



1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23

## Linux-based Mobile Phone Middleware

### Application Programming Interface

### Circuit-Switched Communication Service

Document: CELF\_MPP\_CS\_D\_2.2.6\_20060706

WARNING: This is a working draft for review only, it is NOT a published specification of the CE Linux Forum. It is likely that further substantial changes will be made in the course of review and issue resolution. Send comments on this version to:

[MppApiComments@tree.celinuxforum.org](mailto:MppApiComments@tree.celinuxforum.org)

24 **Revision History**

Revision	Comment	Reviewer	Editor	Date
2.2	Initial	F2F meeting	NEC/Panasonic	05/09/28
2.2.1	Editorial Changes		AK	05/10/03
2.2.1a	NEC comments	NEC	AK	05/11/08
2.2.2	Review comments	Sharp	AK	05/11/20
2.2.2a	Template Change		AK	05/11/21
2.2.3	Appld ref removed Extract Common		AK	05/12/28
2.2.4	Review changes		AK	06/06/08
2.2.5	Clarifications Reformatting	NEC	AK	06/06/15
2.2.6 (FR3)	Minor reformatting, capitalization		Scott Preece	06/07/06

25

26 **0. INTRODUCTION..... 15**

27 0.1 REFERENCES ..... 15

28 0.1.1 Normative ..... 15

29 0.1.2 Informative..... 15

30 **1. PRIMITIVES..... 16**

31 1.1 CONSTANTS..... 16

32 1.1.1 Line type (int)..... 16

33 1.1.2 Dial Number ..... 16

34 1.1.3 TAF address..... 16

35 1.1.4 Additional Service..... 16

36 1.2 ENUMS ..... 18

37 1.2.1 Voice communication status (CelfMpCsComStatus) ..... 18

38 1.2.2 Forwarding result (CelfMpCsFwResult) ..... 19

39 1.2.3 Forwarding result details (CelfMpCsFwError)..... 19

40 1.2.4 Bearer Type (CelfMpCsBtype)..... 19

41 1.2.5 Call Reference Status (CelfMpCsCallRefStatus)..... 19

42 1.2.6 Call Status (CelfMpCsChan) ..... 20

43 1.2.7 Existence of continuation data (CelfMpCsContData) ..... 20

44 1.2.8 Busy Tone sound flag (CelfMpCsBusyTone)..... 20

45 1.2.9 Cause of Calling Line Identity (CLI) not available (CelfMpCsNoCLI)..... 20

46 1.2.10 Dial number / Redirect number display indicator (CelfMpCsPrsntInd)..... 21

47 1.2.11 Signal information (CelfMpCsSignal) ..... 21

48 1.2.12 Originating Number notification (CelfMpCsNotice) ..... 21

49 1.2.13 Line status (CelfMpCsLineStatus) ..... 21

50 1.2.14 Normal and emergency originating restriction (CelfMpCsLineRestrictData) ..... 21

51 1.2.15 Receive level (CelfMpCsRSSILevel)..... 22

52 1.2.16 Area status information (CelfMpCsLineCvrStatus) ..... 22

53 1.2.17 RRC mode (CelfMpCsLineRRCMode)..... 22

54 1.2.18 Network identification information (CelfMpCsLineNetwork)..... 22

55 1.2.19 Service status (CelfMpLineSrvStatus)..... 22

56 1.2.20 Restriction status (CelfMpCsLineRestrict)..... 22

57 1.2.21 Identifying flag (CelfMpCsFlag)..... 23

58 1.2.22 Notification Set (CelfMpCsNotifySet) ..... 23

59 1.2.23 Event Structure Category..... 23

60 1.2.24 Event Structure Subtype..... 23

61 1.2.25 DCF Event Set (CelfMpCsDCFSet)..... 23

62 1.2.26 Voice message (CelfMpCsRecMsg) ..... 23

63 1.2.27 Off Hook Option (CelfMpCsOffHk)..... 24

64 1.2.28 64K/AV Communication (CelfMpCsUDComStatus)..... 24

65 1.2.29 AV Communication (CelfMpAVComStatus) ..... 24

66 1.2.30 Receive Types (CelfMpCsRcvType) ..... 25

67 1.2.31 Line Monitoring (CelfMpCsMtype) ..... 25

68 1.2.32 Coverage Indicators (CelfMpCsCoverage) ..... 25

69 1.2.33 Incoming Call Selection (CelfMpCallSelect)..... 25

70 1.2.34 Optional Registered Number (CelfMpRegNum) ..... 25

71 1.2.35 Service Data (CelfMpCsSrvData)..... 25

72 1.2.36 Reconnection Tone (CelfMpCsReconnectionTone) ..... 25

73 1.2.37 Noise Canceling (CelfMpCsNoiseCancel)..... 26

74 1.2.38 Call Quality Alarm (CelfMpCsCallQualAlarm) ..... 26

75 1.2.39 Connection Priority Setting (CelfMpCsHiPrioCom) ..... 26

76 1.2.40 Message Sound Settings (CelfMpCsVmSound)..... 26

77 1.2.41 Incoming Call Auto Receive (CelfMpCsAutoRcv) ..... 26

Classification: **Circuit-Switched Service**

78	1.3	DATA TYPES AND STRUCTURES .....	27
79	1.3.1	<i>Circuit switched status notification event structure</i> .....	27
80	1.3.2	<i>Call duration notification event structure</i> .....	27
81	1.3.3	<i>Disconnection cause notification event structure</i> .....	27
82	1.3.4	<i>Disconnection cause information structure</i> .....	27
83	1.3.5	<i>Forwarding result notification event structure</i> .....	28
84	1.3.6	<i>Forwarding result structure (CelfMpCsFwResult)</i> .....	28
85	1.3.7	<i>Off-hook transmission timeout event structure</i> .....	28
86	1.3.8	<i>Connection Destination Information (CelfMpConnectInfo)</i> .....	28
87	1.3.9	<i>Connection Request (CelfMpCsConReq)</i> .....	29
88	1.3.10	<i>Redirection number</i> .....	29
89	1.3.11	<i>Channel Number Information (CelfMpCsChanNum)</i> .....	29
90	1.3.12	<i>Channel not in use Flag</i> .....	29
91	1.3.13	<i>DCF Event Structure</i> .....	29
92	1.3.14	<i>Line status change notification event structure</i> .....	30
93	1.3.15	<i>Restriction display information structure (CelfMpCsResChgInf)</i> .....	30
94	1.3.16	<i>Receive level change notification event structure</i> .....	30
95	1.3.17	<i>Line Status structure (CelfMpCsAreaRefChgInf)</i> .....	30
96	1.3.18	<i>Additional service data structure (CelfMpCsAddSrvData)</i> .....	31
97	1.3.19	<i>Response Message Data Structure (CelfMpCsResponseMsgData)</i> .....	31
98	1.3.20	<i>Line Status Extension (CelfMpCsLineStatusEx)</i> .....	32
99	1.3.21	<i>Number of stored messages (CelfMpCsVMNum)</i> .....	32
100	1.3.22	<i>Date Format Structure (CelfMpCsDate)</i> .....	32
101	1.3.23	<i>Dial Buffer (CelfMpCsDialBuffer)</i> .....	32
102	1.3.24	<i>Dial Buffer Length (CelfMpCsDialLen)</i> .....	32
103	1.3.25	<i>Multi Party Operation (CelfMpCsMop)</i> .....	32
104	1.3.26	<i>Timer Value (CelfMpCsTimer)</i> .....	32
105	1.4	EVENTS TYPE .....	33
106	1.4.1	<i>DCF Event Type</i> .....	33
107	1.4.2	<i>CCP Notification type</i> .....	33
108	1.4.3	<i>Notification type</i> .....	35
109	1.4.4	<i>Restriction status</i> .....	35
110	1.5	STATUS CODES (CELFMPSTATUS) .....	36
111	<b>2.</b>	<b>START NOTIFICATION</b> .....	<b>37</b>
112	2.1	SYMBOL: CELF_MP_CS_NOTIFICATION_START .....	37
113	2.1.1	<i>Syntax</i> .....	37
114	2.1.2	<i>Argument</i> .....	37
115	2.1.3	<i>Return Value</i> .....	38
116	2.1.4	<i>Include File</i> .....	38
117	2.1.5	<i>Functional Description</i> .....	38
118	<b>3.</b>	<b>STOP NOTIFICATION</b> .....	<b>39</b>
119	3.1	SYMBOL: CELF_MP_CS_NOTIFICATION_STOP .....	39
120	3.1.1	<i>Syntax</i> .....	39
121	3.1.2	<i>Argument</i> .....	39
122	3.1.3	<i>Return Value</i> .....	39
123	3.1.4	<i>Include File</i> .....	40
124	3.1.5	<i>Functional Description</i> .....	40
125	<b>4.</b>	<b>GET VOICE COMMUNICATION STATUS</b> .....	<b>41</b>
126	4.1	SYMBOL: CELF_MP_CS_GET_COM_STATUS .....	41
127	4.1.1	<i>Syntax</i> .....	41
128	4.1.2	<i>Argument</i> .....	41
129	4.1.3	<i>Return Value</i> .....	41

130	4.1.4	<i>Include File</i> .....	41
131	4.1.5	<i>Functional Description</i> .....	41
132	<b>5.</b>	<b>GET CONNECTION INFORMATION TO OTHER PARTY</b> .....	<b>42</b>
133	5.1	SYMBOL: CELF_MP_CS_GET_CON_INFO_REF .....	42
134	5.1.1	<i>Syntax</i> .....	42
135	5.1.2	<i>Argument</i> .....	42
136	5.1.3	<i>Return Value</i> .....	42
137	5.1.4	<i>Include File</i> .....	43
138	5.1.5	<i>Functional Description</i> .....	43
139	<b>6.</b>	<b>GET CALL DURATION</b> .....	<b>44</b>
140	6.1	SYMBOL: CELF_MP_CS_GET_CALL_DURATION .....	44
141	6.1.1	<i>Syntax</i> .....	44
142	6.1.2	<i>Argument</i> .....	44
143	6.1.3	<i>Return Value</i> .....	44
144	6.1.4	<i>Include File</i> .....	44
145	6.1.5	<i>Functional Description</i> .....	44
146	<b>7.</b>	<b>OFF-HOOK NOTIFICATION</b> .....	<b>46</b>
147	7.1	SYMBOL: CELF_MP_CS_NOTIFICATION_OFF_HOOK.....	46
148	7.1.1	<i>Syntax</i> .....	46
149	7.1.2	<i>Argument</i> .....	46
150	7.1.3	<i>Return Value</i> .....	46
151	7.1.4	<i>Include File</i> .....	47
152	7.1.5	<i>Functional Description</i> .....	47
153	<b>8.</b>	<b>DISCONNECT</b> .....	<b>48</b>
154	8.1	SYMBOL: CELF_MP_CS_DISCONNECT.....	48
155	8.1.1	<i>Syntax</i> .....	48
156	8.1.2	<i>Argument</i> .....	48
157	8.1.3	<i>Return Value</i> .....	48
158	8.1.4	<i>Include File</i> .....	48
159	8.1.5	<i>Functional Description</i> .....	48
160	<b>9.</b>	<b>DIAL</b> .....	<b>50</b>
161	9.1	SYMBOL: CELF_MP_CS_DIAL.....	50
162	9.1.1	<i>Syntax</i> .....	50
163	9.1.2	<i>Argument</i> .....	50
164	9.1.3	<i>Return Value</i> .....	51
165	9.1.4	<i>Include File</i> .....	51
166	9.1.5	<i>Functional Description</i> .....	51
167	<b>10.</b>	<b>DIAL COMPLETE</b> .....	<b>52</b>
168	10.1	SYMBOL: CELF_MP_CS_DIAL_END .....	52
169	10.1.1	<i>Syntax</i> .....	52
170	10.1.2	<i>Argument</i> .....	52
171	10.1.3	<i>Return Value</i> .....	52
172	10.1.4	<i>Include File</i> .....	52
173	10.1.5	<i>Functional Description</i> .....	52
174	<b>11.</b>	<b>RESPONSE TO INCOMING CALL</b> .....	<b>54</b>
175	11.1	SYMBOL: CELF_MP_CS_CALL_RCV .....	54
176	11.1.1	<i>Syntax</i> .....	54
177	11.1.2	<i>Argument</i> .....	54

178	11.1.3	<i>Return Value</i> .....	54
179	11.1.4	<i>Include File</i> .....	54
180	11.1.5	<i>Functional Description</i> .....	54
181	<b>12.</b>	<b>FORWARD INCOMING CALL</b> .....	<b>56</b>
182	12.1	SYMBOL: CELF_MP_CS_CALL_FORWARD .....	56
183	12.1.1	<i>Syntax</i> .....	56
184		<i>Argument</i> .....	56
185	12.1.2	56	
186	12.1.3	<i>Return Value</i> .....	56
187	12.1.4	<i>Include File</i> .....	56
188	12.1.5	<i>Functional Description</i> .....	56
189	<b>13.</b>	<b>FORWARD TO VOICE MAIL SYSTEM</b> .....	<b>57</b>
190	13.1	SYMBOL: CELF_MP_CS_CALL_FORWARD_VOICE_MSG .....	57
191	13.1.1	<i>Syntax</i> .....	57
192		<i>Argument</i> .....	57
193	13.1.2	57	
194	13.1.3	<i>Return Value</i> .....	57
195	13.1.4	<i>Include File</i> .....	57
196	13.1.5	<i>Functional Description</i> .....	57
197	<b>14.</b>	<b>CALL HOLD</b> .....	<b>59</b>
198	14.1	SYMBOL: CELF_MP_CS_CALL_HOLD .....	59
199	14.1.1	<i>Syntax</i> .....	59
200		<i>Argument</i> .....	59
201	14.1.2	59	
202	14.1.3	<i>Return Value</i> .....	59
203	14.1.4	<i>Include File</i> .....	59
204	14.1.5	<i>Functional Description</i> .....	59
205	<b>15.</b>	<b>CALL REJECT</b> .....	<b>61</b>
206	15.1	SYMBOL: CELF_MP_CS_CALL_REJECT .....	61
207	15.1.1	<i>Syntax</i> .....	61
208		<i>Argument</i> .....	61
209	15.1.2	61	
210	15.1.3	<i>Return Value</i> .....	61
211	15.1.4	<i>Include File</i> .....	61
212	15.1.5	<i>Functional Description</i> .....	61
213	<b>16.</b>	<b>MULTI PARTY CALL</b> .....	<b>63</b>
214	16.1	SYMBOL: CELF_MP_CS_MP_CALL .....	63
215	16.1.1	<i>Syntax</i> .....	63
216		<i>Argument</i> .....	63
217	16.1.2	63	
218	16.1.3	<i>Return Value</i> .....	64
219	16.1.4	<i>Include File</i> .....	64
220	16.1.5	<i>Functional Description</i> .....	64
221	<b>17.</b>	<b>ON-HOOK ORIGINATING</b> .....	<b>66</b>
222	17.1	SYMBOL: CELF_MP_CS_ORIGINATING_ON_HOOK .....	66
223	17.1.1	<i>Syntax</i> .....	66
224		<i>Argument</i> .....	66
225	17.1.2	66	
226	17.1.3	<i>Return Value</i> .....	66

227	17.1.4	<i>Include File</i> .....	66
228	17.1.5	<i>Functional Description</i> .....	67
229	<b>18.</b>	<b>GET CALL REFERENCE</b> .....	<b>68</b>
230	18.1	SYMBOL: CELF_MP_CS_GET_CALL_REFERENCE.....	68
231	18.1.1	<i>Syntax</i> .....	68
232	18.1.2	<i>Argument</i> .....	68
233	18.1.3	<i>Return Value</i> .....	68
234	18.1.4	<i>Include File</i> .....	68
235	18.1.5	<i>Functional Description</i> .....	68
236	<b>19.</b>	<b>START DCF MESSAGE NOTIFICATION</b> .....	<b>70</b>
237	19.1	SYMBOL: CELF_MP_CS_DCF_NOTIFICATION_START.....	70
238	19.1.1	<i>Syntax</i> .....	70
239	19.1.2	<i>Argument</i> .....	70
240	19.1.3	<i>Return Value</i> .....	71
241	19.1.4	<i>Include File</i> .....	71
242	19.1.5	<i>Functional Description</i> .....	71
243	<b>20.</b>	<b>STOP DCF MESSAGE NOTIFICATION</b> .....	<b>73</b>
244	20.1	SYMBOL: CELF_MP_CS_DCF_NOTIFICATION_STOP.....	73
245	20.1.1	<i>Syntax</i> .....	73
246	20.1.2	<i>Argument</i> .....	73
247	20.1.3	<i>Return Value</i> .....	73
248	20.1.4	<i>Include File</i> .....	74
249	20.1.5	<i>Functional Description</i> .....	74
250	<b>21.</b>	<b>VOICE MESSAGE NOTIFICATION</b> .....	<b>75</b>
251	21.1	SYMBOL: CELF_MP_CS_VOICE_MSG_NOTIFY.....	75
252	21.1.1	<i>Syntax</i> .....	75
253		<i>Argument</i> .....	75
254	21.1.2	75	75
255	21.1.3	<i>Return Value</i> .....	75
256	21.1.4	<i>Include File</i> .....	75
257	21.1.5	<i>Functional Description</i> .....	75
258	<b>22.</b>	<b>HOLD TONE START</b> .....	<b>76</b>
259	22.1	SYMBOL: CELF_MP_CS_HOLD_TONE_START.....	76
260	22.1.1	<i>Syntax</i> .....	76
261		<i>Argument</i> .....	76
262	22.1.2	76	76
263	22.1.3	<i>Return Value</i> .....	76
264	22.1.4	<i>Include File</i> .....	76
265	22.1.5	<i>Functional Description</i> .....	76
266	<b>23.</b>	<b>HOLD TONE STOP</b> .....	<b>77</b>
267	23.1	SYMBOL: CELF_MP_CS_HOLD_TONE_STOP.....	77
268	23.1.1	<i>Syntax</i> .....	77
269		<i>Argument</i> .....	77
270	23.1.2	77	77
271	23.1.3	<i>Return Value</i> .....	77
272	23.1.4	<i>Include File</i> .....	77
273	23.1.5	<i>Functional Description</i> .....	77
274	<b>24.</b>	<b>GET 64K / AV COMMUNICATION STATUS</b> .....	<b>78</b>

Classification: *Circuit-Switched Service*

275	24.1	SYMBOL: CELF_MP_CS_GET_UD_COM_STAT .....	78
276	24.1.1	<i>Syntax</i> .....	78
277		<i>Argument</i> .....	78
278	24.1.2	<i>Return Value</i> .....	78
279	24.1.3	<i>Include File</i> .....	78
280	24.1.4	<i>Functional Description</i> .....	78
281	24.1.5	<i>Functional Description</i> .....	78
282	<b>25.</b>	<b>GET INTERNAL/EXTERNAL AV COMMUNICATION STATUS.....</b>	<b>79</b>
283	25.1	SYMBOL: CELF_MP_CS_GET_AV_COM_STAT .....	79
284	25.1.1	<i>Syntax</i> .....	79
285	25.1.2	<i>Argument</i> .....	79
286	25.1.3	<i>Return Value</i> .....	79
287	25.1.4	<i>Include File</i> .....	80
288	25.1.5	<i>Functional Description</i> .....	80
289	<b>26.</b>	<b>GET COMMUNICATION STATUS.....</b>	<b>81</b>
290	26.1	SYMBOL: CELF_MP_CS_GET_COM_STAT .....	81
291	26.1.1	<i>Syntax</i> .....	81
292	26.1.2	<i>Argument</i> .....	81
293	26.1.3	<i>Return Value</i> .....	82
294	26.1.4	<i>Include File</i> .....	82
295	26.1.5	<i>Functional Description</i> .....	82
296	<b>27.</b>	<b>START LINE STATUS NOTIFICATION.....</b>	<b>83</b>
297	27.1	SYMBOL: CELF_MP_CS_LINE_STATUS_NOTIFICATION_START.....	83
298	27.1.1	<i>Syntax</i> .....	83
299	27.1.2	<i>Argument</i> .....	83
300	27.1.3	<i>Return Value</i> .....	84
301	27.1.4	<i>Include File</i> .....	84
302	27.1.5	<i>Functional Description</i> .....	84
303	<b>28.</b>	<b>STOP LINE STATUS NOTIFICATION .....</b>	<b>85</b>
304	28.1	SYMBOL: CELF_MP_CS_LINE_STATUS_NOTIFICATION_STOP .....	85
305	28.1.1	<i>Syntax</i> .....	85
306	28.1.2	<i>Argument</i> .....	85
307	28.1.3	<i>Return Value</i> .....	85
308	28.1.4	<i>Include File</i> .....	86
309	28.1.5	<i>Functional Description</i> .....	86
310	<b>29.</b>	<b>GET RECEPTION LEVEL .....</b>	<b>87</b>
311	29.1	SYMBOL: CELF_MP_CS_GET_RECEPTION_LEVEL.....	87
312	29.1.1	<i>Syntax</i> .....	87
313	29.1.2	<i>Argument</i> .....	87
314	29.1.3	<i>Return Value</i> .....	87
315	29.1.4	<i>Include File</i> .....	87
316	29.1.5	<i>Functional Description</i> .....	87
317	<b>30.</b>	<b>GET LINE STATUS .....</b>	<b>88</b>
318	30.1	SYMBOL: CELF_MP_CS_GET_LINE_STATUS .....	88
319	30.1.1	<i>Syntax</i> .....	88
320	30.1.2	<i>Argument</i> .....	88
321	30.1.3	<i>Return Value</i> .....	88
322	30.1.4	<i>Include File</i> .....	88
323	30.1.5	<i>Functional Description</i> .....	88



