Establishment mode with tx_flag enabled the ADC conversion will be triggered every RACH activities (the periodicity is meaningless).
In Idle mode the tx_flag is meaningless.

(2) In Packet Idle mode and in Packet Connection Establishment mode with tx_flag disabled, the ADC conversion will be triggered every "packet idle_period" packet paging period (51 frames times packet paging periodicity), in the Normal Packet Paging activity.
In Connection Establishment mode with tx_flag enabled the ADC conversion will be triggered every packet RACH activities (the periodicity is meaningless).
In Packet Idle mode the tx_flag is meaningless.

(3) In Cell selection mode, the ADC conversion will be triggered every Measurement Session (started by MPHC_RXLEV_REQ) (only in the first frame of the session)

(4) In No GSM mode (Cf. the section "Network Synchronization Lost"), the ADC conversion will be triggered each time: (FN modulo (102 times "idle_period")) == 0.
Note: the tx_flag is meaningless in this mode.

37.2.3. CST_ADC_RESULT

T_CST_ADC_RESULT message type
(UWORD16) **adc_result[9]** **AN ARRAY THAT CONTAINS THE 9 AUXILIARY ANALOG INPUT READING RESULTS**

38. Power Management Interface

In the current implementation, primitive sent to Layer3 enables the Power management feature, in order to facilitate the debug. TST_SLEEP_REQ primitive is defined as follows:

T_TST_SLEEP_REQ message type

```
Typedef struct
{
 byte            sleep_mode;
 UNSIGNED_SHORT  clocks;
}
T_TST_SLEEP_REQ
```

Allowed values for **sleep_mode**:

0 no sleep mode
1 small sleep
2 big sleep
3 deep sleep
4 all sleep modes

The **clocks** parameter is a bitmap of the modules clocks that the BIG sleep may switch OFF *(GSM 1.0 only while on GSM1.5 the BIG sleep feature does not switch OFF the clocks):*

```
#define ARMIO_CLK     0x0001
#define RIF_CLKR      0x0002
#define RIF_CLKX      0x0004
#define RIF_CLK13     0x0010
#define UWIRE_CLK     0x0020
#define SIM_CLK       0x0040
#define TSP_CLK       0x0080
```

```
#define UART_CLK        0x0400
```

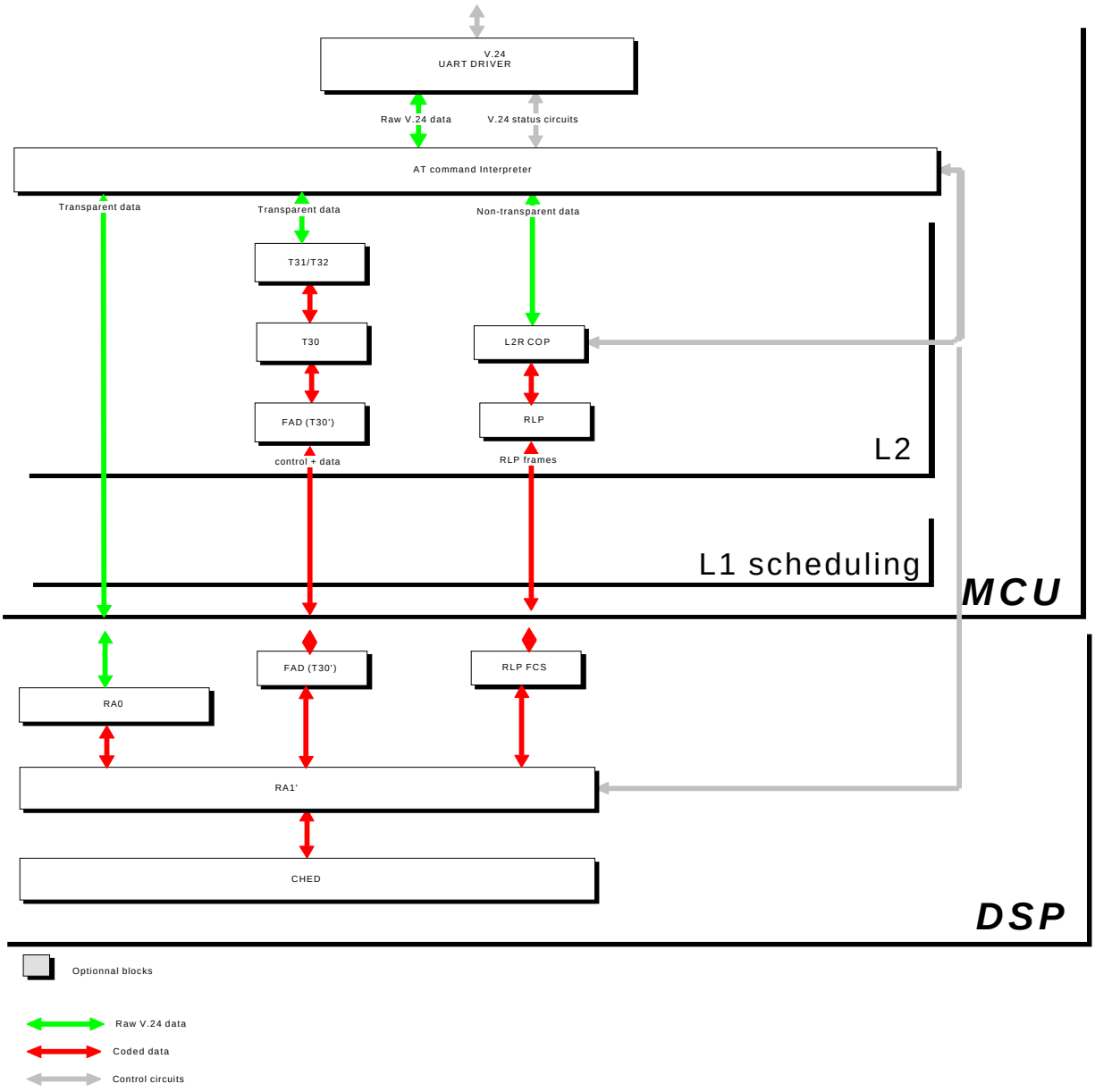
At reception of this primitive, the clock bitmap is masked with “0x5FF” while TPU_CLK and SWITCH_IRQ_CLK remain ON during big sleep. The Layer3 sets **l1s.pw_mgr.mode_authorized** that will be used by the **l1s_sleep_manager()**. At wake-up RIF_CLKR, RIF_CLKX, RIF_CLK13 are switch ON systematically for GSM purpose. The other modules clocks are restored if necessary.