324	<b>31. GET COVERAGE STATUS .....</b>	<b>89</b>
325	31.1 SYMBOL: CELF_MP_CS_GET_COVERAGE_STATUS .....	89
326	31.1.1 <i>Syntax</i> .....	89
327	31.1.2 <i>Argument</i> .....	89
328	31.1.3 <i>Return Value</i> .....	89
329	31.1.4 <i>Include File</i> .....	89
330	31.1.5 <i>Functional Description</i> .....	90
331	<b>32. GET VOICE MAIL INFORMATION .....</b>	<b>91</b>
332	32.1 SYMBOL: CELF_MP_CS_GET_VM_INFO .....	91
333	32.1.1 <i>Syntax</i> .....	91
334	32.1.2 <i>Argument</i> .....	91
335	32.1.3 <i>Return Value</i> .....	91
336	32.1.4 <i>Include File</i> .....	91
337	32.1.5 <i>Functional Description</i> .....	91
338	<b>33. SET VOICE MAIL INFORMATION .....</b>	<b>92</b>
339	33.1 SYMBOL: CELF_MP_CS_SET_VM_INFO .....	92
340	33.1.1 <i>Syntax</i> .....	92
341	33.1.2 <i>Argument</i> .....	92
342	33.1.3 <i>Return Value</i> .....	92
343	33.1.4 <i>Include File</i> .....	92
344	33.1.5 <i>Functional Description</i> .....	92
345	<b>34. GET CALL SELECTION .....</b>	<b>93</b>
346	34.1 SYMBOL: CELF_MP_CS_GET_CALL_SELECT .....	93
347	34.1.1 <i>Syntax</i> .....	93
348	34.1.2 <i>Argument</i> .....	93
349	34.1.3 <i>Return Value</i> .....	93
350	34.1.4 <i>Include File</i> .....	93
351	34.1.5 <i>Functional Description</i> .....	93
352	<b>35. SET CALL SELECTION .....</b>	<b>94</b>
353	35.1 SYMBOL: CELF_MP_CS_SET_CALL_SELECT .....	94
354	35.1.1 <i>Syntax</i> .....	94
355	35.1.2 <i>Argument</i> .....	94
356	35.1.3 <i>Return Value</i> .....	94
357	35.1.4 <i>Include File</i> .....	94
358	35.1.5 <i>Functional Description</i> .....	94
359	<b>36. SET SERVICE INFORMATION .....</b>	<b>95</b>
360	36.1 SYMBOL: CELF_MP_CS_SET_SERVICE_INFO .....	95
361	36.1.1 <i>Syntax</i> .....	95
362	36.1.2 <i>Argument</i> .....	95
363	36.1.3 <i>Return Value</i> .....	95
364	36.1.4 <i>Include File</i> .....	95
365	36.1.5 <i>Functional Description</i> .....	95
366	<b>37. GET SERVICE INFORMATION .....</b>	<b>97</b>
367	37.1 SYMBOL: CELF_MP_CS_GET_SERVICE_INFO .....	97
368	37.1.1 <i>Syntax</i> .....	97
369	37.1.2 <i>Argument</i> .....	97
370	37.1.3 <i>Return Value</i> .....	97
371	37.1.4 <i>Include File</i> .....	97
372	37.1.5 <i>Functional Description</i> .....	97

373	<b>38. DELETE SERVICE INFORMATION.....</b>	<b>99</b>
374	38.1 SYMBOL: CELF_MP_CS_DEL_SERVICE_INFO .....	99
375	38.1.1 <i>Syntax</i> .....	99
376	38.1.2 <i>Argument</i> .....	99
377	38.1.3 <i>Return Value</i> .....	99
378	38.1.4 <i>Include File</i> .....	99
379	38.1.5 <i>Functional Description</i> .....	99
380	<b>39. REMOVE SERVICE INFORMATION.....</b>	<b>100</b>
381	39.1 SYMBOL: CELF_MP_CS_REMOVE_ALL_SERVICE_INFO .....	100
382	39.1.1 <i>Syntax</i> .....	100
383	39.1.2 <i>Argument</i> .....	100
384	39.1.3 <i>Return Value</i> .....	100
385	39.1.4 <i>Include File</i> .....	100
386	39.1.5 <i>Functional Description</i> .....	100
387	<b>40. SET RESPONSE MESSAGE SETTINGS .....</b>	<b>101</b>
388	40.1 SYMBOL: CELF_MP_CS_SET_RESP_MSG .....	101
389	40.1.1 <i>Syntax</i> .....	101
390	40.1.2 <i>Argument</i> .....	101
391	40.1.3 <i>Return Value</i> .....	101
392	40.1.4 <i>Include File</i> .....	101
393	40.1.5 <i>Functional Description</i> .....	101
394	<b>41. GET RESPONSE MESSAGE SETTINGS .....</b>	<b>103</b>
395	41.1 SYMBOL: CELF_MP_CS_GET_RESP_MSG.....	103
396	41.1.1 <i>Syntax</i> .....	103
397	41.1.2 <i>Argument</i> .....	103
398	41.1.3 <i>Return Value</i> .....	103
399	41.1.4 <i>Include File</i> .....	103
400	41.1.5 <i>Functional Description</i> .....	103
401	<b>42. DELETE RESPONSE MESSAGE SETTINGS.....</b>	<b>105</b>
402	42.1 SYMBOL: CELF_MP_CS_DEL_RESP_MSG.....	105
403	42.1.1 <i>Syntax</i> .....	105
404	42.1.2 <i>Argument</i> .....	105
405	42.1.3 <i>Return Value</i> .....	105
406	42.1.4 <i>Include File</i> .....	105
407	42.1.5 <i>Functional Description</i> .....	105
408	<b>43. REMOVE ALL RESPONSE MESSAGE SETTINGS.....</b>	<b>106</b>
409	43.1 SYMBOL: CELF_MP_CS_REMOVE_ALL_RESP_MSG .....	106
410	43.1.1 <i>Syntax</i> .....	106
411	43.1.2 <i>Argument</i> .....	106
412	43.1.3 <i>Return Value</i> .....	106
413	43.1.4 <i>Include File</i> .....	106
414	43.1.5 <i>Functional Description</i> .....	106
415	<b>44. SET RECONNECTION TONE .....</b>	<b>107</b>
416	44.1 SYMBOL: CELF_MP_CS_SET_RECONNECTION_TONE .....	107
417	44.1.1 <i>Syntax</i> .....	107
418	44.1.2 <i>Argument</i> .....	107
419	44.1.3 <i>Return Value</i> .....	107
420	44.1.4 <i>Include File</i> .....	107
421	44.1.5 <i>Functional Description</i> .....	107

422	<b>45. GET RECONNECTION TONE .....</b>	<b>108</b>
423	45.1 SYMBOL: CELF_MP_CS_GET_RECONNECTION_TONE.....	108
424	45.1.1 <i>Syntax</i> .....	108
425	<i>Argument</i> .....	108
426	45.1.2 <i>108</i>	
427	45.1.3 <i>Return Value</i> .....	108
428	45.1.4 <i>Include File</i> .....	108
429	45.1.5 <i>Functional Description</i> .....	108
430	<b>46. GET NOISE CANCEL .....</b>	<b>109</b>
431	46.1 SYMBOL: CELF_MP_CS_GET_NOISE_CANCEL .....	109
432	46.1.1 <i>Syntax</i> .....	109
433	46.1.2 <i>Argument</i> .....	109
434	46.1.3 <i>Return Value</i> .....	109
435	46.1.4 <i>Include File</i> .....	109
436	46.1.5 <i>Functional Description</i> .....	109
437	<b>47. SET NOISE CANCEL .....</b>	<b>110</b>
438	47.1 SYMBOL: CELF_MP_CS_SET_NOISE_CANCEL.....	110
439	47.1.1 <i>Syntax</i> .....	110
440	47.1.2 <i>Argument</i> .....	110
441	47.1.3 <i>Return Value</i> .....	110
442	47.1.4 <i>Include File</i> .....	110
443	47.1.5 <i>Functional Description</i> .....	110
444	<b>48. GET CALL QUALITY ALARM.....</b>	<b>111</b>
445	48.1 SYMBOL: CELF_MP_CS_GET_CALL_QUALITY_ALARM.....	111
446	48.1.1 <i>Syntax</i> .....	111
447	48.1.2 <i>Argument</i> .....	111
448	48.1.3 <i>Return Value</i> .....	111
449	48.1.4 <i>Include File</i> .....	111
450	48.1.5 <i>Functional Description</i> .....	111
451	<b>49. SET CALL QUALITY ALARM.....</b>	<b>112</b>
452	49.1 SYMBOL: CELF_MP_CS_SET_CALL_QUALITY_ALARM.....	112
453	49.1.1 <i>Syntax</i> .....	112
454	49.1.2 <i>Argument</i> .....	112
455	49.1.3 <i>Return Value</i> .....	112
456	49.1.4 <i>Include File</i> .....	112
457	49.1.5 <i>Functional Description</i> .....	112
458	<b>50. GET NOISE CANCEL PERMIT.....</b>	<b>113</b>
459	50.1 SYMBOL: CELF_MP_CS_GET_NOISE_CANCEL_PERMIT .....	113
460	50.1.1 <i>Syntax</i> .....	113
461	50.1.2 <i>Argument</i> .....	113
462	50.1.3 <i>Return Value</i> .....	113
463	50.1.4 <i>Include File</i> .....	113
464	50.1.5 <i>Functional Description</i> .....	113
465	<b>51. GET HIGH PRIORITY COMMUNICATION MODE.....</b>	<b>114</b>
466	51.1 SYMBOL: CELF_MP_CS_GET_HI_PRIO_COM .....	114
467	51.1.1 <i>Syntax</i> .....	114
468	51.1.2 <i>Argument</i> .....	114
469	51.1.3 <i>Return Value</i> .....	114
470	51.1.4 <i>Include File</i> .....	114

471	51.1.5	<i>Functional Description</i> .....	114
472	<b>52.</b>	<b>SET HIGH PRIORITY COMMUNICATION MODE</b> .....	<b>115</b>
473	52.1	SYMBOL: CELF_MP_CS_SET_HI_PRIO_COM.....	115
474	52.1.1	<i>Syntax</i> .....	115
475	52.1.2	<i>Argument</i> .....	115
476	52.1.3	<i>Return Value</i> .....	115
477	52.1.4	<i>Include File</i> .....	115
478	52.1.5	<i>Functional Description</i> .....	115
479	<b>53.</b>	<b>GET PHONE ANSWERING SOUND ACTIVATION</b> .....	<b>116</b>
480	53.1	SYMBOL: CELF_MP_CS_GET_VM_SOUND_STATUS.....	116
481	53.1.1	<i>Syntax</i> .....	116
482	53.1.2	<i>Argument</i> .....	116
483	53.1.3	<i>Return Value</i> .....	116
484	53.1.4	<i>Include File</i> .....	116
485	53.1.5	<i>Functional Description</i> .....	116
486	<b>54.</b>	<b>SET PHONE ANSWERING SOUND ACTIVATION</b> .....	<b>117</b>
487	54.1	SYMBOL: CELF_MP_CS_SET_VM_SOUND_STATUS.....	117
488	54.1.1	<i>Syntax</i> .....	117
489	54.1.2	<i>Argument</i> .....	117
490	54.1.3	<i>Return Value</i> .....	117
491	54.1.4	<i>Include File</i> .....	117
492	54.1.5	<i>Functional Description</i> .....	117
493	<b>55.</b>	<b>GET AUTOMATIC RECEIVE STATUS</b> .....	<b>118</b>
494	55.1	SYMBOL: CELF_MP_CS_GET_AUTO_RCV_STATUS.....	118
495	55.1.1	<i>Syntax</i> .....	118
496	55.1.2	<i>Argument</i> .....	118
497	55.1.3	<i>Return Value</i> .....	118
498	55.1.4	<i>Include File</i> .....	118
499	55.1.5	<i>Functional Description</i> .....	118
500	<b>56.</b>	<b>SET AUTOMATIC RECEIVE STATUS</b> .....	<b>119</b>
501	56.1	SYMBOL: CELF_MP_CS_SET_AUTO_RCV_STATUS.....	119
502	56.1.1	<i>Syntax</i> .....	119
503	56.1.2	<i>Argument</i> .....	119
504	56.1.3	<i>Return Value</i> .....	119
505	56.1.4	<i>Include File</i> .....	119
506	56.1.5	<i>Functional Description</i> .....	119
507	<b>57.</b>	<b>GET AUTOMATIC TIMER</b> .....	<b>120</b>
508	57.1	SYMBOL: CELF_MP_CS_GET_AUTO_TIMER.....	120
509	57.1.1	<i>Syntax</i> .....	120
510	57.1.2	<i>Argument</i> .....	120
511	57.1.3	<i>Return Value</i> .....	120
512	57.1.4	<i>Include File</i> .....	120
513	57.1.5	<i>Functional Description</i> .....	120
514	<b>58.</b>	<b>SET AUTOMATIC TIMER</b> .....	<b>121</b>
515	58.1	SYMBOL: CELF_MP_CS_SET_AUTO_TIMER.....	121
516	58.1.1	<i>Syntax</i> .....	121
517	58.1.2	<i>Argument</i> .....	121
518	58.1.3	<i>Return Value</i> .....	121

519	58.1.4	<i>Include File</i> .....	121
520	58.1.5	<i>Functional Description</i> .....	121
521	<b>59.</b>	<b>GET RESET DATE</b> .....	<b>122</b>
522	59.1	SYMBOL: CELF_MP_CS_GET_RESET_DATE.....	122
523	59.1.1	<i>Syntax</i> .....	122
524	59.1.2	<i>Argument</i> .....	122
525	59.1.3	<i>Return Value</i> .....	122
526	59.1.4	<i>Include File</i> .....	122
527	59.1.5	<i>Functional Description</i> .....	122
528	<b>60.</b>	<b>SET RESET DATE</b> .....	<b>123</b>
529	60.1	SYMBOL: CELF_MP_CS_SET_RESET_DATE .....	123
530	60.1.1	<i>Syntax</i> .....	123
531	60.1.2	<i>Argument</i> .....	123
532	60.1.3	<i>Return Value</i> .....	123
533	60.1.4	<i>Include File</i> .....	123
534	60.1.5	<i>Functional Description</i> .....	123
535	<b>61.</b>	<b>GET CALL SILENT TIME</b> .....	<b>124</b>
536	61.1	SYMBOL: CELF_MP_CS_GET_CALL_SILENT_TIME.....	124
537	61.1.1	<i>Syntax</i> .....	124
538	61.1.2	<i>Argument</i> .....	124
539	61.1.3	<i>Return Value</i> .....	124
540	61.1.4	<i>Include File</i> .....	124
541	61.1.5	<i>Functional Description</i> .....	124
542	<b>62.</b>	<b>SET CALL SILENT TIME</b> .....	<b>125</b>
543	62.1	SYMBOL: CELF_MP_CS_SET_CALL_SILENT_TIME.....	125
544	62.1.1	<i>Syntax</i> .....	125
545	62.1.2	<i>Argument</i> .....	125
546	62.1.3	<i>Return Value</i> .....	125
547	62.1.4	<i>Include File</i> .....	125
548	62.1.5	<i>Functional Description</i> .....	125
549	<b>63.</b>	<b>GET CALL RECORDED</b> .....	<b>126</b>
550	63.1	SYMBOL: CELF_MP_CS_GET_CALL_RECORDED.....	126
551	63.1.1	<i>Syntax</i> .....	126
552	63.1.2	<i>Argument</i> .....	126
553	63.1.3	<i>Return Value</i> .....	126
554	63.1.4	<i>Include File</i> .....	126
555	63.1.5	<i>Functional Description</i> .....	126
556	<b>64.</b>	<b>SET CALL RECORDED</b> .....	<b>127</b>
557	64.1	SYMBOL: CELF_MP_CS_SET_CALL_RECORDED .....	127
558	64.1.1	<i>Syntax</i> .....	127
559	64.1.2	<i>Argument</i> .....	127
560	64.1.3	<i>Return Value</i> .....	127
561	64.1.4	<i>Include File</i> .....	127
562	64.1.5	<i>Functional Description</i> .....	127
563	<b>65.</b>	<b>SET RADIO</b> .....	<b>128</b>
564	65.1	SYMBOL: CELF_MP_CS_SET_RADIO .....	128
565	65.1.1	<i>Syntax</i> .....	128
566	65.1.2	<i>Argument</i> .....	128

567	65.1.3	<i>Return Value</i> .....	128
568	65.1.4	<i>Include File</i> .....	128
569	65.1.5	<i>Functional Description</i> .....	128
570	<b>66.</b>	<b>GET RADIO STATUS</b> .....	<b>129</b>
571	66.1	SYMBOL: CELF_MP_CS_GET_RADIO .....	129
572	66.1.1	<i>Syntax</i> .....	129
573	66.1.2	<i>Argument</i> .....	129
574	66.1.3	<i>Return Value</i> .....	129
575	66.1.4	<i>Include File</i> .....	129
576	66.1.5	<i>Functional Description</i> .....	129
577			

DRAFT

578 **0. Introduction**

579 Circuit-Switched Communication Service (CS Service) has the function of the call control, the call  
580 state management, the tone control and the log processing.

581 Circuit-Switched Communication Service includes

582 Voice communication service,

583 Video communication service, and

584 Digital data communication service.

585

586 **0.1 References**

587 **0.1.1 Normative**

588 RFC 2119: "Key words for use in RFCs to Indicate Requirement Levels", S. Bradner,  
589 March 1997, [URL:http://www.ietf.org/rfc/rfc2119.txt](http://www.ietf.org/rfc/rfc2119.txt)

590 RFC 2234: "Augmented BNF for Syntax Specifications: ABNF". D. Crocker, Ed., P. Overell.  
591 November 1997, [URL:http://www.ietf.org/rfc/rfc2234.txt](http://www.ietf.org/rfc/rfc2234.txt)

592

593 **0.1.2 Informative**

594

# 1. Primitives

595

This section contains the definitions of the data types and constants used in the interfaces of this service.

596

597

## 1.1 Constants

598

### 1.1.1 Line type (int)

599

CELF\_MP\_CS\_LINE\_CDMA1      CDMA1

600

CELF\_MP\_CS\_LINE\_UMTS      UMTS

601

CELF\_MP\_CS\_LINE\_FOMA      FOMA

602

CELF\_MP\_CS\_LINE\_GSM      GSM

603

CELF\_MP\_CS\_LINE\_WLAN      WLAN

604

CELF\_MP\_CS\_LINE\_BLUET      BLUET

605

The constants denote several different forms of wireless connectivity. Currently the first 3 are supported by this specification.

606

607

608

### 1.1.2 Dial Number

609

Dial number length of the other party

610

This data is valid when this mobile phone originates a call.

611

**Definition:**

612

CELF\_MP\_CS\_DIAL\_MAX      45

613

614

### 1.1.3 TAF address

615

TAF address is the connection ID in TAF (Terminal Adaptation Function).

616

This is external to the CELF MPP specification.

617

See the following documents for further details: 3GPP TS 27.001 / 3GPP TS 24.002

618

619

### 1.1.4 Additional Service

620

Service Info Name Length

621

**Definition:**

622

CELF\_SRVINFO\_TITLE      21

623

624

Dial data length for accessing the service

625

**Definition:**

626

CELF\_SRVINFO\_DATA      40

627

628

Response Message Name Length



629	<b>Definition:</b>		
630	CELF_RESMSG_TITLE	21	
631			
632	Response data length for accessing the service		
633	<b>Definition:</b>		
634	CELF_RESMSG_DATA	40	

DRAFT

## 635 1.2 Enums

### 636 1.2.1 Voice communication status (CelfMpCsComStatus)

637 In this operation mode, following multiple calls can be handled by a mobile phone,  
638 simultaneously. Each state of handling calls is:

- 639 • Holding a receiving call
- 640 • Talk over the phone
- 641 • Receive another incoming call

642 The number of handling states is three.

643 This is called the multiple calls.

644

645 In case that one call is AV call, the mobile phone handles this call only.

646

#### 647 1.2.1.1 Condition: only one call

648 CELF\_MP\_CS\_COM\_STATUS\_WAIT : Standby  
649 CELF\_MP\_CS\_COM\_STATUS\_IN : Incoming call  
650 CELF\_MP\_CS\_COM\_STATUS\_OUT : Outgoing call  
651 CELF\_MP\_CS\_COM\_STATUS\_CALL\_ALERT : Calling  
652 CELF\_MP\_CS\_COM\_STATUS\_TALK : In conversation  
653 CELF\_MP\_CS\_COM\_STATUS\_HOLD : In hold

654 This status is

655 (a) that incoming call was received, and

656 (b) that this incoming call cannot transit to conversation status because of the mobile  
657 phone.

658 CELF\_MP\_CS\_COM\_STATUS\_DISCONNECT : Disconnecting

659

#### 660 1.2.1.2 Condition: two call

661 One call is in conversation, and another call is in some status.

662 CELF\_MP\_CS\_COM\_STATUS\_2ND\_IN : In conversation and incoming  
663 CELF\_MP\_CS\_COM\_STATUS\_2ND\_OUT : In conversation and outgoing  
664 CELF\_MP\_CS\_COM\_STATUS\_2ND\_TALK : In conversation and calling  
665 CELF\_MP\_CS\_COM\_STATUS\_2ND\_HOLD : In conversation and hold  
666 CELF\_MP\_CS\_COM\_STATUS\_2ND\_DISCONNECT : In conversation and  
667 disconnecting

668 - three call One call is in conversation, another call is in hold, and 3rd call is in  
669 incoming.

670 CELF\_MP\_CS\_COM\_STATUS\_2ND\_HOLD\_IN : In conversation, hold, and incoming

671           **1.2.1.3 Condition: only one AV call**

- 672           CELF\_MP\_CS\_COM\_STATUS\_AV\_IN                 : Incoming AV call
- 673           CELF\_MP\_CS\_COM\_STATUS\_AV\_OUT                : Outgoing AV call
- 674           CELF\_MP\_CS\_COM\_STATUS\_AV\_CALL\_ALERT         : Calling AV call
- 675           CELF\_MP\_CS\_COM\_STATUS\_AV\_TALK                : In conversation
- 676           CELF\_MP\_CS\_COM\_STATUS\_AV\_HOLD                : In hold
- 677           CELF\_MP\_CS\_COM\_STATUS\_AV\_DISCONNECT         : Disconnecting an AV call

678           Other combinations of voice communication calls are not defined.

679

680           **1.2.2 Forwarding result (CelfMpCsFwResult)**

- 681           CELF\_MP\_CS\_OK                 : Successful forwarding
- 682           CELF\_MP\_CS\_ERR                : Forwarding failure

683

684           **1.2.3 Forwarding result details (CelfMpCsFwError)**

685           Set only at forwarding failure.

- 686           CELF\_MP\_CS\_FW\_ERROR\_NO\_JOIN                 : Service is not contracted.
- 687           CELF\_MP\_CS\_FW\_ERROR\_NO\_SETDATA               : the forwarded destination is not registered.
- 688
- 689           CELF\_MP\_CS\_FW\_ERROR\_ETC                 : Others

690

691           **1.2.4 Bearer Type (CelfMpCsBtype)**

- 692           CELF\_MP\_CS\_BTYPE\_NONE                 : None (unfixed)
- 693           CELF\_MP\_CS\_BTYPE\_ANY                 : Not Specified
- 694           CELF\_MP\_CS\_BTYPE\_VOICE                 : Voice
- 695           CELF\_MP\_CS\_BTYPE\_UD32                 : 32K communication
- 696           CELF\_MP\_CS\_BTYPE\_UD64                 : 64K communication
- 697           CELF\_MP\_CS\_BTYPE\_AV32                 : 32K communication
- 698           CELF\_MP\_CS\_BTYPE\_AV64                 : 64K communication

699

700           **1.2.5 Call Reference Status (CelfMpCsCallRefStatus)**

- 701           CELF\_MP\_CS\_USED                 : Valid "CN\_No" – Connection Number entry.
- 702           CELF\_MP\_CS\_UNUSED                : "CN\_No" – Connection Number is not used.

703           In some cases the Call reference status is unused, indicated by CELF\_MP\_CS\_UNUSED.  
704           If so, there is no connection between this mobile phone and other party and all data is void.

705

## 706 1.2.6 Call Status (CelfMpCsChan)

707 Call status for this mobile phone

708	CELF_MP_CS_CHAN_NULL	: Vacant
709	CELF_MP_CS_CHAN_OFFHK	: Off-hook
710	CELF_MP_CS_CHAN_OUT	: Outgoing call
711	CELF_MP_CS_CHAN_CALL_ALERT	: Calling
712	CELF_MP_CS_CHAN_IN	: Incoming call
713	CELF_MP_CS_CHAN_REQ_IN	: Response (conversation)
714	(The status of responding mobile phone is in conversation.)	
715	CELF_MP_CS_CHAN_TALK	: In conversation
716	CELF_MP_CS_CHAN_REQ_HOLD	: Response (hold)
717	(The status of responding mobile phone is in hold.)	
718	CELF_MP_CS_CHAN_HOLD	: Hold response
719	CELF_MP_CS_CHAN_2ND_HOLD	: In conversation and hold
720	CELF_MP_CS_CHAN_DISCONNECT	: Disconnecting

721

## 722 1.2.7 Existence of continuation data (CelfMpCsContData)

723 CELF\_MP\_CS\_ON : valid below data

724 CELF\_MP\_CS\_OFF : non valid below data

725 The following data entries in 1.3.8, from "Calling\_Dial" to "cause", is valid data if the call  
726 status is incoming or conversation and incoming call.

727

## 728 1.2.8 Busy Tone sound flag (CelfMpCsBusyTone)

729 Whether Busy Tone (engaged tone) sounds for this phone, or not

730 CELF\_MP\_CS\_BUSY\_TONE\_ON : Busy Tone sounding.

731 CELF\_MP\_CS\_BUSY\_TONE\_OFF : Busy Tone not sounding.

732

## 733 1.2.9 Cause of Calling Line Identity (CLI) not available 734 (CelfMpCsNoCLI)

735 The reason why the dial number of the other party is not notified.

736 The dial number of the other party is in "Calling dial" or "Called dial" status.

737 CELF\_MP\_CS\_NOCLI\_NOSRV : service is not supported.

738 CELF\_MP\_CS\_NOCLI\_BLOCKED : user blocked the display.

739 CELF\_MP\_CS\_NOCLI\_CONFLICT : service conflicts.

740 CELF\_MP\_CS\_NOCLI\_PUBLICPHONE : origination is from a public phone.

741 This data is valid, when the unsigned char num\_presentation\_indicator is filled.

742 1.2.10 Dial number / Redirect number display indicator  
743 (CelfMpCsPrsntInd)

744 Whether dial number / redirection number of the other party can be displayed, or not.

745 CELF\_MP\_CS\_PRSNT\_IND\_ALLOWED : Displayable

746 CELF\_MP\_CS\_PRSNT\_IND\_RESTRICTED : Impossible to display

747 CELF\_MP\_CS\_PRSNT\_IND\_NOT\_AVAILABLE : Displayable number does not exist.

748

749 1.2.11 Signal information (CelfMpCsSignal)

750 The type of tone of this phone

751 CELF\_MP\_CS\_SIGNAL\_DIAL\_TONE\_ON : Dial tone on

752 CELF\_MP\_CS\_SIGNAL\_RINGBACK\_TONE\_ON : Ring back tone on

753 CELF\_MP\_CS\_SIGNAL\_INTERCEPT\_TONE\_ON : Intercept tone on

754 CELF\_MP\_CS\_SIGNAL\_NW\_CONGESTION\_TONE\_ON : Network congestion tone on

755 CELF\_MP\_CS\_SIGNAL\_BUSY\_TONE\_ON : Busy tone on

756 CELF\_MP\_CS\_SIGNAL\_CONFIRM\_TONE\_ON : Confirm tone on

757 CELF\_MP\_CS\_SIGNAL\_ANSWER\_TONE\_ON : Answer tone on

758 CELF\_MP\_CS\_SIGNAL\_CALLWAITING\_TONE\_ON : Call waiting tone on

759 CELF\_MP\_CS\_SIGNAL\_OFFHK\_WARNING\_TONE\_ON : Off-hook warning tone on

760 CELF\_MP\_CS\_SIGNAL\_TONES\_OFF : Tones off

761 CELF\_MP\_CS\_SIGNAL\_ALERTING\_OFF : Alerting off

762 CELF\_MP\_CS\_SIGNAL\_NOTSET : Signal information is not set.

763

764 1.2.12 Originating Number notification (CelfMpCsNotice)

765 Whether the originating dial number is notified or not.

766 CELF\_MP\_CS\_NOTICE\_ON : Notified

767 CELF\_MP\_CS\_NOTICE\_OFF : Not notified

768 CELF\_MP\_CS\_NOTICE\_NOSET : Not set

769

770 1.2.13 Line status (CelfMpCsLineStatus)

771 CELF\_MP\_CS\_LINE\_STATUS\_OUT : Out-of-communication area

772 CELF\_MP\_CS\_LINE\_STATUS\_IN : Within-communication area

773

774 1.2.14 Normal and emergency originating restriction  
775 (CelfMpCsLineRestrictData)

776 CELF\_MP\_CS\_LINE\_RESTRICT\_DATA\_ON : With originating restriction

777 CELF\_MP\_CS\_LINE\_RESTRICT\_DATA\_OFF : Without originating restriction

778

### 779 1.2.15 Receive level (CelfMpCsRSSIlevel)

780 CELF\_MP\_CS\_RSSI\_LEVEL\_NONE : No reception

781 CELF\_MP\_CS\_RSSI\_LEVEL\_LOW : Receive level (Lowest)

782 CELF\_MP\_CS\_RSSI\_LEVEL\_MEDIUM1 : Receive level

783 CELF\_MP\_CS\_RSSI\_LEVEL\_MEDIUM2 : Receive level

784 CELF\_MP\_CS\_RSSI\_LEVEL\_HIGH : Receive level (Highest)

785

### 786 1.2.16 Area status information (CelfMpCsLineCvrStatus)

787 CELF\_MP\_CS\_LINE\_CVR\_STATUS\_IN : Inside the area

788 CELF\_MP\_CS\_LINE\_CVR\_STATUS\_OUT : Outside the area

789

### 790 1.2.17 RRC mode (CelfMpCsLineRRCMode)

791 CELF\_MP\_CS\_LINE\_RRC\_MODE\_IDLE : idle-mode

792 CELF\_MP\_CS\_LINE\_RRC\_MODE\_UTRAN : utran-connected-mode

793

### 794 1.2.18 Network identification information (CelfMpCsLineNetwork)

795 CELF\_MP\_CS\_LINE\_NETWORK\_HOME : home network

796 CELF\_MP\_CS\_LINE\_NETWORK\_VISIT : visiting network (roamed)

797 CELF\_MP\_CS\_LINE\_NO\_INFORMATION : No network sinformation

798

### 799 1.2.19 Service status (CelfMpLineSrvStatus)

800 CELF\_MP\_LINE\_SRV\_STATUS\_CS : CS is in service.

801 CELF\_MP\_LINE\_SRV\_STATUS\_PS : PS is in service.

802 CELF\_MP\_LINE\_SRV\_STATUS\_CSPS : CS and PS are in service.

803 CELF\_MP\_LINE\_NO\_INFORMATION : No information

804 CS is the Circuit-Switched Communication Service, and

805 PS is the Packet-Switched Communication Service.

806

### 807 1.2.20 Restriction status (CelfMpCsLineRestrict)

808 CELF\_MP\_CS\_LINE\_RESTRICT\_ON : In traffic restriction

809 CELF\_MP\_CS\_LINE\_RESTRICT\_OFF : Out of traffic restriction

810

811 1.2.21 Identifying flag (CelfMpCsFlag)

- 812 CELF\_MP\_CS\_NO\_FLAG : no Flag
- 813 CELF\_MP\_CS\_OPT\_FLAG : Network Service Option
- 814 CELF\_MP\_CS\_USSD\_FLAG : USSD

815

816 1.2.22 Notification Set (CelfMpCsNotifySet)

- 817 CELF\_MP\_CS\_CLASS\_COM\_STATUS : Voice communication status notification
- 818 CELF\_MP\_CS\_CLASS\_TLK\_TIME : Call duration notification
- 819 CELF\_MP\_CS\_CLASS\_DISC\_CAUSE : Disconnection cause notification
- 820 CELF\_MP\_CS\_CLASS\_FW\_RESULT : Call forwarding result notification
- 821 CELF\_MP\_CS\_CLASS\_OFFHK\_TO : Off-hook originating timeout notification

822

823 1.2.23 Event Structure Category

824 VoiceNotify

825

826 1.2.24 Event Structure Subtype

- 827 ConnInfo
- 828 TelCallTime
- 829 DiscCause
- 830 FW\_Result
- 831 OffHk\_Trn
- 832 DCF\_Event\_type
- 833 ArealInfo
- 834 RssiLevel

835

836 1.2.25 DCF Event Set (CelfMpCsDCFSet)

- 837 CELF\_MP\_CS\_DCF\_DISP : Display-related message
- 838 CELF\_MP\_CS\_DCF\_HISTORY : History-related message
- 839 CELF\_MP\_CS\_DCF\_TONE1 : Tone 1-related message
- 840 CELF\_MP\_CS\_DCF\_TONE2 : Tone 2-related message
- 841 CELF\_MP\_CS\_DCF\_ETC : Other messages
- 842 CELF\_MP\_CS\_CLASS\_ALL : All notified

843

844 1.2.26 Voice message (CelfMpCsRecMsg)

- 845 CELF\_MP\_CS\_REC\_MSG\_START : Start of a voice message

846 CELF\_MP\_CS\_REC\_MSG\_STOP : Stop of a voice message  
847

### 848 1.2.27 Off Hook Option (CelfMpCsOffHk)

849 CELF\_MP\_CS\_OFFHK\_AUTO : Automatic transmission  
850 CELF\_MP\_CS\_OFFHK\_MANUAL : Manual transmission  
851

### 852 1.2.28 64K/AV Communication (CelfMpCsUDComStatus)

853 CELF\_MP\_CS\_UD\_STOP : Under stop  
854 CELF\_MP\_CS\_UD\_TALK : Under communication  
855 CELF\_MP\_CS\_UD\_IN : Under incoming  
856 CELF\_MP\_CS\_UD\_OUT : Under outgoing  
857 CELF\_MP\_CS\_UD\_DISCONNECT : Under disconnection  
858 CELF\_MP\_CS\_UD\_CALL\_ALERT : Under calling  
859 CELF\_MP\_CS\_UD\_HOLD : Under hold  
860 CELF\_MP\_CS\_UD\_ERR : Error in UD communication  
861

### 862 1.2.29 AV Communication (CelfMpAVComStatus)

863 The INTERNAL reference denotes the originating event utilizing the terminal. The  
864 EXTERNAL reference denotes the usage of an outside of the terminal originating device.

865 CELF\_MP\_CS\_AV\_INTERNAL\_STOP : Under stop  
866 CELF\_MP\_CS\_AV\_INTERNAL\_TALK : Under communication  
867 CELF\_MP\_CS\_AV\_INTERNAL\_IN : Under incoming  
868 CELF\_MP\_CS\_AV\_INTERNAL\_OUT : Under outgoing  
869 CELF\_MP\_CS\_AV\_INTERNAL\_DISCONNECT : Under disconnection  
870 CELF\_MP\_CS\_AV\_INTERNAL\_CALL\_ALERT : Under calling  
871 CELF\_MP\_CS\_UD\_INTERNAL\_HOLD : Under hold  
872 CELF\_MP\_CS\_AV\_EXTERNAL\_STOP : Under stop  
873 CELF\_MP\_CS\_AV\_EXTERNAL\_TALK : Under communication  
874 CELF\_MP\_CS\_AV\_EXTERNAL\_IN : Under incoming  
875 CELF\_MP\_CS\_AV\_EXTERNAL\_OUT : Under outgoing  
876 CELF\_MP\_CS\_AV\_EXTERNAL\_DISCONNECT : Under disconnection  
877 CELF\_MP\_CS\_AV\_EXTERNAL\_CALLING\_ALERT : Under calling  
878 CELF\_MP\_CS\_UD\_EXTERNAL\_HOLD : Under hold  
879 CELF\_MP\_CS\_UD\_ERR : Error in UD communication  
880



881        **1.2.30 Receive Types (CelfMpCsRcvType)**

- 882        CELF\_MP\_CS\_RCV\_TYPE\_COMPETE\_TRN        : Communication Conflict
- 883        CELF\_MP\_CS\_RCV\_TYPE\_RSV\_RETURN        : Reestablish held call
- 884        CELF\_MP\_CS\_RCV\_TYPE\_CALL\_BACK        : Network Call Back
- 885        CELF\_MP\_CS\_RCV\_TYPE\_NORMAL        : Normal
- 886        CELF\_MP\_CS\_RCV\_TYPE\_NONE        : No Incoming Call

887

888        **1.2.31 Line Monitoring (CelfMpCsMtype)**

- 889        CELF\_MP\_CS\_MONITOR\_LINE\_STATUS        : Line status change notification
- 890        CELF\_MP\_CS\_MONITOR\_RESTRICT        : Restriction status change notification
- 891        CELF\_MP\_CS\_MONITOR\_RSSI        : Receive level change notification
- 892        CELF\_MP\_CS\_MONITOR\_ALL        : All notified

893

894        **1.2.32 Coverage Indicators (CelfMpCsCoverage)**

- 895        CELF\_MP\_CS\_LINE\_STATUS\_IN        : Within-communication area
- 896        CELF\_MP\_CS\_LINE\_STATUS\_OUT        : Out-of-communication area

897

898        **1.2.33 Incoming Call Selection (CelfMpCallSelect)**

- 899        CELF\_MP\_CS\_INCOMING\_VOICE\_ANSWERING : Forward to the phone-answering  
900        message
- 901        CELF\_MP\_CS\_INCOMING\_FORWARD        : Forward
- 902        CELF\_MP\_CS\_INCOMING\_REJECT        : Reject (disconnect)
- 903        CELF\_MP\_CS\_INCOMING\_NORMAL        : Receipt of an incoming call (normal  
904        incoming)

905

906        **1.2.34 Optional Registered Number (CelfMpRegNum)**

907        Within a network, optional numbers can be registered to specific services. It is reflected by  
908        this index into the list of these numbers.