Changing of the **sleep_mode** or **clocks** parameters during a scenario execution is not recommended and may result in some trouble of the software behavior. We advise to send this setup message at first.

39. Data application Interface

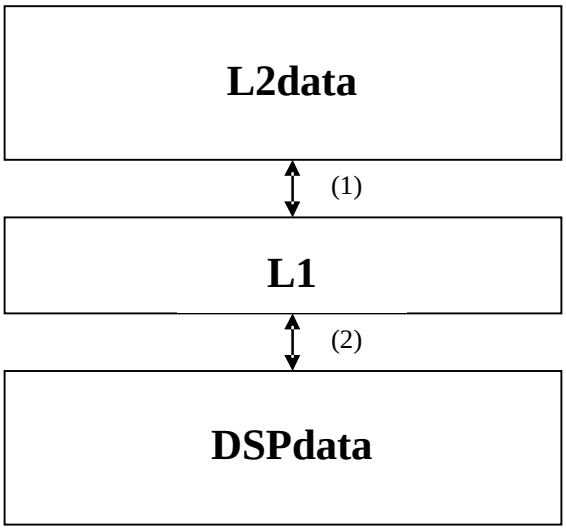
39.1. General

The data flow protocol/processing software has been partitioned between the MCU and the DSP as shown in figure below.



The MCU software part and DSP software part are respectively called **"L2data"** layer and **"DSPdata"** layer in the rest of this chapter.

A third layer, L1, plays the role of time scheduler and manages the MCU/DSP interface. Naturally L1 is then situated between the **L2data** layer and the **DSPdata** layer as shown in the figure below.



This is the L1 / L2data interface, subject of this specification.

This is the MCU/DSP communication interface defined is L1_GS800 for the general aspects and in S844 for the details on Data Traffic.

This chapter defines the L1/L2data interface for each of the Data traffic modes (RA / RLP / FAX). Same approach will be used for V42bis/MNP5 and next data applications. It provides also the detailed schedule for the use of this interface. The details concerning the content and meaning of the different buffers and their control information are defined in a separate specification S844 ("Data Interfaces").

39.2. L1 / L2data Interface

The interface between L1 and L2data is achieved by two functions used as shown in the figure below. These functions are provided by L2data (they are part

of it), they are defined as follow:

- **void dll_data_ul(** API *ul_buffer_address,
  API *d_ra_conf,
  API *d_ra_act,
  API *d_ra_statu,
  API *d_fax **)**

- **void dll_data_dl(** API *dl_buffer_address,
  API *d_ra_act,
  API *d_ra_statd **)**

Where,

```
#if CHIPSET == 0
  typedef UWORD32  API;
#endif

#if CHIPSET == 2
  typedef UWORD16  API;
#endif
```

ul_buffer_address / dl_buffer_address are the addresses of the Uplink and Downlink data buffers used to communicate between the MCU and the DSP.

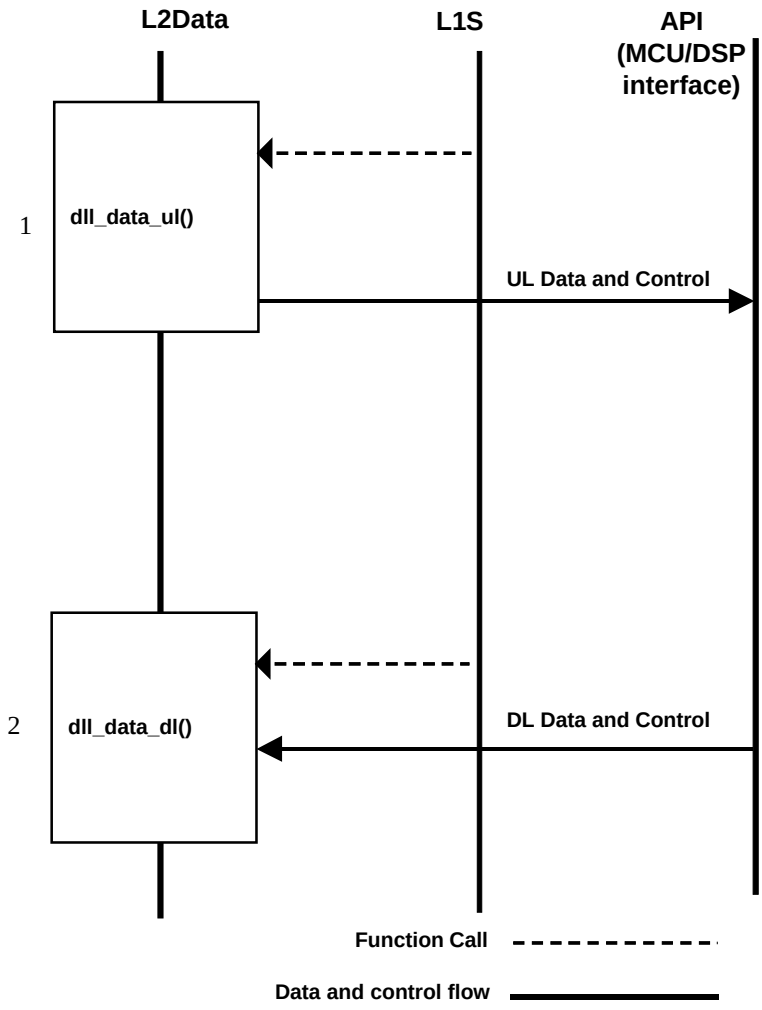
d_ra_conf is the general data traffic control register (as defined within S844).

d_ra_act is the activity word (as defined within S844).

d_ra_statu is the uplink status word for the RADAP module (as defined within S844).

d_ra_statd is the downlink status word for the RADAP module (as defined within S844).

d_fax is the status and parameter word for the FAX module (as defined within S844).



(1) L1 gets the UL data block from L2data at the correct time by calling the "dll_data_ul()" function. It then processes it (performs the Rate Adaptation for instance) and sends it to the base station.

(2) L1 receives the DL data from the base station, processes the data (performs the Rate Adaptation for instance) and pass the result data block to L2data by calling the "dll_data_dl()" function.