- 909        int CelfMpRegNum        : Registration number

910

911        **1.2.35 Service Data (CelfMpCsSrvData)**

- 912        char\* CelfMpCsSrvData        : Pointer to additional service data

913

914        **1.2.36 Reconnection Tone (CelfMpCsReconnectionTone)**

- 915        CELF\_MP\_CS\_RECONN\_TONE\_OFF        : Tone OFF
- 916        CELF\_MP\_CS\_RECONN\_TONE\_LOW        : Tone ON low tone

917 CELF\_MP\_CS\_RECONN\_TONE\_HI : Tone ON high tone

918

### 919 1.2.37 Noise Canceling (CelfMpCsNoiseCancel)

920 CELF\_MP\_CS\_ON : Noise canceller ON

921 CELF\_MP\_CS\_OFF : Noise canceller OFF

922

### 923 1.2.38 Call Quality Alarm (CelfMpCsCallQualAlarm)

924

925 CELF\_MP\_CS\_CALL\_QUALITY\_ALM\_OFF : Quality alarm OFF

926 CELF\_MP\_CS\_CALL\_QUALITY\_ALM\_LOW : Quality alarm ON low tone

927 CELF\_MP\_CS\_CALL\_QUALITY\_ALM\_HI : Quality alarm ON high tone

928

### 929 1.2.39 Connection Priority Setting (CelfMpCsHiPrioCom)

930 CELF\_MP\_CS\_COMPRI\_NONE : No setting

931 CELF\_MP\_CS\_COMPRI\_VOICE : Voice

932 CELF\_MP\_CS\_COMPRI\_PACKET : Packet

933

### 934 1.2.40 Message Sound Settings (CelfMpCsVmSound)

935 CELF\_MP\_CS\_ON : Message sound ON

936 CELF\_MP\_CS\_OFF : Message sound OFF

937

### 938 1.2.41 Incoming Call Auto Receive (CelfMpCsAutoRcv)

939 CELF\_MP\_CS\_ON : Automatic incoming call ON

940 CELF\_MP\_CS\_OFF : Automatic incoming call OFF

## 941 1.3 Data Types and Structures

### 942 1.3.1 Circuit switched status notification event structure

943 In this sub-section, the associated data structure is CelfMpEvent with the following values:

944 category = VoiceNotify;

945 subtype = ConnInfo;

946 The value of field "info" is from enum CelfMpCsComStatus.

947 The field "data" carries:

948 CelfMpCsResChgInf res\_chg\_inf; // to be used in the case of:

949 // Restriction display information structure

950

### 951 1.3.2 Call duration notification event structure

952 In this sub-section, the associated data structure is CelfMpEvent with the following values:

953 category = VoiceNotify;

954 subtype = TelCallTime;

955 The value of field "info" is Call duration (seconds).

956 The field "data" is unused.

957

### 958 1.3.3 Disconnection cause notification event structure

959 In this sub-section, the associated data structure is CelfMpEvent with the following values:

960 category = VoiceNotify;

961 subtype = DiscCause;

962 The value of field "info" is the call reference.

963 The field "data" carries:

964 CelfMpCsDiscCause cme; // Disconnection cause information structure

965

### 966 1.3.4 Disconnection cause information structure

967 The error codes and cause information in this section directly correspond to 3GPP TS  
968 24.008.

969 typedef struct {

970 unsigned char e\_code; // Result code flag

971 unsigned char code; // Result code

972 unsigned char e\_cause; // Error reason flag

973 unsigned char cause; // Error reason

974 } CelfMpCsDiscCause;

975

976 **1.3.5 Forwarding result notification event structure**

977 In this sub-section, the associated data structure is CelfMpEvent with the following values:

978 category = VoiceNotify;

979 subtype = FW\_Result

980 The value of field "info" is the call reference.

981 The value of "subinfo" carries the forwarding result.

982 The field "data" carries:

983 CelfMpCsFwResult fw\_result; // Forwarding result structure

984

985 **1.3.6 Forwarding result structure (CelfMpCsFwResult)**

986 typedef struct {

987 int cause ; // forwarding result details

988 } CelfMpCsFwResult;

989

990 **1.3.7 Off-hook transmission timeout event structure**

991 In this sub-section, the associated data structure is CelfMpEvent with the following values:

992 category = VoiceNotify;

993 subtype = OffHk\_Trn

994 The value of field "info" is the call reference.

995 The field "data" is unused.

996

997 **1.3.8 Connection Destination Information (CelfMpConnectInfo)**

998 typedef struct {

999 CelfMpCallRef CN\_No; // Call reference

1000 CelfMpCsCallRefStatus CN\_status;

1001 int continue\_flag;

1002 unsigned char Calling\_Dial [CELF\_MP\_CS\_DIAL\_MAX+1];

1003 unsigned char Called\_Dial [CELF\_MP\_CS\_DIAL\_MAX+1];

1004 CelfMpCsBusyTone BusyToneSound\_inf;

1005 CelfMpCsBtype bc\_type;

1006 unsigned char[10] taf\_address;

1007 unsigned char cause\_of\_NoCLI;

1008 unsigned char num\_presentation\_indicator;

1009 unsigned char redirectnum [CELF\_MP\_CS\_DIAL\_MAX+1];

1010 unsigned char redirect\_presentation\_indicator;

Classification: *Circuit-Switched Service*

```

1011     unsigned char        signal;
1012     CelfMpCsDiscCause    cause; // Disconnection cause information structure
1013 } CelfMpCsConnectInf
1014

```

### 1.3.9 Connection Request (CelfMpCsConReq)

```

1015 typedef struct {
1016     CelfMpCsBtype        type;
1017     unsigned char *      dial_buf;
1018     int                   dial_len;
1019     CelfMpCsNotice       notice;
1020     unsigned char *      subaddr_buf;
1021     int                   subaddr_len;
1022 } CelfConReq
1023

```

### 1.3.10 Redirection number

Destination number of call transfer.

```

1026     redirectnum [CELF_MP_CS_DIAL_MAX+1]
1027

```

### 1.3.11 Channel Number Information (CelfMpCsChanNum)

CelfMpCsChanNum is used to hold call reference information.

If a channel is not used, CELF\_MP\_CS\_CHAN\_NOUSE is set as the call reference.

```

1032
1033 typedef struct {
1034     int Slot_0            // Call reference information 00
1035     int Slot_1            // Call reference information 01
1036     int Slot_2            // Call reference information 02
1037 } CelfMpCsChanNum
1038

```

### 1.3.12 Channel not in use Flag

```

1040     int CELF_MP_CS_CHAN_NOUSE : usually holds the call reference if channels are
1041     not used.

```

### 1.3.13 DCF Event Structure

In this sub-section, the associated data structure is CelfMpEvent with the following values:

```

1045     category = VoiceNotify;

```

```

1046     subtype = DCF_Event_type;

```

1047 The value of field "info" is the notification type.  
1048 The value of field "subinfo" is the bearer type  
1049 The field "data" carries:  
1050 DCF message structure corresponding to report types.  
1051

### 1.3.14 Line status change notification event structure

1052 In this sub-section, the associated data structure is CelfMpEvent with the following values:  
1053 category = VoiceNotify;  
1054 subtype = AreaInfo;  
1055 The value of field "info" is the line status.  
1056 The value of field "subinfo" is the line type.  
1057 The field "data" is unused.  
1058  
1059

### 1.3.15 Restriction display information structure (CelfMpCsResChgInf)

```
1060 typedef struct {  
1061     CelfMpCsLineRestrict    NcRestriction; // Normal originating restriction  
1062     CelfMpLineSrvStatus    ServiceStatus; // Service status  
1063     CelfMpCsLineRestrict    EcRestriction; // Emergency originating restriction  
1064 } CelfMpCsResChgInf;  
1065  
1066  
1067
```

### 1.3.16 Receive level change notification event structure

1068 In this sub-section, the associated data structure is CelfMpEvent with the following values:  
1069 category = VoiceNotify;  
1070 subtype = RssiLevel;  
1071 The value of field "info" is the receive level.  
1072 The value of field "subinfo" is the line type.  
1073 The field "data" is unused.  
1074  
1075

### 1.3.17 Line Status structure (CelfMpCsAreaRefChgInf)

```
1076 typedef struct {  
1077     CelfMpCsLineStatus      LineStatus ; // Line status  
1078     CelfMpCsCoverage        CoverageStatus ; // Service status  
1079     CelfMpCsLineRRCMode     RRC_mode ; // RRC mode  
1080     CelfMpCsLineNetwork     Network ; // Network identification  
1081     information  
1082
```

**Classification: *Circuit-Switched Service***

```

1083     CelfMpCsLineCvrStatus      ServiceStatus_AREA ; // Area status information
1084     // This information is set up when a cellular phone receives "AreaStatus-ind" from the
1085     // network.
1086     // It shows effective service in the present area.
1087     // Either "CS", "PS", "CS & PS" or "No data" is set.
1088     CelfMpCsLineRestrict      RestrictStatus ; // Restriction status
1089     CelfMpCsLineRestrict      NcRestriction ; // Normal originating restriction
1090     CelfMpLineSrvStatus      ServiceStatus_RES ; // Service status restriction
1091     // This information is set up when a cellular phone receives "Permission-ind" from the
1092     // network side.
1093     // It shows the service regulated in the present area.
1094     // Either "CS", "PS" or "CS & PS" is set.
1095     CelfMpCsLineRestrict      EcRestriction ; // Emergency originating
1096     restriction
1097 } CelfMpCsAreaRefChgInf ;
1098

```

**1.3.18 Additional service data structure (CelfMpCsAddSrvData)**

```

1100 typedef struct {
1101     CelfMpCsFlag  flag ;
1102     char  title[CELF_SRVINFO_TITLE]; // Additional service name
1103                                           // CELF_SRVINFO_TITLE=21
1104     char  send_no[CELF_SRVINFO_DATA]; // Dial data for accessing the service
1105                                           // CELF_SRVINFO_DATA=40
1106 } CelfMpCsAddSrvData;
1107

```

**1.3.19 Response Message Data Structure  
(CelfMpCsResponseMsgData)**

1110 The additional response message information is the service name and Dial data, which is  
1111 response message to send the network.

```

1112
1113 typedef struct {
1114     unsigned char  title[CELF_RESMSG_TITLE]; // Service name
1115     unsigned char  res_msg[CELF_RESMSG_DATA]; // Dial data
1116 } CelfMpCsResponseMsgData;
1117

```

```
1118     1.3.20 Line Status Extension (CelfMpCsLineStatusEx)
1119         unsigned char* CelfMpCsLineStatusEx;           // data for additional line status
1120         information
1121
1122     1.3.21 Number of stored messages (CelfMpCsVMNum)
1123         int CelfMpCsVMNum
1124
1125     1.3.22 Date Format Structure (CelfMpCsDate)
1126         typedef struct {
1127             unsigned char    Month
1128             unsigned char    Day
1129             unsigned char    Hour
1130             unsigned char    Minute
1131         } CelfMpCsDate
1132
1133     1.3.23 Dial Buffer (CelfMpCsDialBuffer)
1134         char* CelfMpCsDialBuffer
1135
1136     1.3.24 Dial Buffer Length (CelfMpCsDialLen)
1137         int CelfMpCsDialLen
1138
1139     1.3.25 Multi Party Operation (CelfMpCsMop)
1140         CELF_MP_CS_MOP_RSV_DISC:           // Disconnect the hold call
1141         CELF_MP_CS_MOP_DISC_AND_RSP:      // Response after disconnection
1142         CELF_MP_CS_MOP_RSV_AND_RSP:       // Response after hold (including operation for
1143         switching a call)
1144         CELF_MP_CS_MOP_CR_DISC:           // Disconnect call specified by the call reference
1145
1146     1.3.26 Timer Value (CelfMpCsTimer)
1147         int CelfMpCsTimer           // value 1 .. 120 seconds
1148
```



1149 **1.4 Events Type**

1150 **1.4.1 DCF Event Type**

- 1151 DCF\_Disp : Display-related message
- 1152 DCF\_History : History-related message
- 1153 DCF\_Tone1 : Tone 1-related message
- 1154 DCF\_Tone2 : Tone 2-related message
- 1155 DCF\_ETC : Other messages

1156

1157 **1.4.2 CCP Notification type**

1158 **1.4.2.1 CELF\_MP\_CS\_CCP\_CALLING\_START\_REQ**

1159 **Description:** Notification of starting display during CCP outgoing

1160

1161 **1.4.2.2 CELF\_MP\_CS\_CCP\_CALLED\_START\_IND**

1162 **Description:** Notification of starting display during CCP incoming

1163

1164 **1.4.2.3 CELF\_MP\_CS\_CCP\_CALLING\_ALERT\_IND**

1165 **Description:** Notification of starting display during CCP calling

1166

1167 **1.4.2.4 CELF\_MP\_CS\_CCP\_CONNECT\_START\_RSP**

1168 **Description:** Notification of starting display during CCP connection

1169

1170 **1.4.2.5 CELF\_MP\_CS\_CCP\_CONNECT\_START\_IND**

1171 **Description:** Notification of starting display during CCP communication

1172

1173 **1.4.2.6 CELF\_MP\_CS\_CCP\_RELEASE\_IND**

1174 **Description:** Notification of ending CCP display

1175

1176 **1.4.2.7 CELF\_MP\_CS\_CCP\_DISCONNECT\_REQ**

1177 **Description:** Notification of starting CCP disconnection (on a mobile device) display

1178

1179 **1.4.2.8 CELF\_MP\_CS\_CCP\_DISCONNECT\_START\_IND**

1180 **Description:** Notification of starting CCP disconnection (on a network) display

1181

1182           **1.4.2.9 CELF\_MP\_CS\_CCP\_CALLING\_REJ\_IND**

1183           **Description:** Notification of rejecting CCP outgoing

1184

1185           **1.4.2.10 CELF\_MP\_CS\_CCP\_HOLD\_CNF**

1186           **Description:** Notification of CCP hold

1187

1188           **1.4.2.11 CELF\_MP\_CS\_CCP\_RETREIVE\_CNF**

1189           **Description:** Notification of releasing CCP hold

1190

1191           **1.4.2.12 CELF\_MP\_CS\_CCP\_CALLING\_SETUP\_REQ**

1192           **Description:** Notification of registering CCP outgoing call history

1193

1194           **1.4.2.13 CELF\_MP\_CS\_CCP\_CALLED\_REJ\_REQ**

1195           **Description:** Notification of registering CCP absence incoming call history

1196

1197           **1.4.2.14 CELF\_MP\_CS\_CCP\_CALLED\_SETUP\_RSP**

1198           **Description:** Notification of registering CCP incoming call history

1199

1200           **1.4.2.15 CELF\_MP\_CS\_CCP\_RGT\_START**

1201           **Description:** Notification of CCP RGT start

1202

1203           **1.4.2.16 CELF\_MP\_CS\_CCP\_RGT\_STOP**

1204           **Description:** Notification of CCP RGT stop

1205

1206           **1.4.2.17 CELF\_MP\_CS\_CCP\_HRGT\_START**

1207           **Description:** Start notification of incoming of a CCP hold call

1208

1209           **1.4.2.18 CELF\_MP\_CS\_CCP\_HRGT\_STOP**

1210           **Description:** Stop notification of incoming of a CCP hold call

1211

1212           **1.4.2.19 CELF\_MP\_CS\_CCP\_DST\_START**

1213           **Description:** Notification of CCP DST start

1214

1215           **1.4.2.20 CELF\_MP\_CS\_CCP\_DST\_STOP**

1216           **Description:** Notification of CCP DST stop

1217

#### 1.4.2.21 CELF\_MP\_CS\_CCP\_RBT\_START

1218

**Description:** Notification of CCP RBT start

1219

1220

#### 1.4.2.22 CELF\_MP\_CS\_CCP\_RBT\_STOP

1221

**Description:** Notification of CCP RBT stop

1222

1223

#### 1.4.2.23 CELF\_MP\_CS\_CCP\_BT\_START

1224

**Description:** Notification of CCP BT start

1225

1226

#### 1.4.2.24 CELF\_MP\_CS\_CCP\_CWT\_START

1227

**Description:** Notification of CCP CWT start

1228

1229

#### 1.4.2.25 CELF\_MP\_CS\_CCP\_CWT\_STOP

1230

**Description:** Notification of CCP CWT stop

1231

1232

#### 1.4.2.26 CELF\_MP\_CS\_CCP\_REJECT\_ASK

1233

**Description:** Inquiry report of rejecting a CCP CS incoming call

1234

1235

### 1.4.3 Notification type

1236

CELF\_MP\_CS\_RSMP\_REST\_START : Restriction display start notification

1237

CELF\_MP\_CS\_RSMP\_REST\_STOP : Restriction display stop notification

1238

1239

### 1.4.4 Restriction status

1240

The 0th bit is used for PS restriction status, and the 1st bit is used for CS restriction status.

1241

(Bit ON means "restricted." Bit OFF means "unrestricted.")

1242

CELF\_MP\_BIT\_RESTINF\_CS : CS restriction information

1243

CELF\_MP\_BIT\_RESTINF\_PS : PS restriction information

1244

The 2nd bit is used for PS emergency restriction status, and the 3rd bit is used for CS emergency restriction status.

1245

1246

CELF\_MP\_BIT\_ECRESTINF\_CS : Emergency CS restriction information

1247

CELF\_MP\_BIT\_ECRESTINF\_PS : Emergency PS restriction information

1248

## 1249 **1.5 Status Codes (CelfMpStatus)**

1250	CELF_MP_STATUS_OK	: successful completion
1251	CELF_MP_STATUS_APP_ID_ERR	: Application ID is not valid
1252	CELF_MP_STATUS_CALL_NO_ERR	: Call number is not valid
1253	CELF_MP_STATUS_EVENT_SET_ERR	: Notification event set is not valid
1254	CELF_MP_STATUS_COM_TYPE_ERR	: Communication type is not valid
1255	CELF_MP_STATUS_MON_TYPE_ERR	: Monitor type is not valid
1256	CELF_MP_STATUS_ERR	: Other unsuccessful completion
1257	CELF_MP_CS_ONHOOK_DENY	: On-hook originating is impossible.
1258	CELF_MP_CS_ONHOOK_STATUS_ERR	: Error due to communication conflict
1259	CELF_MP_CS_ONHOOK_OB_CR	: Excess of the maximum number of calls
1260		
1261		

DRAFT

1262

## 2. Start Notification

1263

### 2.1 Symbol: `celf_mp_cs_notification_start`

1264

#### 2.1.1 Syntax

1265

```
CelfMpStatus celf_mp_cs_notification_start (
```

1266

```
    CelfMpAppID          app_id,
```

1267

```
    CelfMpCsNotifySet   event_set,
```

1268

```
    CelfMpCallback      callback_func);
```

1269

1270

#### 2.1.2 Argument

1271

**Name:** `app_id`

1272

**Type:** `CelfMpAppId`

1273

**I/O:** |

1274

**Description:**

1275

Application identifier.

1276

1277

**Name:** `event_set`

1278

**Type:** `CelfMpCsNotifySet`

1279

**I/O:** |

1280

**Description:**

1281

Notification event set. Events that are classified as belonging to one of the

1282

`CelfMpCsNotifySet` class **may** be registered to have a callback function called when

1283

the event occurs for the application identified by `app_id`. Classes of events are enabled by

1284

setting the corresponding bit in `event_set`.

1285

1286

The event classes are defined as follows:

1287

`CELF_MP_CS_CLASS_COM_STATUS`: Voice communication status notification

1288

`CELF_MP_CS_CLASS_TLK_TIME`: Call duration notification

1289

`CELF_MP_CS_CLASS_DISC_CAUSE`: Disconnection cause notification

1290

`CELF_MP_CS_CLASS_FW_RESULT`: Call forwarding result notification

1291

`CELF_MP_CS_CLASS_OFFHK_TO`: Off-hook originating timeout notification

1292

1293

A callback **may** be registered for all classes of events using special event class

1294

`CELF_MP_CS_CLASS_ALL`, however to reduce overhead it is recommended that only the

1295

needed event classes **should** be registered.

1296

1297

**Name:** `callback_func`

1298

**Type:** `CelfMpCallback`

1299           **I/O:**     |  
1300           **Description:**  
1301           The callback function, which **shall** be called when an event occurs from one of the classes  
1302           in `event_set`.

1303

### 1304   2.1.3 Return Value

1305           **Type:**    CelfMpStatus

1306           **Description:**

1307           `celf_mp_cs_notification_start()` **shall** return one of the following values:

1308           CELF\_MP\_STATUS\_OK:               successful completion

1309           CELF\_MP\_STATUS\_APP\_ID\_ERR:       Application ID is not valid

1310           CELF\_MP\_STATUS\_EVENT\_SET\_ERR:    Notification event set is not valid

1311           CELF\_MP\_STATUS\_ERR:            Other unsuccessful completion.

1312

### 1313   2.1.4 Include File

1314           `/usr/include/celf/mp_cs.h`

1315

### 1316   2.1.5 Functional Description

1317           This is a synchronous function.

1318           This function is used to start notification callbacks for events related to circuit-switched  
1319           communication.