The following table summarizes the schedule of the use of the L1/L2data interface for each GSM data traffic rate. The FN mentioned is the Full Frame Number as defined in GSM specification except otherwise indicated.

Data Rate	Downlink: Call of function dll_data_dl ()	Uplink: Call of function dll_data_ul ()	Comments
TCH/F14.4	FN % 13 % 4 = 2	FN % 13 % 4 = 2	(a)
TCH/F9.6	FN % 13 % 4 = 2	FN % 13 % 4 = 2	(a)
TCH/F4.8	FN % 13 % 4 = 2	FN % 13 % 4 = 2	(a)
TCH/F2.4	FN % 13 % 4 = 0	FN % 13 % 4 = 2	(a)
TCH/H4.8 SC0	Not defined	FN % 26 = 6 / 15 / 23	Half Rate Data not yet supported
TCH/H4.8 SC1	Not defined	FN % 26 = 7 / 16 / 24	Half Rate Data not yet supported
TCH/H2.4 SC0	Not defined	FN % 26 = 6 / 15 / 23	Half Rate Data not yet supported
TCH/H2.4 SC1	Not defined	FN % 26 = 7 / 16 / 24	Half Rate Data not yet supported

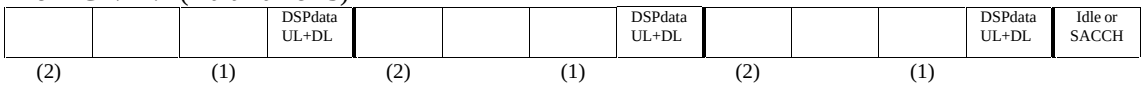
Table: L1/L2 data interface scheduling

(a) From the fact that periodicity is modulo %13%4, FN information can be used as FN_in_report information for TN != 0.

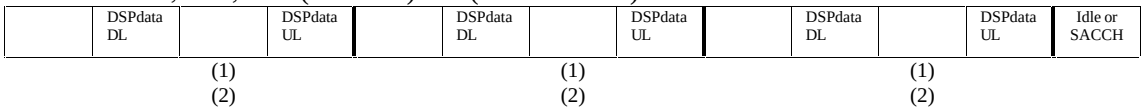
Note:

Due to the C/W/R pipeline operation of the MCU/DSP interface (see S820), the DSPdata and L1 are never accessing to the MCU/DSP interface at the same time as shown in figure below. A particular attention will be caught to TCH/F4.8 mode due to its 40ms time structure:

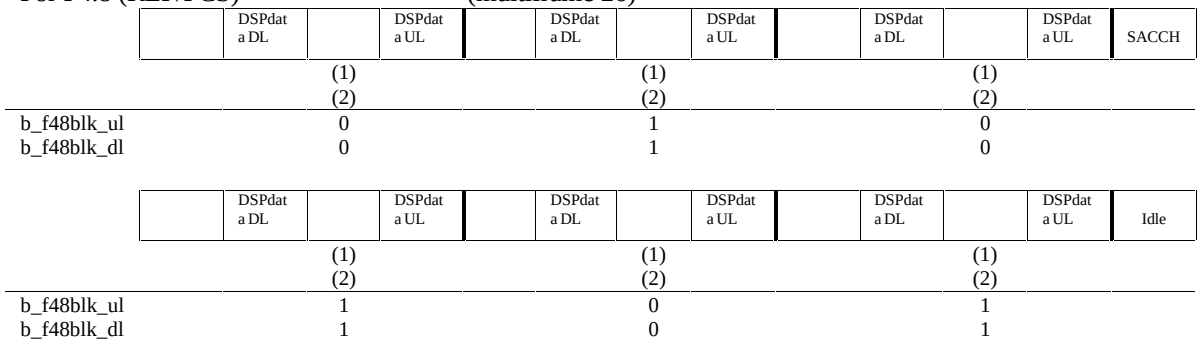
For TCH/F2.4 (multiframe 13)



For TCH/14.4, F9.6, F4.8 (RA/FAX) (multiframe 13)



For F4.8 (RLP/FCS) (multiframe 26)



(1) L1 calls “dll_data_ul()” which loads the MCU/DSP interface with the uplink data block with its associated control. The DSP to start its uplink processing on the next frame (shown on the figure by “DSPdata UL”).

(2) L1 calls “dll_data_dl()”. The downlink data block with its associated status have been stored by the DSP in the previous frame (shown on the figure by “DSPdata DL”).

The function dll_data_dl() is called during the READ phase and the function dll_data_dl() is called during the CONTROL phase. When these functions are called in the same TDMA (F14.4,F9.6,F4.8), the dll_data_dl() function is called before the dll_data_ul() function.

TCH/F4.8 RLP/FCS specificity: two control bits (mapped in “d_ra_act”)

- b_f48blk_ul for UL Data Block Tx
- b_f48blk_dl for DL Data Block Rx

have been created in order to specify both to L2 and DSP which block needs to be processed (cf. S844 §2.4).
As detailed in S844 §2.4:

- b_f48blk_ul = 1 means first part (120 bits) of the 216 bits (120 + 96) is transmitted from L2 to DSP on a L1S request (dll_data_ul()). b_f48blk_ul = 1 corresponds to the beginning of a block to Tx (Block B0, B2, B4, cf. GSM 05.02)
- b_f48blk_ul = 0 means second part (96 bits) of the 216 bits is transmitted from L2 to DSP on a L1S request (dll_data_ul()). FCS is validated after receiving this block.
- b_f48blk_dl = 0 means first part (120 bits) of the Rx data block is received and going to be processed by the DSP
- b_f48blk_dl = 1 means second part (96 bits) of the 216 bits (120 + 96) is currently received and going to be processed by the DSP. b_f48blk_dl = 1 marks the end of a received data block (Block B1, B3, B5, cf. GSM 05.02). FCS is validated after processing of this block.

b_f48blk_ul and b_f48blk_dl are managed by L1S. These bits are updated on FN %13%4 = 2 as detailed in **Error! Reference source not found.**. b_f48blk_ul and b_f48blk_dl (d_ra_act) are updated before each dll_data_ul() call in order to:

- specify to L2 how many bits are expected by L1S: 120 or 96 bits
- to indicates to DSP if first or second part of data block has been received. This information is passed from L1S to DSP during dll_data_ul() call to be present before the DSP/CHED processing

starting time: let’s assume an assignment on FN = 24,25,0,1,2 or 7,8,9,10. This means in UL that L2 writes 96 bits in MCU-DSP memory (second part of UL data block), DSP processes this 96 bits and computes the FCS only on these 96 bits, in spite of the fact that FCS will be bad, the Data block will be sent.

On the other hand, let’s assume an assignment on FN = 3,4,5,6 or 20,21,22,23. This means in DL that DSP has received only the second part of the DL data block. DSP RLP/FCS module will be call all the same.

39.3. Use of L1 / L2data Interface for Rate Adaptation (RA)

- dll_data_ul(**a_rau,**
  &d_ra_conf,
  &d_ra_act,
  &d_ra_test,
  &d_ra_statu,
  Not Used*)

- dll_data_dl(**a_rad,**
  &d_ra_act,
  &d_ra_statd)

As already mentioned, all parameters used are fully described in specification S844.

* Not Used means that a pointer is transmitted by L1 to L2 but L2 does not take care of the value.

39.4. Use of L1 / L2data Interface for RLP/FCS calculation

- dll_data_ul(**a_ntu,**
  &d_ra_conf,
  &d_ra_act,
  &d_ra_test,
  Not Used,
  Not Used)

- dll_data_dl(**a_ntd,**
  &d_ra_act,
  Not Used)

As already mentioned, all parameters used are fully described in specification S844.

39.5. Use of L1 / L2data Interface for FAX

- dll_data_ul(**a_faxu,**
  &d_ra_conf,
  &d_ra_act,
  &d_ra_test,
  &d_ra_statu,

&d_fax)

Ø dll_data_dl**(a_faxd,**
&d_ra_act,
&d_ra_statd)

As already mentioned, all parameters used are fully described in specification S844.

40. Multiband interface

Direction	Message name	Type
L3->L1	**MPHC_INIT_L1_REQ**	**T_MPHC_INIT_L1_REQ**
L1->L3	**MPHC_INIT_L1_CON**	**Trigger**

40.1. MPHC_INIT_L1_REQ

T_MPHC_INIT_L1_REQ message type

- **radio_band_config**

radio_band_config (UWORD8)
Layer 1 radio frequency band configuration. (See table below)

Frequency Band configuration	L3/L1 interface parameter "radio_band_config"initialized by "STD"	ARFCN coding according to ETSI GSM spec.	"ARFCN" coding used in L1
GSM 900	1	1-124	1-124
E-GSM	2	1-124 975-1023,0	1-124 125-174
PCS 1900	3	512-810	512-810
DCS 1800	4	512-885	512-885
DUAL BAND GSM 900/DCS 1800	5	1-124 512-885	1-124 125-498
DUALEXT BAND E-GSM/DCS 1800	6	1-124 975-1023,0 512-885	1-124 125-174 175-548
GSM 850	7	128-251	128-251
DUAL_US GSM 850/PCS 1900	8	128-251 512-810	1-124 125-424

A message sequence chart is given to show an example of band selection in case of multi-band support DUALEXT/PCS1900 (tri-band).

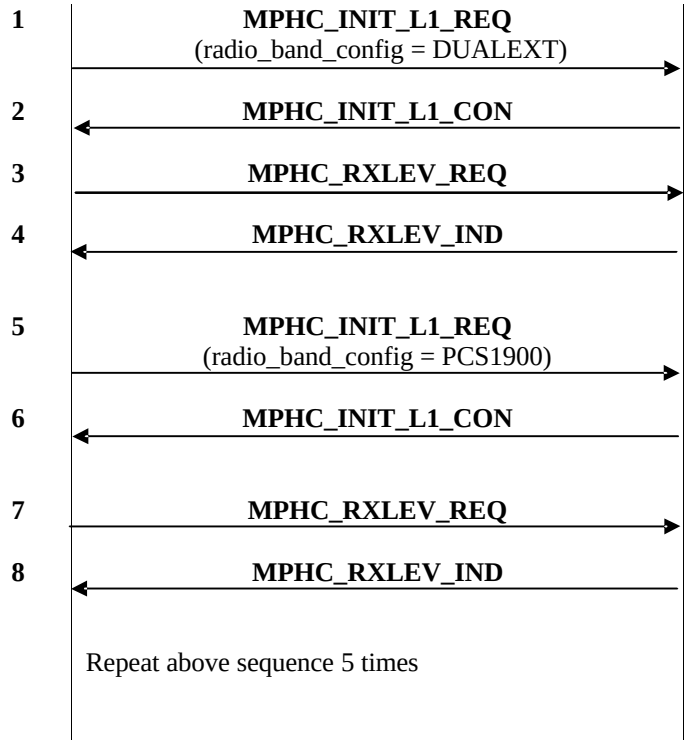


Figure 33: Dynamic band configuration of L1 for cell selection

1 An MPHC_INIT_L1_REQ message is sent by L3 to L1 to set dual extended frequency band (E-GSM / DCS 1800).
In this example we are starting from this dual band configuration since it was the last successful configuration.

2 A confirmation is then sent by L1 to L3 to confirm the setting.

3/4 A full list power measurement campaign is launched over all carriers of the DUALEXT band.

5/6 An MPHC_INIT_L1_REQ message is sent by L3 to L1 to set PCS1900 frequency band..