1320           Events from a registered class **shall** cause the registered callback function to be called  
1321           when the event occurs for the application identified by `app_id`. If a class of events does not  
1322           have a registered callback function, no callback **shall** occur for those events.

1323

1324           The event structure `CelfMpEvent` **must** be used and the value subtype **shall be set to**  
1325           **ConnInfo**. See section 1.3.1.

1326

1327

## 1328 3. Stop Notification

### 1329 3.1 Symbol: `celf_mp_cs_notification_stop`

#### 1330 3.1.1 Syntax

```
1331 CelfMpStatus celf_mp_cs_notification_stop (
1332     CelfMpAppId      app_id,
1333     CelfMpCsNotifySet event_set);
1334
```

#### 1335 3.1.2 Argument

1336 **Name:** `app_id`

1337 **Type:** `CelfMpAppId`

1338 **I/O:** `I`

1339 **Description:**

1340 Application identifier.

1341

1342 **Name:** `event_set`

1343 **Type:** `CelfMpCsNotifySet`

1344 **I/O:** `I`

1345 **Description:**

1346 Notification event set. Events that are classified as belonging to one of the  
 1347 `CelfMpCsNotifySet` class **may** be registered to have a callback function called when  
 1348 the event occurs for the application identified by `app_id`. Classes of events are enabled by  
 1349 setting the corresponding bit in `event_set`.

1350

1351 The event classes are defined as follows:

1352 `CELF_MP_CS_CLASS_COM_STATUS`: Voice communication status notification

1353 `CELF_MP_CS_CLASS_TLK_TIME`: Call duration notification

1354 `CELF_MP_CS_CLASS_DISC_CAUSE`: Disconnection cause notification

1355 `CELF_MP_CS_CLASS_FW_RESULT`: Call forwarding result notification

1356 `CELF_MP_CS_CLASS_OFFHK_TO`: Off-hook originating timeout notification

1357

#### 1358 3.1.3 Return Value

1359 **Type:** `CelfMpStatus`

1360 **Description:**

1361 `celf_mp_cs_notification_stop()` **shall** return one of the following values:

1362 `CELF_MP_STATUS_OK`: successful completion

1363 `CELF_MP_STATUS_APP_ID_ERR`: Application ID is not valid.

1364 CELF\_MP\_STATUS\_EVENT\_SET\_ERR: Notification event set is not valid

1365 CELF\_MP\_STATUS\_ERR: Other unsuccessful completion.

1366

### 1367 3.1.4 Include File

1368 /usr/include/celf/mp\_cs.h

1369

### 1370 3.1.5 Functional Description

1371 This function stops all voice communication related event reporting.

1372 If any voice communication function is called after the notification has been stopped, the  
1373 call **shall** report CELF\_MP\_STATUS\_ERR.

1374 For notification events, see "Start notification".

1375 Note: For further information about the event structure consult section 1.3 in this document.

1376

1377

1378

DRAFT



1379

## 4. Get Voice Communication Status

1380

### 4.1 Symbol: `celf_mp_cs_get_com_status`

1381

#### 4.1.1 Syntax

1382

```
CelfMpStatus celf_mp_cs_get_com_status (
```

1383

```
    CelfMpAppld app_id);
```

1384

1385

#### 4.1.2 Argument

1386

**Name:** `app_id`

1387

**Type:** `CelfMpAppld`

1388

**I/O:** `I`

1389

**Description:**

1390

Application identifier.

1391

1392

#### 4.1.3 Return Value

1393

**Type:** `CelfMpStatus`

1394

**Description:**

1395

`celf_mp_cs_get_com_status()` shall return one of the values defined in section 0.1.

1396

1397

#### 4.1.4 Include File

1398

`/usr/include/celf/mp_cs.h`

1399

1400

#### 4.1.5 Functional Description

1401

This function gets the current voice communication status.

1402

Without the monitoring the voice communication, it is possible to get the status of voice communication.

1403

1404

1405

1406

## 1407 5. Get Connection Information to Other Party

### 1408 5.1 Symbol: `celf_mp_cs_get_con_info_ref`

#### 1409 5.1.1 Syntax

```
1410 CelfMpStatus celf_mp_cs_get_con_info_ref (  
1411     CelfMpAppld      app_id,  
1412     CelfMpCallRef    call_no,  
1413     CelfMpConnectInfo connect_inf_p);  
1414
```

#### 1415 5.1.2 Argument

1416 **Name:** `app_id`

1417 **Type:** `CelfMpAppld`

1418 **I/O:** |

1419 **Description:**

1420 Application identifier.

1421

1422 **Name:** `call_no`

1423 **Type:** `CelfMpCallRef`

1424 **I/O:** |

1425 **Description:**

1426 Call reference (0 to 255).

1427

1428 **Name:** `connect_inf_p`

1429 **Type:** `CelfMpConnectInfo`

1430 **I/O:** |

1431 **Description:**

1432 Pointer to the connection destination information. See section 0.1 for details.

1433

#### 1434 5.1.3 Return Value

1435 **Type:** `CelfMpStatus`

1436 **Description:**

1437 `celf_mp_cs_get_con_info_ref()` **shall** return one of the values defined:

1438 `CELF_MP_STATUS_OK:` successful completion

1439 `CELF_MP_STATUS_APP_ID_ERR:` Application ID is not valid.

1440 `CELF_MP_STATUS_CALL_NO_ERR:` Call number is not valid

1441 `CELF_MP_STATUS_ERR:` Other unsuccessful completion.

1442

#### 1443 5.1.4 Include File

1444 /usr/include/celf/mp\_cs.h

1445

#### 1446 5.1.5 Functional Description

1447 This function refers to the connection information to other party specified call reference

1448 Without the monitoring the voice communication, it is possible to get the connection  
1449 information

1450

1451 In the following cases, the CelfMpStatus is set CELF\_MP\_CS\_ERR.

- 1452 1. The call specified by call reference does not exist.  
1453 2. Other parameter Error.

1454

DRAFT

1455

## 6. Get Call Duration

1456

### 6.1 Symbol: `celf_mp_cs_get_call_duration`

1457

#### 6.1.1 Syntax

1458

```
CelfMpStatus celf_mp_cs_get_call_duration (
```

1459

```
    CelfMpAppId app_id,
```

1460

```
    CelfMpTime time);
```

1461

1462

#### 6.1.2 Argument

1463

**Name:** app\_id

1464

**Type:** CelfMpAppId

1465

**I/O:** I

1466

**Description:**

1467

Application identifier.

1468

1469

**Name:** time

1470

**Type:** CelfMpTime

1471

**I/O:** O

1472

**Description:**

1473

time returns the current call duration in seconds.

1474

1475

#### 6.1.3 Return Value

1476

**Type:** CelfMpStatus

1477

**Description:**

1478

`celf_mp_cs_get_con_info_ref()` **shall** return one of the values defined:

1479

CELF\_MP\_STATUS\_OK: successful completion

1480

CELF\_MP\_STATUS\_APP\_ID\_ERR: Application ID is not valid.

1481

CELF\_MP\_STATUS\_ERR: Other unsuccessful completion.

1482

1483

#### 6.1.4 Include File

1484

`/usr/include/celf/mp_cs.h`

1485

1486

#### 6.1.5 Functional Description

1487

This function gets the call duration on the current call.

1488

The call duration is counted by the voice communication service.

1489           When no call exists, the function returns zero.  
1490

DRAFT

1491

## 7. Off-Hook Notification

1492

### 7.1 Symbol: `celf_mp_cs_notification_off_hook`

1493

#### 7.1.1 Syntax

1494

```
CelfMpStatus celf_mp_cs_notification_off_hook (
```

1495

```
    CelfMpAppld      app_id,
```

1496

```
    CelfMpCsBtype    com_type,
```

1497

```
    CelfMpCsOffHk    option);
```

1498

1499

#### 7.1.2 Argument

1500

**Name:** `app_id`

1501

**Type:** `CelfMpAppld`

1502

**I/O:** |

1503

**Description:**

1504

Application identifier.

1505

1506

**Name:** `com_type`

1507

**Type:** `CelfMpCsBtype`

1508

**I/O:** |

1509

**Description:**

1510

Communication type as defined in section 1.2.4

1511

1512

**Name:** `option`

1513

**Type:** `CelfMpCsOffHk`

1514

**I/O:** |

1515

**Description:**

1516

One the following options **shall** be set:

1517

`CELF_MP_CS_OFFHK_AUTO` Automatic transmission

1518

`CELF_MP_CS_OFFHK_MANUAL` Manual transmission

1519

1520

#### 7.1.3 Return Value

1521

**Type:** `CelfMpStatus`

1522

**Description:**

1523

`celf_mp_cs_notification_off_hook()` **shall** return one of the values defined:

1524

`CELF_MP_STATUS_OK:` successful completion

1525

`CELF_MP_STATUS_APP_ID_ERR:` Application ID is not valid.

1526 CELF\_MP\_STATUS\_COM\_TYPE\_ERR: Communication type is not valid

1527 CELF\_MP\_STATUS\_ERR: Other unsuccessful completion.

1528

#### 1529 7.1.4 Include File

1530 /usr/include/celf/mp\_cs.h

1531

#### 1532 7.1.5 Functional Description

1533 This function receives the request of off-hook.

1534

1535 The term "off-hook" refers to the user first presses the "dial" button, then enters the number  
1536 to dial.

1537

1538 By this function,

1539 (1) When the mobile phone is in the wait (standby) status, the dial tone (DT) sounds and it  
1540 is possible to input dial number, or

1541 (2) When the input of dial number is completed, the mobile phone starts the originating.

1542 Because the function is an immediate return function, to confirm the complete result,  
1543 including the negotiation with the network, `celf_mp_cs_notification_status()`  
1544 shall be used to obtain the communication status.

1545

1546 When a mobile phone is moved to low voltage mode, a low voltage notification is sent.

1547 During low voltage, when the communication status is other than the under standby, this  
1548 Off-hook is disabled.

1549

1550 If an incoming call arrives during off-hook, this Off-hook is cancelled.

1551 In case of using the subaddress, it should be use the function "On-hook originating".

## 1552 8. Disconnect

### 1553 8.1 Symbol: `celf_mp_cs_disconnect`

#### 1554 8.1.1 Syntax

```
1555 CelfMpStatus celf_mp_cs_disconnect (  
1556     CelfMpAppId      app_id,  
1557     CelfMpCsBtype    com_type);  
1558
```

#### 1559 8.1.2 Argument

1560 **Name:** app\_id  
1561 **Type:** CelfMpAppId  
1562 **I/O:** |  
1563 **Description:**  
1564 Application identifier.

1565  
1566 **Name:** com\_type  
1567 **Type:** CelfMpCsBtype  
1568 **I/O:** |  
1569 **Description:**  
1570 Communication type as defined in section 1.2.4.  
1571

#### 1572 8.1.3 Return Value

1573 **Type:** CelfMpStatus  
1574 **Description:**  
1575 `celf_mp_cs_disconnect()` shall return one of the values defined:  
1576 CELF\_MP\_STATUS\_OK: successful completion  
1577 CELF\_MP\_STATUS\_APP\_ID\_ERR: Application ID is not valid  
1578 CELF\_MP\_STATUS\_COM\_TYPE\_ERR: Communication type is not valid  
1579 CELF\_MP\_STATUS\_ERR: Other unsuccessful completion.

1580

#### 1581 8.1.4 Include File

1582 `/usr/include/celf/mp_cs.h`

1583

#### 1584 8.1.5 Functional Description

1585 This function receives the request to disconnect the call.



1586

1587 Because the function is an immediate return function, to confirm the complete result,  
1588 including the negotiation with the network, it should be issued  
1589 `celf_mp_cs_notification_status()` to obtain the communication status.

1590

1591 An incoming call cannot be disconnected by this function. (Use "Reject incoming call")

1592

1593 If multiple calls exist, all calls are disconnected.

1594

DRAFT

## 1595 9. Dial

### 1596 9.1 Symbol: `celf_mp_cs_dial`

#### 1597 9.1.1 Syntax

```
1598 CelfMpStatus celf_mp_cs_dial (  
1599     CelfMpAppId      app_id,  
1600     CelfMpCsBtype    com_type,  
1601     CelfMpCsDialBuffer dial_buf,  
1602     CelfMpCsDialLen  dial_len);  
1603
```

#### 1604 9.1.2 Argument

1605 **Name:** `app_id`

1606 **Type:** `CelfMpAppId`

1607 **I/O:** |

1608 **Description:**

1609 Application identifier.

1611 **Name:** `com_type`

1612 **Type:** `CelfMpCsBtype`

1613 **I/O:** |

1614 **Description:**

1615 Communication type as defined in section 1.2.4.

1617 **Name:** `dial_buf`

1618 **Type:** `CelfMpCsDialBuffer`

1619 **I/O:** |

1620 **Description:**

1621 Dial data buffer address

1623 **Name:** `dial_len`

1624 **Type:** `CelfMpCsDialLen`

1625 **I/O:** |

1626 **Description:**

1627 Dial data length

1628

1629 **9.1.3 Return Value**

1630 **Type:** CelfMpStatus

1631 **Description:**

1632 `celf_mp_cs_dial()` shall return one of the values defined:

1633 CELF\_MP\_STATUS\_OK: successful completion

1634 CELF\_MP\_STATUS\_APP\_ID\_ERR: Application ID is not valid.

1635 CELF\_MP\_STATUS\_COM\_TYPE\_ERR: Communication type is not valid

1636 CELF\_MP\_STATUS\_ERR: Other unsuccessful completion.

1637

1638 **9.1.4 Include File**

1639 `/usr/include/celf/mp_cs.h`

1640

1641 **9.1.5 Functional Description**

1642 This function receives the sequence of dial number.

1643

1644 Because the function is an immediate return function, to confirm the complete result,  
1645 including the negotiation with the network, it should be issued  
1646 "celf\_mp\_cs\_notification\_status()" to obtain the communication status.

1647

1648 The dial data stores the following ASCII codes.

1649 1 : 0 x 31    2 : 0 x 32    3 : 0 x 33

1650 4 : 0 x 34    5 : 0 x 35    6 : 0 x 36

1651 7 : 0 x 37    8 : 0 x 38    9 : 0 x 39

1652 \* : 0 x 2a    0 : 0 x 30    # : 0 x 23

1653

1654 Under this off-hook status, the mobile phone starts an outgoing call with "Dial" and  
1655 "Complete dial".

1656 Five seconds later from the last digit has been entered, the outgoing process starts  
1657 automatically, when automatic transmission is specified in "Off-hook".

1658 When "Off-hook" is called, the mobile phone is in off-hook status.

1659

1660 Under this on-hook status, DTMF is sent, if the status is (a) the conversation or (b) the  
1661 conversation and hold.

1662

## 10.Dial Complete

1663

### 10.1 Symbol: `celf_mp_cs_dial_end`

1664

#### 10.1.1 Syntax

1665

```
CelfMpStatus celf_mp_cs_dial_end (
```

1666

```
    CelfMpAppId      app_id,
```

1667

```
    CelfMpCsBtype    com_type);
```

1668

1669

#### 10.1.2 Argument

1670

**Name:** `app_id`

1671

**Type:** `CelfMpAppId`

1672

**I/O:** `I`

1673

**Description:**

1674

Application identifier.

1675

1676

**Name:** `com_type`

1677

**Type:** `CelfMpCsBtype`

1678

**I/O:** `I`

1679

**Description:**

1680

Communication type as defined in section 1.2.4.

1681

1682

#### 10.1.3 Return Value

1683

**Type:** `CelfMpStatus`

1684

**Description:**

1685

`celf_mp_cs_dial_end()` **shall** return one of the values defined:

1686

`CELF_MP_STATUS_OK:` successful completion

1687

`CELF_MP_STATUS_APP_ID_ERR:` Application ID is not valid.

1688

`CELF_MP_STATUS_COM_TYPE_ERR:` Communication type is not valid

1689

`CELF_MP_STATUS_ERR:` Other unsuccessful completion.

1690

1691

#### 10.1.4 Include File

1692

`/usr/include/celf/mp_cs.h`

1693

1694

#### 10.1.5 Functional Description

1695

This function receives the request to end the dial entry.

1696

1697

Because this is an asynchronous function the service will return the result through a notification.

1698

1699

`celf_mp_cs_notification_status()` shall be used to obtain the communication status.

1700

1701

Under off-hook status, the mobile phone starts outgoing operation by calling this function with dial number, which was given by preceding function calls "Dial".

1702

1703

1704

Under on-hook status, the calling this function is disabled, a `CELF_MP_STATUS_ERR` shall be returned.

1705

1706

DRAFT

## 1707 11.Response to Incoming Call

### 1708 11.1 Symbol: `celf_mp_cs_call_rcv`

#### 1709 11.1.1 Syntax

```
1710 CelfMpStatus celf_mp_cs_call_rcv (  
1711     CelfMpAppId      app_id,  
1712     CelfMpCsBtype    com_type);  
1713
```

#### 1714 11.1.2 Argument

1715 **Name:** `app_id`

1716 **Type:** `CelfMpAppId`

1717 **I/O:** `I`

1718 **Description:**

1719 Application identifier.

1720

1721 **Name:** `com_type`

1722 **Type:** `CelfMpCsBtype`

1723 **I/O:** `I`

1724 **Description:**

1725 Communication type as defined in section 1.2.4.

1726

#### 1727 11.1.3 Return Value

1728 **Type:** `CelfMpStatus`

1729 **Description:**

1730 `celf_mp_cs_call_rcv()` **shall** return one of the values defined:

1731 `CELf_MP_STATUS_OK:` successful completion

1732 `CELf_MP_STATUS_APP_ID_ERR:` Application ID is not valid.

1733 `CELf_MP_STATUS_COM_TYPE_ERR:` Communication type is not valid

1734 `CELf_MP_STATUS_ERR:` Other unsuccessful completion.

1735

#### 1736 11.1.4 Include File

1737 `/usr/include/celf/mp_cs.h`

1738

#### 1739 11.1.5 Functional Description

1740 This function receives the request to process an incoming call.

1741

1742 Because the function is an immediate return function, to confirm the complete result,  
1743 including the negotiation with the network, it should be issued  
1744 `celf_mp_cs_notification_status()` to obtain the communication status.

1745

1746 One of the following operations is performed depending on the mobile phone status.

1747 Incoming : Responds to the incoming call.

1748 In hold : Responds to the response hold call

1749 Others : Disabled

1750

1751 If the mobile phone is in low voltage mode, this function is disabled.

1752

1753 To respond to the incoming call in the status, "in conversation and incomings", use "Reject  
1754 incoming call".

DRAFT

## 1755 12.Forward Incoming Call

### 1756 12.1 Symbol: `celf_mp_cs_call_forward`

#### 1757 12.1.1 Syntax

```
1758 CelfMpStatus celf_mp_cs_call_forward (  
1759 CelfMpCsBtype com_type);
```

1760

#### 1761 12.1.2 Argument

1762 **Name:** `com_type`

1763 **Type:** `CelfMpCsBtype`

1764 **I/O:** |

1765 **Description:**

1766 Communication type as defined in section 1.2.4.

1767

#### 1768 12.1.3 Return Value

1769 **Type:** `CelfMpStatus`

1770 **Description:**

1771 `celf_mp_cs_call_forward()` shall return one of the values defined:

1772 `CELF_MP_STATUS_OK:` successful completion

1773 `CELF_MP_STATUS_COM_TYPE_ERR:` Communication type is not valid

1774 `CELF_MP_STATUS_ERR:` Other unsuccessful completion.

1775

#### 1776 12.1.4 Include File

1777 `/usr/include/celf/mp_cs.h`

1778

#### 1779 12.1.5 Functional Description

1780 This function receives the request to forward an incoming call.

1781

1782 Because the function is an immediate return function, to confirm the complete result,  
1783 including the negotiation with the network, a `celf_mp_cs_notification_status()`  
1784 should be issued to obtain the communication status.