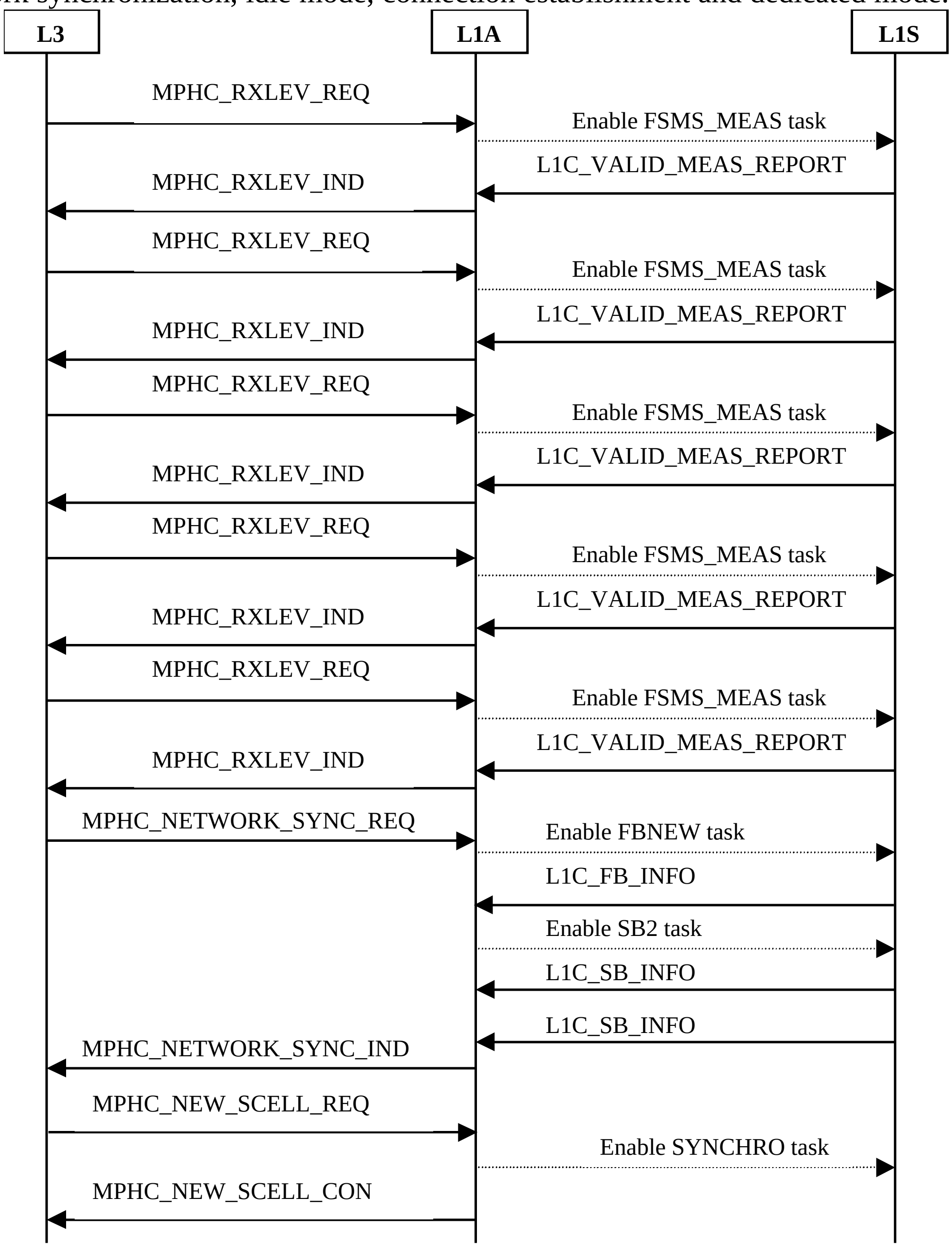
7/8 A full list power measurement campaign is launched over all carriers of the PCS1900 band.

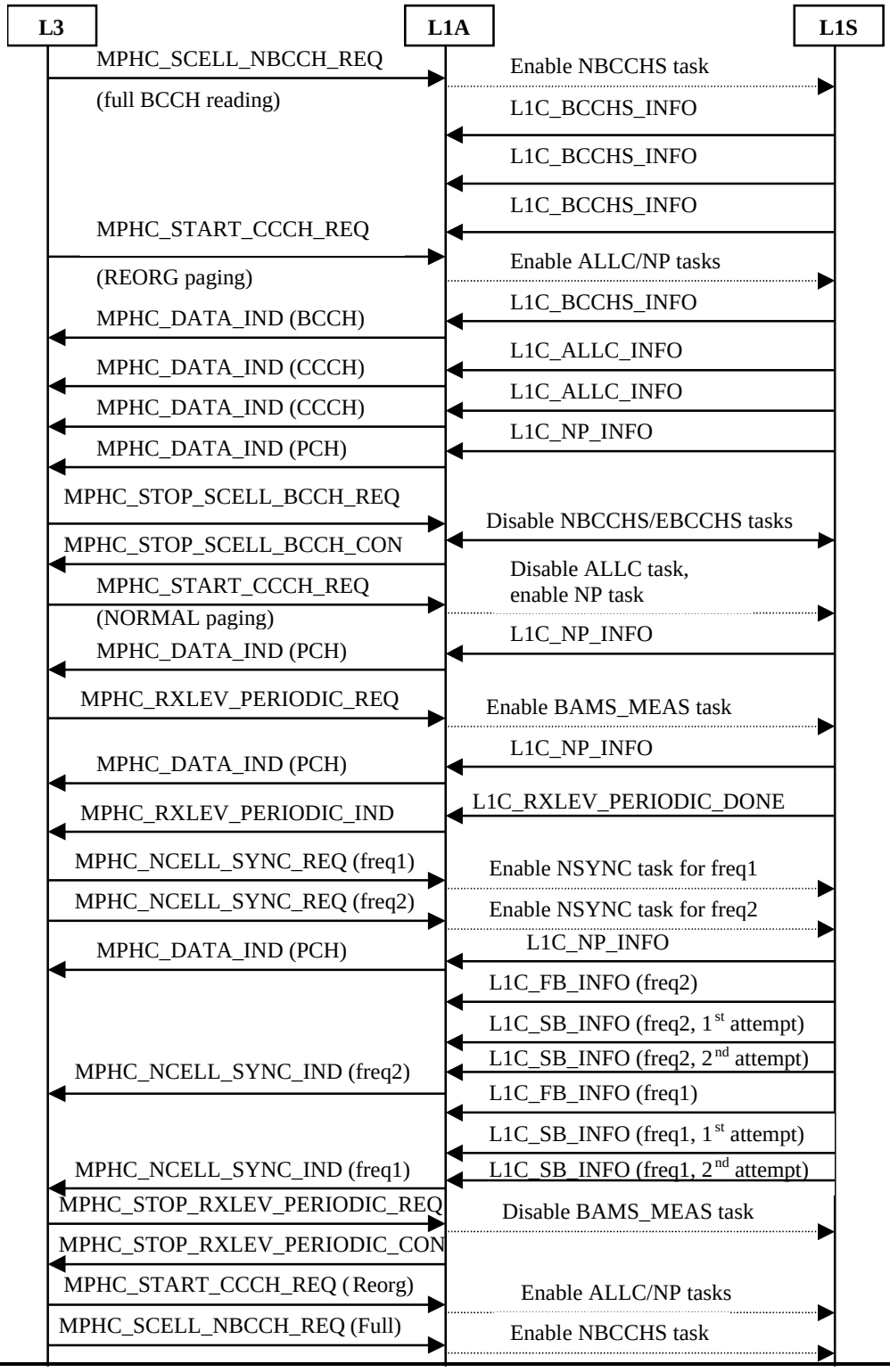
Once the ordered lists have been established for the case DUALEXT/DUAL_US (2 ordered lists required), L1 is configured for the band with the maximum power level and the MPHC_NETWORK_SYNC_REQ messages are sent for the carriers of this band. If no successful cell selection could be carried out on this band L1 is configured for the other band and MPHC_NETWORK_SYNC_REQ messages are posted for that band.
As E-GSM/DCS and GSM850/PCS bands cannot exist in parallel on a real network (due to the overlap of GSM850 DL frequencies with E-GSM uplink frequencies and the overlap of DCS DL frequencies with PCS UL frequencies) the cell selection has to be done entirely in one band before switching to the other in case cell selection was unsuccessful in the first band.

All activities have either to be completed or stopped before switching from one band to another.

Appendix A: Example of circuit switch message flow

The figure below gives an example of message flow between L3 <->L1A<->L1S for network synchronization, idle mode, connection establishment and dedicated mode.





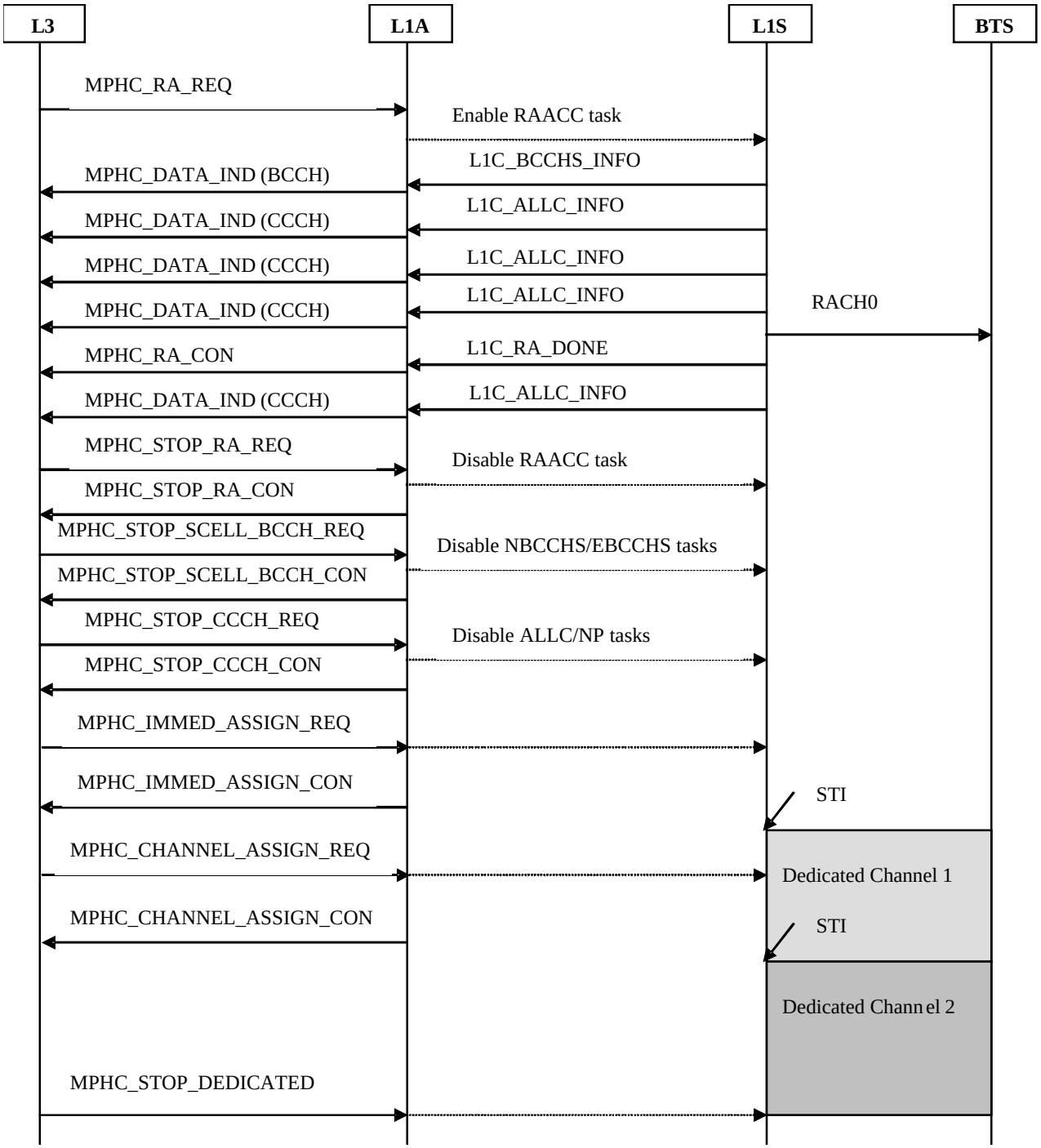


Figure 34: Example of circuit switched message flow

Appendix B : L1 message Identifier

L1 Message type structure

```
typedef NU_QUEUE   *QID;
typedef QID         SDL_PId;

typedef struct xSignalHeaderStruct *xSignalHeader;

typedef struct xSignalHeaderStruct
{
      int               SignalCode;
      xSignalHeader     Pre, Suc;
      SDL_PId           Sender;
      char              *SigP;
} xSignalHeaderRec;
```

Exemple: message creation

```
typedef struct
{
 UWORD32  fn;
 UWORD8   channel_request;
}
T_MPHC_RA_CON;    // message structure

Func()
{
   xSignalHeaderRec *msg;

    msg = os_alloc_sig(sizeof(T_MPHC_RA_CON));      // memory allocation
    msg -> SigP = msg + 1 ;                         // pointer initialization
    msg->SignalCode = MPHC_RA_CON;
    ((T_MPHC_RA_CON *)(msg->SigP))->fn = VALUE1;
    ((T_MPHC_RA_CON *)(msg->SigP))->channel_request = VALUE2;
}
```