1785

1786 The incoming call is forwarded when the communication status is (a) under the incoming,  
1787 (b) under conversation and incoming, or (c) under hold and incoming.

1788

1789 If the forwarding fails, incoming call is continued between other party and this phone.



1790

## 13. Forward to Voice Mail System

1791

### 13.1 Symbol: `celf_mp_cs_call_forward_voice_msg`

1792

#### 13.1.1 Syntax

1793

```
CelfMpStatus celf_mp_cs_call_forward_voice_msg (
```

1794

```
    CelfMpCsBtype    com_type);
```

1795

1796

#### 13.1.2 Argument

1797

**Name:** `com_type`

1798

**Type:** `CelfMpCsBtype`

1799

I/O: `I`

1800

**Description:**

1801

Communication type as defined in section 1.2.4.

1802

1803

#### 13.1.3 Return Value

1804

**Type:** `CelfMpStatus`

1805

**Description:**

1806

`celf_mp_cs_call_forward_voice_msg()` **shall** return one of the values defined:

1807

`CELF_MP_STATUS_OK:` successful completion

1808

`CELF_MP_STATUS_COM_TYPE_ERR:` Communication type is not valid

1809

`CELF_MP_STATUS_ERR:` Other unsuccessful completion.

1810

1811

#### 13.1.4 Include File

1812

`/usr/include/celf/mp_cs.h`

1813

1814

#### 13.1.5 Functional Description

1815

This function receives the request to forward a call to a voice mail system.

1816

1817

Because the function is an immediate return function, to confirm the complete result, including the negotiation with the network, it should be issued

1818

`celf_mp_cs_notification_status()` to obtain the communication status.

1819

1820

1821

The incoming call is forwarded to phone-answering message when the communication status is (a) under the incoming, (b) under conversation and incoming, or (c) under hold and incoming.

1822

1823

1824

1825

If the forwarding fails, incoming call is continued between other party and this phone.

DRAFT

1826

## 14.Call Hold

1827

### 14.1 Symbol: `celf_mp_cs_call_hold`

1828

#### 14.1.1 Syntax

1829

```
CelfMpStatus celf_mp_cs_call_hold (
```

1830

```
    CelfMpCsBtype    com_type);
```

1831

1832

#### 14.1.2 Argument

1833

**Name:** `com_type`

1834

**Type:** `CelfMpCsBtype`

1835

**I/O:** |

1836

**Description:**

1837

Communication type as defined in section 1.2.4.

1838

1839

#### 14.1.3 Return Value

1840

**Type:** `CelfMpStatus`

1841

**Description:**

1842

`celf_mp_cs_call_hold()` **shall** return one of the values defined:

1843

`CELF_MP_STATUS_OK:` successful completion

1844

`CELF_MP_STATUS_COM_TYPE_ERR:` Communication type is not valid

1845

`CELF_MP_STATUS_ERR:` Other unsuccessful completion.

1846

1847

#### 14.1.4 Include File

1848

`/usr/include/celf/mp_cs.h`

1849

1850

#### 14.1.5 Functional Description

1851

This function receives the requests response hold.

1852

1853

Because the function is an immediate return function, to confirm the complete result, including the negotiation with the network, it should be issued

1854

`celf_mp_cs_notification_status()` to obtain the communication status.

1855

1856

1857

This response hold is performed for an incoming call, only when the communication status

1858

is under incoming.

1859

1860  
1861

To release response hold (move to the under conversation status) call "Response to an incoming call".

DRAFT

1862 **15.Call Reject**

1863 **15.1 Symbol: celf\_mp\_cs\_call\_reject**

1864 **15.1.1 Syntax**

1865 CelfMpStatus celf\_mp\_cs\_call\_reject (  
1866 CelfMpCsBtype com\_type);  
1867

1868 **15.1.2 Argument**

1869 **Name:** com\_type

1870 **Type:** CelfMpCsBtype

1871 **I/O:** |

1872 **Description:**

1873 Communication type as defined in section 1.2.4.  
1874

1875 **15.1.3 Return Value**

1876 **Type:** CelfMpStatus

1877 **Description:**

1878 celf\_mp\_cs\_call\_reject() **shall** return one of the values defined:

1879 CELF\_MP\_STATUS\_OK: successful completion

1880 CELF\_MP\_STATUS\_COM\_TYPE\_ERR: Communication type is not valid

1881 CELF\_MP\_STATUS\_ERR: Other unsuccessful completion.  
1882

1883 **15.1.4 Include File**

1884 /usr/include/celf/mp\_cs.h  
1885

1886 **15.1.5 Functional Description**

1887 This function receives the request to reject an incoming call.  
1888

1889 Because the function is an immediate return function, to confirm the complete result,  
1890 including the negotiation with the network, it should be issued  
1891 celf\_mp\_cs\_notification\_start() to obtain the communication status.  
1892

1893 The operation for each communication status is as follows:

1894 incoming: Rejects an incoming call

1895 In conversation and incoming: Rejects an incoming call

1896 In hold and incoming: Rejects an incoming call

1897

In conversation, hold, and incoming:

Rejects an incoming call

DRAFT

1898 **16.Multi Party Call**

1899 **16.1 Symbol: celf\_mp\_cs\_mp\_call**

1900 **16.1.1 Syntax**

```
1901 CelfMpStatus celf_mp_cs_mp_call (  
1902     CelfMpCsBtype      com_type,  
1903     CelfMpCsMop        mode,  
1904     CelfMpCallRef      call_reference);  
1905
```

1906 **16.1.2 Argument**

1907 **Name:** com\_type

1908 **Type:** CelfMpCsBtype

1909 **I/O:** |

1910 **Description:**

1911 Communication type as defined in section 1.2.4.

1912

1913 **Name:** mode

1914 **Type:** CelfMpCsMop

1915 **I/O:** |

1916 **Description:**

1917 Operation type

1918 CELF\_MP\_CS\_MOP\_RSV\_DISC: Disconnect the hold call

1919 CELF\_MP\_CS\_MOP\_DISC\_AND\_RSP: Response after disconnection

1920 CELF\_MP\_CS\_MOP\_RSV\_AND\_RSP: Response after hold (including operation for  
1921 switching a call)

1922 CELF\_MP\_CS\_MOP\_CR\_DISC: Disconnect call specified by the call reference

1923

1924 **Name:** call\_reference

1925 **Type:** CelfMpCallRef

1926 **I/O:** |

1927 **Description:**

1928 Call reference of the call to be disconnected

1929 Valid only if CELF\_MP\_CS\_MOP\_CR\_DISC is specified for the second argument.

1930

1931

1932        **16.1.3 Return Value**

1933        **Type:**    CelfMpStatus

1934        **Description:**

1935        celf\_mp\_cs\_mp\_call() **shall** return one of the values defined:

1936        CELF\_MP\_STATUS\_OK:                    successful completion

1937        CELF\_MP\_STATUS\_COM\_TYPE\_ERR:        Communication type is not valid

1938        CELF\_MP\_STATUS\_ERR:                 Other unsuccessful completion.

1939

1940        **16.1.4 Include File**

1941        /usr/include/celf/mp\_cs.h

1942

1943        **16.1.5 Functional Description**

1944        This function receives the request to operate for each call, when communication is made  
1945        with multiple calls.

1946

1947        The operation is as follows depending on CELF\_MP\_CS\_MOP:

1948        - CELF\_MP\_CS\_MOP\_RSV\_DISC

1949        If a hold call exists, this hold call is disconnected.

1950

1951        - CELF\_MP\_CS\_MOP\_DISC\_AND\_RSP

1952        If a conversation call exists and if another call status is incoming or hold, the conversation  
1953        call transits to disconnect status and another call transits to conversation status.

1954        See detail below.

1955        (1) Under conversation and incoming

1956        This status is that 1st call is in conversation, and 2nd call is incoming.

1957        The result is that 1st call is released, and 2nd call is conversation.

1958        (2) Under conversation and hold

1959        This status is that 1st call is in conversation, and 2nd call is hold.

1960        The result is that 1st call is released, and 2nd call is conversation.

1961        (3) Under conversation, hold, and incoming

1962        This status is that 1st call is in conversation, that 2nd call is hold, and that 3rd call is  
1963        incoming.

1964        The result is that 1st call is released, that 2nd call maintains hold, and that 3rd call is  
1965        conversation.

1966        (4) Under response hold

1967        This status is not changed.

1968



Classification: *Circuit-Switched Service*

1969 - CELF\_MP\_CS\_MOP\_RSV\_AND\_RSP  
1970 If a conversation call exists and if another call status is incoming or hold,  
1971 the conversation call transits to hold status and another call transits to conversation  
1972 status.  
1973 See detail below.  
1974 (1) Under conversation and incoming  
1975 This status is that 1st call is in conversation, and 2nd call is incoming.  
1976 The result is that 1st call is hold, and 2nd call is conversation.  
1977 (2) Under conversation and hold  
1978 This status is that 1st call is in conversation, and 2nd call is hold.  
1979 The result is that 1st call is hold, and 2nd call is conversation.  
1980 (3) Under conversation, hold, and incoming  
1981 This status is that 1st call is in conversation, that 2nd call is hold, and that 3rd call is  
1982 incoming.  
1983 The result is that 1st call is hold, that 2nd call is in conversation, that 3rd call maintains  
1984 incoming.  
1985 (4) Under response hold  
1986 This status is that 1st call is hold.  
1987 The result is that 1st call is in conversation.  
1988  
1989 - CELF\_MP\_CS\_MOP\_CR\_DISC It is disconnect the call specified the call reference.  
1990  
1991 Because the function is an immediate return function, to confirm the complete result,  
1992 including the negotiation with the network, it should be issued  
1993 `celf_mp_cs_notification_start()` to obtain the communication status.

1994  
1995  
1996  
1997  
1998  
1999  
2000  
2001  
2002  
2003  
2004  
2005  
2006  
2007  
2008  
2009  
2010  
2011  
2012  
2013  
2014  
2015  
2016  
2017  
2018  
2019  
2020  
2021  
2022  
2023  
2024  
2025  
2026  
2027

## 17. On-Hook Originating

### 17.1 Symbol: `celf_mp_cs_originating_on_hook`

#### 17.1.1 Syntax

```
CelfMpStatus celf_mp_cs_originating_on_hook (  
    CelfMpAppId      app_id,  
    CelfMpCsConReq  con_req);
```

#### 17.1.2 Argument

**Name:** `app_id`

**Type:** `CelfMpAppId`

**I/O:** `I`

**Description:**

Application identifier.

**Name:** `con_req`

**Type:** `CelfMpCsConReq`

**I/O:** `I`

**Description:**

Communication request type as defined in section 1.2.4.

#### 17.1.3 Return Value

**Type:** `CelfMpStatus`

**Description:**

`celf_mp_cs_call_reject()` shall return one of the values defined:

<code>CELf_MP_STATUS_OK:</code>	successful completion
<code>CELf_MP_STATUS_APP_ID_ERR:</code>	Application ID is not valid
<code>CELf_MP_CS_ONHOOK_DENY:</code>	On-hook originating is impossible
<code>CELf_MP_CS_ONHOOK_STATUS_ERR:</code>	Error due to communication conflict
<code>CELf_MP_CS_ONHOOK_OB_CR:</code>	Excess of the maximum number of calls
<code>CELf_MP_STATUS_ERR:</code>	Other unsuccessful completion.

#### 17.1.4 Include File

`/usr/include/celf/mp_cs.h`

2028        **17.1.5 Functional Description**

2029            This function receives the request to start an outgoing call with the specified dial number.

2030            The communication status should be Standby.

2031

2032            The dial number is specified by "dial\_buf" and "subaddr\_buf" in the "con\_req" structure.

2033

2034            If the character string, "184" or "186", is placed at the head of dial data, this character  
2035            string is deleted.

2036            Whether the originating dial number is notified or not, it is identified by "notice".

2037

2038            The dial data and subaddress stores the following ASCII codes.

2039            1 : 0 x 31    2 : 0 x 32    3 : 0 x 33

2040            4 : 0 x 34    5 : 0 x 35    6 : 0 x 36

2041            7 : 0 x 37    8 : 0 x 38    9 : 0 x 39

2042            \* : 0 x 2a    0 : 0 x 30    # : 0 x 23

2043

2044            Because the function is an immediate return function, to confirm the complete result,  
2045            including the negotiation with the network, it should be issued  
2046            `celf_mp_cs_notification_start()` to obtain the communication status.

2047

2048            The originating request during low voltage is disabled.

2049 **18.Get Call Reference**

2050 **18.1 Symbol: celf\_mp\_cs\_get\_call\_reference**

2051 **18.1.1 Syntax**

```
2052 CelfMpStatus celf_mp_cs_get_call_reference (  
2053     CelfMpAppId      app_id,  
2054     CelfMpCsChanNum  channel_num);  
2055
```

2056 **18.1.2 Argument**

2057 **Name:** app\_id

2058 **Type:** CelfMpAppId

2059 **I/O:** I

2060 **Description:**

2061 Application identifier.

2062

2063 **Name:** channel\_num

2064 **Type:** CelfMpCsChanNum

2065 **I/O:** O

2066 **Description:**

2067 Channel number information as defined in section 1.2.4.

2068

2069 **18.1.3 Return Value**

2070 **Type:** CelfMpStatus

2071 **Description:**

2072 `celf_mp_cs_get_call_reference()` **shall** return one of the values defined:

2073 CELF\_MP\_STATUS\_OK: successful completion

2074 CELF\_MP\_STATUS\_APP\_ID\_ERR: Application ID is not valid

2075 CELF\_MP\_STATUS\_ERR: Other unsuccessful completion.

2076

2077 **18.1.4 Include File**

2078 `/usr/include/celf/mp_cs.h`

2079

2080 **18.1.5 Functional Description**

2081 This function gets the call reference in use.

2082

Classification: *Circuit-Switched Service*

- 2083            A value within 0 to 255 is set to "ChanNum\_00", "ChanNum\_01" and "ChanNum\_02". If  
2084            channel is not used, CELF\_MP\_CS\_CHAN\_NOUSE is set as the call reference.
- 2085
- 2086            Three channels correspond to three calls in multiple calls.

DRAFT

2087 **19.Start DCF message notification**

2088 **19.1 Symbol: celf\_mp\_cs\_DCF\_notification\_start**

2089 **19.1.1 Syntax**

```
2090 CelfMpStatus celf_mp_cs_DCF_notification_start (  
2091     CelfMpAppld      app_id,  
2092     CelfMpDCFSet     event_set,  
2093     CelfMpCallback   callback_func);  
2094
```

2095 **19.1.2 Argument**

2096 **Name:** app\_id

2097 **Type:** CelfMpAppld

2098 **I/O:** |

2099 **Description:**

2100 Application identifier.

2102 **Name:** event\_set

2103 **Type:** CelfMpCsDCFSet

2104 **I/O:** |

2105 **Description:**

2106 Notification event set. Events that are classified as belonging to one of the  
2107 CelfMpCsDCFSet class **may** be registered to have a callback function called when the  
2108 event occurs for the application identified by app\_id. Classes of events are enabled by  
2109 setting the corresponding bit in event\_set:

2110 The event classes are defined as follows:

2112	CELF_MP_CS_DCF_DISP	Display-related message
2113	CELF_MP_CS_DCF_HISTORY	History-related message
2114	CELF_MP_CS_DCF_TONE1	Tone 1-related message
2115	CELF_MP_CS_DCF_TONE2	Tone 2-related message
2116	CELF_MP_CS_DCF_ETC	Other messages
2117	CELF_MP_CS_CLASS_ALL	All notified

2118

2119 A callback **may** be registered for all classes of events using special event class  
2120 CELF\_MP\_CS\_CLASS\_ALL, however to reduce overhead it is recommended that only the  
2121 needed event classes **should** be registered.

2122

2123 **Name:** callback\_func

2124       **Type:**    CelfMpCallback

2125       **I/O:**        |

2126       **Description:**

2127       The callback function, which **shall** be called when an event occurs from one of the classes  
2128       in `event_set`.

2129

### 2130    19.1.3 Return Value

2131       **Type:**    CelfMpStatus

2132       **Description:**

2133       `celf_mp_cs_DCF_notification_start ()` **shall** return one of the values defined:

2134       CELF\_MP\_STATUS\_OK:               successful completion

2135       CELF\_MP\_STATUS\_APP\_ID\_ERR:       Application ID is not valid

2136       CELF\_MP\_STATUS\_ERR:             Other unsuccessful completion.

2137

### 2138    19.1.4 Include File

2139       `/usr/include/celf/mp_cs.h`

2140

### 2141    19.1.5 Functional Description

2142       This function starts the monitoring the DCF message on the voice communication or AV  
2143       communication.

2144

2145       The occurrence of the event is notified to the application, specified by `app_id`.

2146

2147       The messages to be notified are described below.

2148

2149       Display-related message:

2150       -Notification of starting display during CCP outgoing

2151       -Notification of starting display during CCP incoming

2152       -Notification of starting display during CCP calling

2153       -Notification of starting display during CCP connecting

2154       -Notification of starting display during CCP communication

2155       -Notification of ending CCP   That is to notifies of release of a CCP call.

2156       -Notification of starting CCP disconnection (on the mobilephone) display

2157       -Notification of starting display of CCP disconnection (on the network) display

2158       -Notification of rejecting CCP outgoing

2159       -Notification of CCP hold

Classification: *Circuit-Switched Service*

- 2160 -Notification of releasing CCP hold
- 2161
- 2162 History-related message:
- 2163 -Notification of registering CCP outgoing call history
- 2164 -Notification of registering CCP absence incoming call history
- 2165 -Notification of registering CCP incoming call history
- 2166
- 2167 Tone 1-related message:(Tone sounding on the AP layer)
- 2168 -Notification of CCP RGT start
- 2169 -Notification of CCP RGT stop
- 2170 -Start report of incoming of a CCP hold call
- 2171 -Stop report of incoming of a CCP hold call
- 2172
- 2173 Tone 2-related message:(Tone sounding by the voice communication service)
- 2174 -Notification of CCP DST start
- 2175 -Notification of CCP DST stop
- 2176 -Notification of CCP RBT start
- 2177 -Notification of CCP RBT stop
- 2178 -Notification of CCP BT start
- 2179 -Notification of CCP CWT start
- 2180 -Notification of CCP CWT stop
- 2181
- 2182 Other messages:
- 2183 -Inquiry report of rejecting a CCP CS incoming call



2184 **20.Stop DCF message notification**

2185 **20.1 Symbol: celf\_mp\_cs\_DCF\_notification\_stop**

2186 **20.1.1 Syntax**

```
2187 CelfMpStatus celf_mp_cs_DCF_notification_stop (  
2188     CelfMpAppId      app_id,  
2189     CelfMpDCFSet     event_set);  
2190
```

2191 **20.1.2 Argument**

2192 **Name:** app\_id

2193 **Type:** CelfMpAppId

2194 **I/O:** I

2195 **Description:**

2196 Application identifier.

2198 **Name:** event\_set

2199 **Type:** CelfMpCsDCFSet

2200 **I/O:** I

2201 **Description:**

2202 Notification event set. Events that are classified as belonging to one of the  
2203 CelfMpCsDCFSet class. Classes of events are enabled by setting the corresponding bit in  
2204 event\_set:

2205  
2206 The event classes are defined as follows:

- |                             |                         |
|-----------------------------|-------------------------|
| 2207 CELF_MP_CS_DCF_DISP    | Display-related message |
| 2208 CELF_MP_CS_DCF_HISTORY | History-related message |
| 2209 CELF_MP_CS_DCF_TONE1   | Tone 1-related message  |
| 2210 CELF_MP_CS_DCF_TONE2   | Tone 2-related message  |
| 2211 CELF_MP_CS_DCF_ETC     | Other messages          |
| 2212 CELF_MP_CS_CLASS_ALL   | All notified            |

2213

2214 **20.1.3 Return Value**

2215 **Type:** CelfMpStatus

2216 **Description:**

2217 celf\_mp\_cs\_DCF\_notification\_stop() **shall** return one of the values defined:

2218 CELF\_MP\_STATUS\_OK: successful completion

2219            CELF\_MP\_STATUS\_APP\_ID\_ERR:        Application ID is not valid  
2220            CELF\_MP\_STATUS\_ERR:            Other unsuccessful completion.  
2221

#### 2222            20.1.4 Include File

2223            /usr/include/celf/mp\_cs.h

2224

#### 2225            20.1.5 Functional Description

2226            This function stops notifying of the DCF message on voice communication or AV  
2227            communication.