L1 message identifier

```
#define P_L1C  0
#define P_DLL  ( P_L1C + 1 )

// Messages Test/PWRMNGT <-> L1A

#if (OP_L1_STANDALONE == 1)
  /* Message used for hardware dynamic configuration */
  #define TST_HW_CONFIG_REQ              ( ( P_L1C << 8 ) | 116)
```

```
#endif

#define TST_TEST_HW_REQ                 ( ( P_L1C << 8 ) | 1  )
#define TST_TEST_HW_CON                 ( ( P_L1C << 8 ) | 2  )
#define TST_TIMESTAMP_MSG               ( ( P_L1C << 8 ) | 3  )
#define TST_SLEEP_REQ                   ( ( P_L1C << 8 ) | 4  )


// Messages L3 <-> L1A
#define MPHC_RXLEV_REQ                  ( ( P_L1C << 8 ) | 11 )
#define MPHC_RXLEV_IND                  ( ( P_L1C << 8 ) | 12 )

#define MPHC_STOP_RXLEV_REQ             ( ( P_L1C << 8 ) | 13 )
#define MPHC_STOP_RXLEV_CON             ( ( P_L1C << 8 ) | 14 )

#define MPHC_NETWORK_SYNC_REQ           ( ( P_L1C << 8 ) | 15 )
#define MPHC_NETWORK_SYNC_IND           ( ( P_L1C << 8 ) | 16 )

#define MPHC_STOP_NETWORK_SYNC_REQ      ( ( P_L1C << 8 ) | 17 )
#define MPHC_STOP_NETWORK_SYNC_CON      ( ( P_L1C << 8 ) | 18 )

#define MPHC_NEW_SCELL_REQ              ( ( P_L1C << 8 ) | 19 )
#define MPHC_NEW_SCELL_CON              ( ( P_L1C << 8 ) | 20 )

#define MPHC_START_CCCH_REQ             ( ( P_L1C << 8 ) | 21 )
#define MPHC_STOP_CCCH_REQ              ( ( P_L1C << 8 ) | 22 )
#define MPHC_STOP_CCCH_CON              ( ( P_L1C << 8 ) | 23 )

#define MPHC_SCELL_NBCCH_REQ            ( ( P_L1C << 8 ) | 24 )
#define MPHC_SCELL_EBCCH_REQ            ( ( P_L1C << 8 ) | 25 )
#define MPHC_STOP_SCELL_BCCH_REQ        ( ( P_L1C << 8 ) | 26 )
#define MPHC_STOP_SCELL_BCCH_CON        ( ( P_L1C << 8 ) | 27 )

#define MPHC_NCELL_BCCH_REQ             ( ( P_L1C << 8 ) | 28 )
#define MPHC_NCELL_BCCH_IND             ( ( P_L1C << 8 ) | 29 )
#define MPHC_STOP_NCELL_BCCH_REQ        ( ( P_L1C << 8 ) | 30 )
#define MPHC_STOP_NCELL_BCCH_CON        ( ( P_L1C << 8 ) | 31 )

#define MPHC_NCELL_SYNC_REQ             ( ( P_L1C << 8 ) | 32 )
#define MPHC_NCELL_SYNC_IND             ( ( P_L1C << 8 ) | 33 )
#define MPHC_STOP_NCELL_SYNC_REQ        ( ( P_L1C << 8 ) | 34 )
#define MPHC_STOP_NCELL_SYNC_CON        ( ( P_L1C << 8 ) | 35 )

#define MPHC_RXLEV_PERIODIC_REQ         ( ( P_L1C << 8 ) | 36 )
#define MPHC_RXLEV_PERIODIC_IND         ( ( P_L1C << 8 ) | 37 )
#define MPHC_STOP_RXLEV_PERIODIC_REQ    ( ( P_L1C << 8 ) | 38 )
#define MPHC_STOP_RXLEV_PERIODIC_CON    ( ( P_L1C << 8 ) | 39 )

#define MPHC_CONFIG_CBCH_REQ            ( ( P_L1C << 8 ) | 40 )
#define MPHC_CBCH_SCHEDULE_REQ          ( ( P_L1C << 8 ) | 41 )
```

```
#define MPHC_CBCH_UPDATE_REQ            ( ( P_L1C << 8 ) | 42 )
#define MPHC_CBCH_INFO_REQ              ( ( P_L1C << 8 ) | 43 )
#define MPHC_STOP_CBCH_REQ              ( ( P_L1C << 8 ) | 44 )
#define MPHC_STOP_CBCH_CON              ( ( P_L1C << 8 ) | 45 )

#define MPHC_RA_REQ                     ( ( P_L1C << 8 ) | 46 )
#define MPHC_RA_CON                     ( ( P_L1C << 8 ) | 47 )

#define MPHC_STOP_RA_REQ                ( ( P_L1C << 8 ) | 48 )
#define MPHC_STOP_RA_CON                ( ( P_L1C << 8 ) | 49 )

#define MPHC_DATA_IND                   ( ( P_L1C << 8 ) | 50 )

#define MPHC_IMMED_ASSIGN_REQ           ( ( P_L1C << 8 ) | 51 )
#define MPHC_CHANNEL_ASSIGN_REQ         ( ( P_L1C << 8 ) | 52 )
#define MPHC_ASYNC_HO_REQ               ( ( P_L1C << 8 ) | 53 )
#define MPHC_SYNC_HO_REQ                ( ( P_L1C << 8 ) | 54 )
#define MPHC_PRE_SYNC_HO_REQ            ( ( P_L1C << 8 ) | 55 )
#define MPHC_PSEUDO_SYNC_HO_REQ         ( ( P_L1C << 8 ) | 56 )
#define MPHC_STOP_DEDICATED_REQ         ( ( P_L1C << 8 ) | 57 )
#define MPHC_STOP_DEDICATED_CON         ( ( P_L1C << 8 ) | 128 )

#define MPHC_CHANGE_FREQUENCY_CON       ( ( P_L1C << 8 ) | 58 )
#define MPHC_ASYNC_HO_CON               ( ( P_L1C << 8 ) | 59 )
#define MPHC_CHANNEL_ASSIGN_CON         ( ( P_L1C << 8 ) | 60 )
#define MPHC_CHANNEL_MODE_MODIFY_CON    ( ( P_L1C << 8 ) | 61 )
#define MPHC_HANDOVER_FAIL_CON          ( ( P_L1C << 8 ) | 62 )
#define MPHC_IMMED_ASSIGN_CON           ( ( P_L1C << 8 ) | 63 )
#define MPHC_PRE_SYNC_HO_CON            ( ( P_L1C << 8 ) | 64 )
#define MPHC_SET_CIPHERING_CON          ( ( P_L1C << 8 ) | 65 )
#define MPHC_SYNC_HO_CON                ( ( P_L1C << 8 ) | 66 )
#define MPHC_TA_FAIL_IND                ( ( P_L1C << 8 ) | 67 )

#define MPHC_HANDOVER_FINISHED          ( ( P_L1C << 8 ) | 68 )

#define MPHC_CHANGE_FREQUENCY           ( ( P_L1C << 8 ) | 69 )
#define MPHC_CHANNEL_MODE_MODIFY_REQ    ( ( P_L1C << 8 ) | 70 )
#define MPHC_HANDOVER_FAIL_REQ          ( ( P_L1C << 8 ) | 71 )
#define MPHC_SET_CIPHERING_REQ          ( ( P_L1C << 8 ) | 72 )

#define MPHC_MEAS_REPORT                ( ( P_L1C << 8 ) | 73 )
#define MPHC_UPDATE_BA_LIST             ( ( P_L1C << 8 ) | 74 )

#define MPHC_NCELL_FB_SB_READ           ( ( P_L1C << 8 ) | 75 )
#define MPHC_NCELL_SB_READ              ( ( P_L1C << 8 ) | 76 )
#define MPHC_NCELL_BCCH_READ            ( ( P_L1C << 8 ) | 77 )


// Messages L1S -> L1A
```

```
#define L1C_VALID_MEAS_INFO             ( ( P_L1C << 8 ) | 80 )
#define L1C_FB_INFO                     ( ( P_L1C << 8 ) | 81 )
#define L1C_SB_INFO                     ( ( P_L1C << 8 ) | 82 )
#define L1C_SBCONF_INFO                 ( ( P_L1C << 8 ) | 83 )
#define L1C_BCCHS_INFO                  ( ( P_L1C << 8 ) | 84 )
#define L1C_BCCHN_INFO                  ( ( P_L1C << 8 ) | 85 )
#define L1C_NP_INFO                     ( ( P_L1C << 8 ) | 86 )
#define L1C_EP_INFO                     ( ( P_L1C << 8 ) | 87 )
#define L1C_ALLC_INFO                   ( ( P_L1C << 8 ) | 88 )
#define L1C_RXLEV_PERIODIC_DONE         ( ( P_L1C << 8 ) | 89 )
#define L1C_CB_INFO                     ( ( P_L1C << 8 ) | 90 )
#define L1C_RA_DONE                     ( ( P_L1C << 8 ) | 91 )
#define L1C_SACCH_INFO                  ( ( P_L1C << 8 ) | 92 )
#define L1C_MEAS_DONE                   ( ( P_L1C << 8 ) | 93 )
#define L1C_DEDIC_DONE                  ( ( P_L1C << 8 ) | 94 )
#define L1C_HANDOVER_FINISHED           ( ( P_L1C << 8 ) | 95 )
#define L1C_REDEF_DONE                  ( ( P_L1C << 8 ) | 96 )
#define L1C_STOP_DEDICATED_DONE         ( ( P_L1C << 8 ) | 129 )

// Messages O&M <-> L1A
#define OML1_CLOSE_TCH_LOOP_REQ         ( ( P_L1C << 8 ) | 97 )
#define OML1_OPEN_TCH_LOOP_REQ          ( ( P_L1C << 8 ) | 98 )
#define OML1_START_DAI_TEST_REQ         ( ( P_L1C << 8 ) | 99 )
#define OML1_STOP_DAI_TEST_REQ          ( ( P_L1C << 8 ) | 100 )

#define OML1_CLOSE_TCH_LOOP_CON         ( ( P_L1C << 8 ) | 101 )
#define OML1_OPEN_TCH_LOOP_CON          ( ( P_L1C << 8 ) | 102 )
#define OML1_START_DAI_TEST_CON         ( ( P_L1C << 8 ) | 103 )
#define OML1_STOP_DAI_TEST_CON          ( ( P_L1C << 8 ) | 104 )


// Message Custom I/F <- L1S
#define CST_ADC_RESULT                  ( ( P_L1C << 8 ) | 105 )


// Messages for trace
#define L1_STATS_REQ                    ( ( P_L1C << 8 ) | 107 )
#define L1_DUMMY_FOR_SIM                ( ( P_L1C << 8 ) | 108 )

#define TRACE_DSP_DEBUG                 ( ( P_L1C << 8 ) | 106 )
#define TRACE_CONFIG                    ( ( P_L1C << 8 ) | 123 )
#define TRACE_CONDENSED_PDTCH           ( ( P_L1C << 8 ) | 124 )
#define TRACE_INFO                      ( ( P_L1C << 8 ) | 125 )
#define QUICK_TRACE                     ( ( P_L1C << 8 ) | 126 )
#define TRACE_DSP_AMR_DEBUG             ( ( P_L1C << 8 ) | 127 )

#define MPHC_NETWORK_LOST_IND           ( ( P_L1C << 8 ) | 110 )

// Messages MMI <-> L1A
#define MMI_ADC_REQ                     ( ( P_L1C << 8 ) | 111 )
```

```
#define MMI_STOP_ADC_REQ                ( ( P_L1C << 8 ) | 112 )
#define MMI_STOP_ADC_CON                ( ( P_L1C << 8 ) | 113 )

#define L1_TEST_HW_INFO                 ( ( P_L1C << 8 ) | 119 )

// Multi-band selection E-GSM900/DCS1800/PCS1900/GSM850
  #define MPHC_INIT_L1_REQ             ( ( P_L1C << 8 ) | 114 )
  #define MPHC_INIT_L1_CON             ( ( P_L1C << 8 ) | 115 )

// MEssage RR -> L1A for Enhanced meas.
#if (L1_12NEIGH ==1)
#define  MPHC_NCELL_LIST_SYNC_REQ        ( ( P_L1C << 8 ) | 122 )
#endif

// Messages L2 <-> L1A
#define PH_DATA_IND                     ( ( P_DLL << 8 ) | 109   )
```