DRAFT

## 2228 21.Voice Message Notification

### 2229 21.1 Symbol: `celf_mp_cs_voice_msg_notify`

#### 2230 21.1.1 Syntax

```
2231 CelfMpStatus celf_mp_cs_voice_msg_notify (  
2232 CelfMpCsRecMsg rec_status);  
2233
```

#### 2234 21.1.2 Argument

2235 **Name:** `rec_status`

2236 **Type:** `CelfMpCsRecMsg`

2237 **I/O:** |

2238 **Description:**

2239 CELF\_MP\_CS\_REC\_MSG\_START: Start of a voice message

2240 CELF\_MP\_CS\_REC\_MSG\_STOP: Stop of a voice message

2241

#### 2242 21.1.3 Return Value

2243 **Type:** `CelfMpStatus`

2244 **Description:**

2245 `celf_mp_cs_call_voice_msg_notify()` **shall** return one of the values defined:

2246 CELF\_MP\_STATUS\_OK: successful completion

2247 CELF\_MP\_STATUS\_COM\_TYPE\_ERR: Communication type is not valid

2248 CELF\_MP\_STATUS\_ERR: Other unsuccessful completion.

2249

#### 2250 21.1.4 Include File

2251 `/usr/include/celf/mp_cs.h`

2252

#### 2253 21.1.5 Functional Description

2254 This function **must** be called before the communication state is changed to "under  
2255 conversation."

2256 After the start notification, a stop notification **must** be issued, when the voice message is  
2257 stopped.

2258 **22.Hold Tone Start**

2259 **22.1 Symbol: celf\_mp\_cs\_hold\_tone\_start**

2260 **22.1.1 Syntax**

2261 CelfMpStatus celf\_mp\_cs\_hold\_tone\_start (  
2262 CelfMpAppld app\_id);  
2263

2264 **22.1.2 Argument**

2265 **Name:** app\_id

2266 **Type:** CelfMpAppld

2267 **I/O:** I

2268 **Description:**

2269 Application identifier.  
2270

2271 **22.1.3 Return Value**

2272 **Type:** CelfMpStatus

2273 **Description:**

2274 celf\_mp\_cs\_hold\_tone\_start() **shall** return one of the values defined:

2275 CELF\_MP\_STATUS\_OK: successful completion

2276 CELF\_MP\_STATUS\_COM\_TYPE\_ERR: Communication type is not valid

2277 CELF\_MP\_STATUS\_APP\_ID\_ERR: Application ID is not valid

2278 CELF\_MP\_STATUS\_ERR: Other unsuccessful completion.  
2279

2280 **22.1.4 Include File**

2281 /usr/include/celf/mp\_cs.h  
2282

2283 **22.1.5 Functional Description**

2284 This function starts to sound a hold tone during a call.

2285

## 23.Hold Tone Stop

2286

### 23.1 Symbol: `celf_mp_cs_hold_tone_stop`

2287

#### 23.1.1 Syntax

2288

```
CelfMpStatus celf_mp_cs_hold_tone_stop (
```

2289

```
    CelfMpAppld app_id);
```

2290

2291

#### 23.1.2 Argument

2292

**Name:** `app_id`

2293

**Type:** `CelfMpAppld`

2294

**I/O:** `I`

2295

**Description:**

2296

Application identifier.

2297

2298

#### 23.1.3 Return Value

2299

**Type:** `CelfMpStatus`

2300

**Description:**

2301

`celf_mp_cs_hold_tone_stop()` **shall** return one of the values defined:

2302

`CELF_MP_STATUS_OK:` successful completion

2303

`CELF_MP_STATUS_COM_TYPE_ERR:` Communication type is not valid

2304

`CELF_MP_STATUS_APP_ID_ERR:` Application ID is not valid

2305

`CELF_MP_STATUS_ERR:` Other unsuccessful completion.

2306

2307

#### 23.1.4 Include File

2308

`/usr/include/celf/mp_cs.h`

2309

2310

#### 23.1.5 Functional Description

2311

This function stops to sound a hold tone during a call.

2312

2313

## 24. Get 64K / AV Communication Status

2314

### 24.1 Symbol: `celf_mp_cs_get_UD_com_stat`

2315

#### 24.1.1 Syntax

2316

```
CelfMpStatus celf_mp_cs_get_UD_com_stat (
```

2317

```
    CelfMpUDComStatus res_status);
```

2318

2319

#### 24.1.2 Argument

2320

**Name:** `res_status`

2321

**Type:** `CelfMpUDComStatus`

2322

**I/O:** `O`

2323

**Description:**

2324

returns one of the values defined:

2325

`CELF_MP_CS_UD_STOP:` Under stop

2326

`CELF_MP_CS_UD_RUN:` Under communication

2327

`CELF_MP_CS_UD_CALLED:` Under incoming

2328

`CELF_MP_CS_UD_CALLING:` Under outgoing

2329

`CELF_MP_CS_UD_DISCONNECT:` Under disconnection

2330

`CELF_MP_CS_UD_CALLING_ALERT:` Under calling

2331

`CELF_MP_CS_UD_HOLD:` Under hold

2332

`CELF_MP_CS_UD_ERR:` Error in UD Communication

2333

2334

#### 24.1.3 Return Value

2335

**Type:** `CelfMpStatus`

2336

**Description:**

2337

`celf_mp_cs_get_UD_com_stat()` **shall** return one of the values defined:

2338

`CELF_MP_STATUS_OK:` successful completion

2339

`CELF_MP_STATUS_ERR:` Other unsuccessful completion.

2340

2341

#### 24.1.4 Include File

2342

`/usr/include/celf/mp_cs.h`

2343

2344

#### 24.1.5 Functional Description

2345

This function refers to the communication status of 64K communication or AV communication.

## 2346 25. Get internal/external AV Communication Status

### 2347 25.1 Symbol: `celf_mp_cs_get_AV_com_stat`

#### 2348 25.1.1 Syntax

```
2349 CelfMpStatus celf_mp_cs_get_AV_com_stat (  
2350 CelfMpUDComStatus res_status );
```

2351

#### 2352 25.1.2 Argument

2353 **Name:** `res_status`

2354 **Type:** `CelfMpUDComStatus`

2355 **I/O:** `O`

2356 **Description:**

2357 returns one of the values defined:

2358 CELF_MP_CS_AV_IN_STOP:	Under stop
2359 CELF_MP_CS_AV_IN_RUN:	Under communication
2360 CELF_MP_CS_AV_IN_CALLED:	Under incoming
2361 CELF_MP_CS_AV_IN_CALLING:	Under outgoing
2362 CELF_MP_CS_AV_IN_DISCONNECT:	Under disconnection
2363 CELF_MP_CS_AV_IN_CALLING_ALERT:	Under calling
2364 CELF_MP_CS_UD_IN_HOLD:	Under hold
2365 CELF_MP_CS_AV_OUT_STOP:	Under stop
2366 CELF_MP_CS_AV_OUT_RUN:	Under communication
2367 CELF_MP_CS_AV_OUT_CALLED:	Under incoming
2368 CELF_MP_CS_AV_OUT_CALLING:	Under outgoing
2369 CELF_MP_CS_AV_OUT_DISCONNECT:	Under disconnection
2370 CELF_MP_CS_AV_OUT_CALLING_ALERT:	Under calling
2371 CELF_MP_CS_UD_OUT_HOLD:	Under hold

2372

#### 2373 25.1.3 Return Value

2374 **Type:** `CelfMpStatus`

2375 **I/O:** `O`

2376 **Description:**

2377 `celf_mp_cs_get_AV_com_stat()` **shall** return one of the values defined:

2378 CELF\_MP\_STATUS\_OK: successful completion

2379 CELF\_MP\_STATUS\_ERR: Other unsuccessful completion.

2380

2381

2382       25.1.4 Include File

2383            /usr/include/celf/mp\_cs.h

2384

2385       25.1.5 Functional Description

2386            This function refers to the communication status of internal or external AV communication.

DRAFT



## 2387 26. Get Communication Status

### 2388 26.1 Symbol: `celf_mp_cs_get_com_stat`

#### 2389 26.1.1 Syntax

```
2390 CelfMpStatus celf_mp_cs_get_com_stat (  
2391     CelfMpAppld      app_id,  
2392     CelfMpCsRcvScene * rcv_type,  
2393     CelfMpCsComStatus res_status);  
2394
```

#### 2395 26.1.2 Argument

2396 **Name:** `app_id`

2397 **Type:** `CelfMpAppld`

2398 **I/O:** `I`

2399 **Description:**

2400 Application identifier.

2401

2402 **Name:** `rcv_type`

2403 **Type:** `CelfMpCsRcvType`

2404 **I/O:** `O`

2405 **Description:**

2406 Incoming call type:

2407 `CELF_MP_CS_RCV_TYPE_COMPETE_TRN:` Communication Conflict

2408 `CELF_MP_CS_RCV_TYPE_RSV_RETURN:` Reestablish held call

2409 `CELF_MP_CS_RCV_TYPE_CALL_BACK:` Network Call Back

2410 `CELF_MP_CS_RCV_TYPE_NORMAL:` Normal

2411 `CELF_MP_CS_RCV_TYPE_NONE:` No Incoming Call

2412

2413 **Name:** `res_status`

2414 **Type:** `CelfMpCsComStatus`

2415 **I/O:** `O`

2416 **Description:**

2417 returns one of the values defined:

2418 Current communication status

2419 `CELF_MP_CS_COM_STATUS_WAIT:` Standby

2420 `CELF_MP_CS_COM_STATUS_RCV:` Under incoming

2421 `CELF_MP_CS_COM_STATUS_TRN:` Under outgoing

Classification: *Circuit-Switched Service*

2422	CELF_MP_CS_COM_STATUS_DLV:	Under calling
2423	CELF_MP_CS_COM_STATUS_TLK:	Under conversation
2424	CELF_MP_CS_COM_STATUS_HLD:	Under response hold
2425	CELF_MP_CS_COM_STATUS_DUMMY1:	Under off-hook
2426	CELF_MP_CS_COM_STATUS_RLS:	Under release
2427	CELF_MP_CS_COM_STATUS_TLK_RCV:	Under conversation and incoming
2428	CELF_MP_CS_COM_STATUS_TLK_TRN:	Under conversation and outgoing
2429	CELF_MP_CS_COM_STATUS_TLK_DLV:	Under conversation and calling
2430	CELF_MP_CS_COM_STATUS_TLK_RSV:	Under conversation and hold
2431	CELF_MP_CS_COM_STATUS_TLK_RLS:	Under conversation and release
2432	CELF_MP_CS_COM_STATUS_TLK_RSV_RCV:	Under conversation, hold, and incoming
2433	CELF_MP_CS_COM_STATUS_RCV_AV:	Under incoming of an AV call
2434	CELF_MP_CS_COM_STATUS_TRN_AV:	Under outgoing of an AV call
2435	CELF_MP_CS_COM_STATUS_DLV_AV:	Under calling of an AV call
2436	CELF_MP_CS_COM_STATUS_TLK_AV:	Under conversation of an AV call
2437	CELF_MP_CS_COM_STATUS_HLD_AV:	Under response hold of an AV call
2438	CELF_MP_CS_COM_STATUS_RLS_AV:	Under release of an AV call
2439	CELF_MP_CS_COM_STATUS_DUMMY2:	Under AV off-hook
2440	CELF_MP_STATUS_APP_ID_ERR:	Application ID is not valid.
2441	CELF_MP_CS_ERR:	Abnormal end

### 26.1.3 Return Value

**Type:** CelfMpStatus

**I/O:** O

**Description:**

`celf_mp_cs_get_com_stat()` **shall** return one of the values defined:

CELF\_MP\_STATUS\_OK: successful completion

CELF\_MP\_STATUS\_ERR: Other unsuccessful completion.

### 26.1.4 Include File

`/usr/include/celf/mp_cs.h`

### 26.1.5 Functional Description

This function returns the incoming call status, when the current call is (a) under incoming status or (b) under conversation and incoming status.

2458 

## 27.Start Line Status Notification

2459 

### 27.1 Symbol: `celf_mp_cs_line_status_notification_start`

2460 

#### 27.1.1 Syntax

```

2461 CelfMpStatus celf_mp_cs_notification_start (
2462     CelfMpAppId      app_id,
2463     CelfMpCsMtype    event_set,
2464     CelfMpCallback   callback_func);

```

2465

2466 

#### 27.1.2 Argument

2467 **Name:** `app_id`2468 **Type:** `CelfMpAppId`2469 **I/O:** |2470 **Description:**

2471 Application identifier.

2472

2473 **Name:** `event_set`2474 **Type:** `CelfMpCsMtype`2475 **I/O:** |2476 **Description:**

2477 Notification event set. Events that are classified as belonging to one of the  
 2478 `CelfMpCsNotifySet` class **may** be registered to have a callback function called when  
 2479 the event occurs for the application identified by `app_id`. Classes of events are enabled by  
 2480 setting the corresponding bit in `event_set`.

2481 `CELF_MP_CS_MONITOR_LINE_STATUS:` Line status change notification2482 `CELF_MP_CS_MONITOR_RESTRICT:` Restriction status change notification2483 `CELF_MP_CS_MONITOR_RSSI:` Receive level change notification2484 `CELF_MP_CS_MONITOR_ALL:` All notified

2485

2486 **Name:** `callback_func`2487 **Type:** `CelfMpCallback`2488 **I/O:** |2489 **Description:**

2490 The callback function, which **shall** be called when an event occurs from one of the classes  
 2491 in `event_set`.

2492

2493        **27.1.3 Return Value**

2494        **Type:**    CelfMpStatus

2495        **Description:**

2496        `celf_mp_cs_notification_start()` **shall** return one of the values defined:

2497        CELF\_MP\_STATUS\_OK:                    successful completion

2498        CELF\_MP\_STATUS\_APP\_ID\_ERR:        Application ID is not valid

2499        CELF\_MP\_STATUS\_MON\_TYPE\_ERR:      Monitor type is not valid

2500        CELF\_MP\_STATUS\_ERR:                Other unsuccessful completion.

2501

2502        **27.1.4 Include File**

2503        `/usr/include/celf/mp_cs.h`

2504

2505        **27.1.5 Functional Description**

2506        This function starts the monitoring the line status.

2507        The occurrence of the event is notified to the application, specified by `app_id`.

2508        The events to be notified are described below.

2509

2510        1. Line status change notification:

2511        This event notifies that the line status is changed.

2512        The line status is the out-of-communication area status and the within-communication  
2513        area.

2514

2515        2. Restriction status change notification:

2516        This event notifies that a restriction status is changed.

2517        The restriction means that the incoming call or the outgoing call is restricted by the network  
2518        due to traffic congestion.

2519

2520        3. Receive level change notification:

2521        This event notifies that the receive level is changed.

2522        The receive level is the intensity of electromagnetic wave. The intensity is four levels,  
2523        high, mid, low and zero (out of area).

2524

2525        See section 1. for structure definitions and values.

2526

2527 **28.Stop Line Status Notification**

2528 **28.1 Symbol: celf\_mp\_cs\_line\_status\_notification\_stop**

2529 **28.1.1 Syntax**

```
2530 CelfMpStatus celf_mp_cs_notification_stop (
2531     CelfMpAppId      app_id,
2532     CelfMpCsMtype    event_set);
2533
```

2534 **28.1.2 Argument**

2535 **Name:** app\_id

2536 **Type:** CelfMpAppId

2537 **I/O:** |

2538 **Description:**

2539 Application identifier.

2540

2541 **Name:** event\_set

2542 **Type:** CelfMpCsMtype

2543 **I/O:** |

2544 **Description:**

2545 Mask of the events for which reporting is to be stopped.

2546 Notification event set. Events that are classified as belonging to one of the  
 2547 CelfMpCsNotifySet class **may** be registered to have a callback function called when  
 2548 the event occurs for the application identified by app\_id. Classes of events are enabled by  
 2549 setting the corresponding bit in event\_set:

- 2550 CELF\_MP\_CS\_MONITOR\_LINE\_STATUS: Line status change notification
- 2551 CELF\_MP\_CS\_MONITOR\_RESTRICT: Restriction status change notification
- 2552 CELF\_MP\_CS\_MONITOR\_RSSI: Received signal strength change
- 2553 notification
- 2554 CELF\_MP\_CS\_MONITOR\_ALL: All notified

2555

2556 **28.1.3 Return Value**

2557 **Type:** CelfMpStatus

2558 **Description:**

2559 celf\_mp\_cs\_notification\_stop() **shall** return one of the values defined:

- 2560 CELF\_MP\_STATUS\_OK: successful completion
- 2561 CELF\_MP\_STATUS\_APP\_ID\_ERR: Application ID is not valid
- 2562 CELF\_MP\_STATUS\_MON\_TYPE\_ERR: Monitor type is not valid

2563            CELF\_MP\_STATUS\_ERR:            Other unsuccessful completion.

2564

#### 2565            28.1.4 Include File

2566            /usr/include/celf/mp\_cs.h

2567

#### 2568            28.1.5 Functional Description

2569            This function ends notifying on the event of the line status.

2570

DRAFT

2571 **29. Get Reception Level**

2572 **29.1 Symbol: celf\_mp\_cs\_get\_reception\_level**

2573 **29.1.1 Syntax**

2574 CelfMpStatus celf\_mp\_cs\_get\_reception\_level (  
2575 CelfMpReceptionLevel result);  
2576

2577 **29.1.2 Argument**

2578 **Name:** result

2579 **Type:** CelfMpReceptionLevel

2580 **I/O:** O

2581 **Description:**

2582 returns one of the values defined:

2583 CELF\_MP\_CS\_RSSI\_LEVEL\_NONE : No reception

2584 CELF\_MP\_CS\_RSSI\_LEVEL\_LOW : Receive level (Lowest)

2585 CELF\_MP\_CS\_RSSI\_LEVEL\_MEDIUM1 : Receive level

2586 CELF\_MP\_CS\_RSSI\_LEVEL\_MEDIUM2 : Receive level

2587 CELF\_MP\_CS\_RSSI\_LEVEL\_HIGH : Receive level (Highest)  
2588

2589 **29.1.3 Return Value**

2590 **Type:** CelfMpStatus

2591 **I/O:** O

2592 **Description:**

2593 `celf_mp_cs_get_reception_level()` **shall** return one of the values defined:

2594 CELF\_MP\_STATUS\_OK: successful completion

2595 CELF\_MP\_STATUS\_APP\_ID\_ERR: Application ID is not valid

2596 CELF\_MP\_STATUS\_ERR: Other unsuccessful completion.  
2597

2598 **29.1.4 Include File**

2599 `/usr/include/celf/mp_cs.h`  
2600

2601 **29.1.5 Functional Description**

2602 This function obtains the current reception level.

2603 Without the line status monitoring by calling the "Start line status monitoring", it is possible  
2604 to get the status of reception level.  
2605

2606 **30. Get Line Status**

2607 **30.1 Symbol: celf\_mp\_cs\_get\_line\_status**

2608 **30.1.1 Syntax**

2609 CelfMpStatus celf\_mp\_cs\_get\_line\_status (  
2610 CelfMpCsAreaRefChgInf \* net);

2611

2612 **30.1.2 Argument**

2613 **Name:** net

2614 **Type:** CelfMpCsAreaRefChgInf

2615 **I/O:** |

2616 **Description:**

2617 Pointer to the struct used to hold line status information

2618

2619 **30.1.3 Return Value**

2620 **Type:** CelfMpStatus

2621 **Description:**

2622 celf\_mp\_cs\_get\_line\_status() **shall** return one of the values defined:

2623 CELF\_MP\_STATUS\_OK: successful completion

2624 CELF\_MP\_STATUS\_ERR: Other unsuccessful completion.

2625

2626 **30.1.4 Include File**

2627 /usr/include/celf/mp\_cs.h

2628

2629 **30.1.5 Functional Description**

2630 This function obtains the current line status.

2631 Without the line status monitoring by calling the "Start line status monitoring", it is possible  
2632 to get the status of line status.

2633 See section 1. for further information.

2634



2635

## 31. Get Coverage Status

2636

### 31.1 Symbol: `celf_mp_cs_get_coverage_status`

2637

#### 31.1.1 Syntax

2638

```
CelfMpStatus celf_mp_cs_get_line_status (
```

2639

```
    CelfMpCsLineStatusEx*    net,
```

2640

```
    CelfMpCsCoverage         cover);
```

2641

2642

#### 31.1.2 Argument

2643

**Name:** net

2644

**Type:** CelfMpCsLineStatusEx

2645

**I/O:** |

2646

**Description:**

2647

Pointer to the struct used to hold line status information

2648

2649

**Name:** cover

2650

**Type:** CelfMpCsCoverage

2651

**I/O:** O

2652

**Description:**

2653

Within- or out-of communication area status

2654

CELF\_MP\_CS\_LINE\_STATUS\_IN: Within-communication area

2655

CELF\_MP\_CS\_LINE\_STATUS\_OUT: Out-of-communication area

2656

2657

#### 31.1.3 Return Value

2658

**Type:** CelfMpStatus

2659

**Description:**

2660

`celf_mp_cs_get_line_status()` **shall** return one of the values defined:

2661

CELF\_MP\_STATUS\_OK: successful completion

2662

CELF\_MP\_STATUS\_ERR: Other unsuccessful completion.

2663

2664

#### 31.1.4 Include File

2665

`/usr/include/celf/mp_cs.h`

2666

2667

### 31.1.5 Functional Description

2668

This function obtains the information on the current status of the within- and out-of-communication areas for current line.

2669

2670

(This function gets only information of inside or outside coverage area status.)

DRAFT

2671 **32. Get Voice Mail Information**

2672 **32.1 Symbol: celf\_mp\_cs\_get\_vm\_info**

2673 **32.1.1 Syntax**

2674 CelfMpStatus celf\_mp\_cs\_get\_vm\_info (  
2675 CelfMpCsVMNum \* vm\_num);

2676 **32.1.2 Argument**

2677 **Name:** vm\_num

2678 **Type:** CelfMpCsVMNum

2679 **I/O:** I

2680 **Description:**

2681 Address of the storage area of the number of stored phone-answering messages

2682

2683 **32.1.3 Return Value**

2684 **Type:** CelfMpStatus

2685 **Description:**

2686 celf\_mp\_cs\_get\_vm\_info() **shall** return one of the values defined:

2687 CELF\_MP\_STATUS\_OK: successful completion

2688 CELF\_MP\_STATUS\_ERR: Other unsuccessful completion.

2689

2690 **32.1.4 Include File**

2691 /usr/include/celf/mp\_cs.h

2692

2693 **32.1.5 Functional Description**

2694 This function obtains the storage status of phone-answering messages from nonvolatile  
2695 memory.

2696 The storage status is the number of message of phone-answering.

2697

2698 **33.Set Voice Mail Information**

2699 **33.1 Symbol: celf\_mp\_cs\_set\_vm\_info**

2700 **33.1.1 Syntax**

2701 CelfMpStatus celf\_mp\_cs\_set\_vm\_info (  
2702 CelfMpCsVMNum vm\_num);

2703

2704 **33.1.2 Argument**

2705 **Name:** vm\_num

2706 **Type:** CelfMpCsVMNum

2707 **I/O:** I

2708 **Description:**

2709 The number of stored phone-answering messages

2710

2711 **33.1.3 Return Value**

2712 **Type:** CelfMpStatus

2713 **Description:**

2714 celf\_mp\_cs\_set\_vm\_info() **shall** return one of the values defined:

2715 CELF\_MP\_STATUS\_OK: successful completion

2716 CELF\_MP\_STATUS\_ERR: Other unsuccessful completion.

2717

2718 **33.1.4 Include File**

2719 /usr/include/celf/mp\_cs.h

2720

2721 **33.1.5 Functional Description**

2722 This function sets the storage status of phone-answering message to non-volatile memory.

2723

2724

2725

2726

## 34. Get Call Selection

2727

### 34.1 Symbol: `celf_mp_cs_get_call_select`

2728

#### 34.1.1 Syntax

2729

```
CelfMpStatus celf_mp_cs_get_call_select (
```

2730

```
    CelfMpCallSelect    select);
```

2731

2732

#### 34.1.2 Argument

2733

**Name:** `select`

2734

**Type:** `CelfMpCallSelect`

2735

**I/O:** `O`

2736

**Description:**

2737

`CELF_MP_CS_INCOMING_VOICE_ANSWERING` Forward to the phone-answering message

2738

2739

`CELF_MP_CS_INCOMING_FORWARD` Forward

2740

`CELF_MP_CS_INCOMING_REJECT` Reject (disconnect)

2741

`CELF_MP_CS_INCOMING_NORMAL` Receipt of an incoming call (normal incoming)

2742

2743

2744

#### 34.1.3 Return Value

2745

**Type:** `CelfMpStatus`

2746

**Description:**

2747

`celf_mp_cs_get_call_select ()` shall return one of the values defined:

2748

`CELF_MP_STATUS_OK:` successful completion

2749

`CELF_MP_STATUS_ERR:` Other unsuccessful completion.

2750

2751

#### 34.1.4 Include File

2752

`/usr/include/celf/mp_cs.h`

2753

2754

#### 34.1.5 Functional Description

2755

This function obtains the incoming call information from non-volatile memory.

2756

Refer "Set incoming function selection"

2757

2758

2759

## 35.Set Call Selection

2760

### 35.1 Symbol: `celf_mp_cs_set_call_select`

2761

#### 35.1.1 Syntax

2762

```
CelfMpStatus celf_mp_cs_set_call_select (
```

2763

```
    CelfMpCallSelect    select);
```

2764

2765

#### 35.1.2 Argument

2766

**Name:** select

2767

**Type:** CelfMpCallSelect

2768

**I/O:** I

2769

**Description:**

2770

CELf\_MP\_CS\_INCOMING\_VOICE\_ANSWERING: Forward to the phone-answering message

2771

2772

CELf\_MP\_CS\_INCOMING\_FORWARD: Forward

2773

CELf\_MP\_CS\_INCOMING\_REJECT: Reject (disconnect)

2774

CELf\_MP\_CS\_INCOMING\_NORMAL: Receipt of an incoming call (normal incoming)

2775

2776

2777

#### 35.1.3 Return Value

2778

**Type:** CelfMpStatus

2779

**Description:**

2780

`celf_mp_cs_set_call_select()` shall return one of the values defined:

2781

CELf\_MP\_STATUS\_OK: successful completion

2782

CELf\_MP\_STATUS\_ERR: Other unsuccessful completion.

2783

2784

#### 35.1.4 Include File

2785

`/usr/include/celf/mp_cs.h`

2786

2787

#### 35.1.5 Functional Description

2788

This function sets the incoming call information to nonvolatile memory.

2789

When an incoming call arrives during conversation mode, it is possible to save this incoming call information.

2790

2791

2792

2793

## 36.Set Service Information

2794

### 36.1 Symbol: `celf_mp_cs_set_service_info`

2795

#### 36.1.1 Syntax

2796

```
CelfMpStatus celf_mp_cs_set_service_info (
```

2797

```
    CelfMpRegNum    reg_no,
```

2798

```
    CelfMpCsSrvData * data);
```

2799

2800

#### 36.1.2 Argument

2801

**Name:** `reg_no`

2802

**Type:** `CelfMpRegNum`

2803

**I/O:** `I`

2804

**Description:**

2805

Registration number: 1 to 10

2806

2807

**Name:** `data`

2808

**Type:** `CelfMpCsSrvData`

2809

**I/O:** `I`

2810

**Description:**

2811

Pointer to additional service data

2812

2813

#### 36.1.3 Return Value

2814

**Type:** `CelfMpStatus`

2815

**Description:**

2816

`celf_mp_cs_set_service_info()` **shall** return one of the values defined:

2817

`CELF_MP_STATUS_OK:` successful completion

2818

`CELF_MP_STATUS_ERR:` Other unsuccessful completion.

2819

2820

#### 36.1.4 Include File

2821

`/usr/include/celf/mp_cs.h`

2822

2823

#### 36.1.5 Functional Description

2824

This function registers the additional service information to the non-volatile memory,

2825

2826

The additional service information is the service name and Dial data for accessing the service.

2827

- 2828           The 'reg\_no' is used as the key for accessing this additional service.
- 2829           The value range is from 0 to 10.
- 2830           See section 1. for additional information.
- 2831

DRAFT



2832 **37. Get Service Information**

2833 **37.1 Symbol: celf\_mp\_cs\_get\_service\_info**

2834 **37.1.1 Syntax**

```
2835 CelfMpStatus celf_mp_cs_get_service_info (  
2836     CelfMpRegNum    reg_no,  
2837     CelfMpCsSrvData * data);  
2838
```

2839 **37.1.2 Argument**

2840 **Name:** reg\_no

2841 **Type:** CelfMpRegNum

2842 **I/O:** |

2843 **Description:**

2844 Registration number: 1 to 10

2845

2846 **Name:** data

2847 **Type:** CelfMpCsSrvData

2848 **I/O:** |

2849 **Description:**

2850 Pointer to additional service data

2851

2852 **37.1.3 Return Value**

2853 **Type:** CelfMpStatus

2854 **Description:**

2855 `celf_mp_cs_get_service_info()` **shall** return one of the values defined:

2856 CELF\_MP\_STATUS\_OK: successful completion

2857 CELF\_MP\_STATUS\_ERR: Other unsuccessful completion.

2858

2859 **37.1.4 Include File**

2860 /usr/include/celf/mp\_cs.h

2861

2862 **37.1.5 Functional Description**

2863 This function obtains additional service information, specified by "reg\_no", from non-volatile  
2864 memory.

2865

2866

See “Register additional service settings”.

DRAFT

2867 **38.Delete Service Information**

2868 **38.1 Symbol: celf\_mp\_cs\_del\_service\_info**

2869 **38.1.1 Syntax**

2870 CelfMpStatus celf\_mp\_cs\_del\_service\_info (  
2871 CelfMpRegNum reg\_no);  
2872

2873 **38.1.2 Argument**

2874 **Name:** reg\_no

2875 **Type:** CelfMpRegNum

2876 **I/O:** I

2877 **Description:**

2878 Registration number: 1 to 10  
2879

2880 **38.1.3 Return Value**

2881 **Type:** CelfMpStatus

2882 **Description:**

2883 celf\_mp\_cs\_del\_service\_info() **shall return one of the values defined:**

2884 CELF\_MP\_STATUS\_OK: successful completion

2885 CELF\_MP\_STATUS\_ERR: Other unsuccessful completion.  
2886

2887 **38.1.4 Include File**

2888 /usr/include/celf/mp\_cs.h  
2889

2890 **38.1.5 Functional Description**

2891 This function deletes the additional service information specified by "reg\_no" from non-  
2892 volatile memory.

2893 **39.Remove Service Information**

2894 **39.1 Symbol: celf\_mp\_cs\_remove\_all\_service\_info**

2895 **39.1.1 Syntax**

2896 CelfMpStatus celf\_mp\_cs\_remove\_all\_service\_info (  
2897 void);

2898 **39.1.2 Argument**

2899 None.

2900

2901 **39.1.3 Return Value**

2902 **Type:** CelfMpStatus

2903 **Description:**

2904 `celf_mp_cs_remove_all_service_info()` **shall** return one of the values defined:

2905 CELF\_MP\_STATUS\_OK: successful completion

2906 CELF\_MP\_STATUS\_ERR: Other unsuccessful completion.

2907

2908 **39.1.4 Include File**

2909 `/usr/include/celf/mp_cs.h`

2910

2911 **39.1.5 Functional Description**

2912 This function deletes all the additional service information from non-volatile memory.

## 2913 40.Set Response Message Settings

### 2914 40.1 Symbol: `celf_mp_cs_set_resp_msg`

#### 2915 40.1.1 Syntax

```
2916 CelfMpStatus celf_mp_cs_set_resp_msg (  
2917     CelfMpRegNum     reg_no,  
2918     CelfMpCsSrvData * data);
```

#### 2920 40.1.2 Argument

2921 **Name:** `reg_no`

2922 **Type:** `CelfMpRegNum`

2923 **I/O:** `|`

2924 **Description:**

2925 Registration number: 1 to 10

2927 **Name:** `data`

2928 **Type:** `CelfMpCsSrvData`

2929 **I/O:** `|`

2930 **Description:**

2931 Pointer to the additional response message data area.

#### 2933 40.1.3 Return Value

2934 **Type:** `CelfMpStatus`

2935 **Description:**

2936 `celf_mp_cs_set_resp_msg()` **shall** return one of the values defined:

2937 `CELF_MP_STATUS_OK:` successful completion

2938 `CELF_MP_STATUS_ERR:` Other unsuccessful completion.

#### 2940 40.1.4 Include File

2941 `/usr/include/celf/mp_cs.h`

2942

#### 2943 40.1.5 Functional Description

2944 This function registers the additional response message information for the additional  
2945 service to the non-volatile memory,

2946

Classification: *Circuit-Switched Service*

2947           When a additional service is activated, and corresponding message from the network is  
2948           received, this additional response message is sent to the network.  
2949  
2950           The additional response message information is the service name and Dial data, which is  
2951           response message to send the network.  
2952  
2953           The "reg\_no" is used as the key for accessing this additional response message.  
2954           The value range is from 0 to 10.  
2955  
2956           For information about the structures, see section 1.  
2957

DRAFT

## 2958 41. Get Response Message Settings

### 2959 41.1 Symbol: `celf_mp_cs_get_resp_msg`

#### 2960 41.1.1 Syntax

```
2961 CelfMpStatus celf_mp_cs_get_resp_msg (  
2962     CelfMpRegNum    reg_no,  
2963     CelfMpCsSrvData * data);  
2964
```

#### 2965 41.1.2 Argument

2966 **Name:** `reg_no`

2967 **Type:** `CelfMpRegNum`

2968 **I/O:** `I`

2969 **Description:**

2970 Registration number: 1 to 10

2971

2972 **Name:** `data`

2973 **Type:** `CelfMpCsSrvData`

2974 **I/O:** `I`

2975 **Description:**

2976 Pointer to the additional response message setting data area

2977

#### 2978 41.1.3 Return Value

2979 **Type:** `CelfMpStatus`

2980 **Description:**

2981 `celf_mp_cs_get_resp_msg()` shall return one of the values defined:

2982 `CELF_MP_STATUS_OK:` successful completion

2983 `CELF_MP_STATUS_ERR:` Other unsuccessful completion.

2984

#### 2985 41.1.4 Include File

2986 `/usr/include/celf/mp_cs.h`

2987

#### 2988 41.1.5 Functional Description

2989 This function obtains the additional response message information, specified by “`reg_no`”,  
2990 from non-volatile memory.

2991

2992

See “Register response message settings”.

DRAFT



2993 **42.Delete Response Message Settings**

2994 **42.1 Symbol: celf\_mp\_cs\_del\_resp\_msg**

2995 **42.1.1 Syntax**

2996 CelfMpStatus celf\_mp\_cs\_del\_resp\_msg (  
2997 CelfMpRegNum reg\_no);  
2998

2999 **42.1.2 Argument**

3000 **Name:** reg\_no

3001 **Type:** CelfMpRegNum

3002 **I/O:** I

3003 **Description:**

3004 Registration number: 1 to 10  
3005

3006 **42.1.3 Return Value**

3007 **Type:** CelfMpStatus

3008 **Description:**

3009 celf\_mp\_cs\_del\_resp\_msg() **shall** return one of the values defined:

3010 CELF\_MP\_STATUS\_OK: successful completion

3011 CELF\_MP\_STATUS\_ERR: Other unsuccessful completion.  
3012

3013 **42.1.4 Include File**

3014 /usr/include/celf/mp\_cs.h  
3015

3016 **42.1.5 Functional Description**

3017 This function deletes the additional response message information, specified by "reg\_no",  
3018 from non-volatile memory.

3019 **43.Remove All Response Message Settings**

3020 **43.1 Symbol: celf\_mp\_cs\_remove\_all\_resp\_msg**

3021 **43.1.1 Syntax**

3022 CelfMpStatus celf\_mp\_cs\_remove\_all\_resp\_msg (  
3023 void);  
3024

3025 **43.1.2 Argument**

3026 None.  
3027

3028 **43.1.3 Return Value**

3029 **Type:** CelfMpStatus

3030 **Description:**

3031 celf\_mp\_cs\_remove\_all\_resp\_msg() **shall** return one of the values defined:

3032 CELF\_MP\_STATUS\_OK: successful completion

3033 CELF\_MP\_STATUS\_ERR: Other unsuccessful completion.  
3034

3035 **43.1.4 Include File**

3036 /usr/include/celf/mp\_cs.h  
3037

3038 **43.1.5 Functional Description**

3039 This function removes from non-volatile memory all the additional response message  
3040 information.  
3041

## 3042 44.Set Reconnection Tone

### 3043 44.1 Symbol: `celf_mp_cs_set_reconnection_tone`

#### 3044 44.1.1 Syntax

```
3045 CelfMpStatus celf_mp_cs_set_reconnection_tone (  
3046 CelfMpCsReconnectionTone reconn);  
3047
```

#### 3048 44.1.2 Argument

3049 **Name:** reconn

3050 **Type:** CelfMpCsReconnectionTone

3051 **I/O:** I

3052 **Description:**

3053 Reconnection tone to be set

3054 CELF\_MP\_CS\_RECONN\_ON\_T\_OFF: Tone OFF

3055 CELF\_MP\_CS\_RECONN\_ON\_T\_LOW: Tone ON low tone

3056 CELF\_MP\_CS\_RECONN\_ON\_T\_HI: Tone ON high tone

#### 3058 44.1.3 Return Value

3059 **Type:** CelfMpStatus

3060 **Description:**

3061 `celf_mp_cs_set_reconnection_tone()` **shall** return one of the values defined:

3062 CELF\_MP\_STATUS\_OK: successful completion

3063 CELF\_MP\_STATUS\_ERR: Other unsuccessful completion.

#### 3065 44.1.4 Include File

3066 `/usr/include/celf/mp_cs.h`

3067

#### 3068 44.1.5 Functional Description

3069 This function sets the reconnection tone information to the non-volatile memory.

3070

3071 **45. Get Reconnection Tone**

3072 **45.1 Symbol: celf\_mp\_cs\_get\_reconnection\_tone**

3073 **45.1.1 Syntax**

3074 CelfMpStatus celf\_mp\_cs\_get\_reconnection\_tone (  
3075 CelfMpCsReconnectionTone result);  
3076

3077 **45.1.2 Argument**

3078 **Name:** result

3079 **Type:** CelfMpCsReconnectionTone

3080 **I/O:** O

3081 **Description:**

3082 returns one of the values defined:

3083 CELF\_MP\_CS\_RECONN\_ON\_T\_OFF: Tone OFF

3084 CELF\_MP\_CS\_RECONN\_ON\_T\_LOW: Tone ON low tone

3085 CELF\_MP\_CS\_RECONN\_ON\_T\_HI: Tone ON high tone  
3086

3087 **45.1.3 Return Value**

3088 **Type:** CelfMpStatus

3089 **Description:**

3090 celf\_mp\_cs\_get\_reconnection\_tone() **shall** return one of the values defined:

3091 CELF\_MP\_STATUS\_OK: successful completion

3092 CELF\_MP\_STATUS\_ERR: Other unsuccessful completion  
3093

3094 **45.1.4 Include File**

3095 /usr/include/celf/mp\_cs.h  
3096

3097 **45.1.5 Functional Description**

3098 This function gets the reconnection tone information to the non-volatile memory.

3099 **46. Get Noise Cancel**

3100 **46.1 Symbol: celf\_mp\_cs\_get\_noise\_cancel**

3101 **46.1.1 Syntax**

3102 CelfMpStatus celf\_mp\_cs\_get\_noise\_cancel (  
3103 CelfMpCsNoiseCancel result);  
3104

3105 **46.1.2 Argument**

3106 **Name:** result

3107 **Type:** CelfMpCsNoiseCancel

3108 **I/O:** O

3109 **Description:**

3110 returns one of the values defined:

3111 CELF\_MP\_CS\_ON: Noise canceller ON

3112 CELF\_MP\_CS\_OFF: Noise canceller OFF  
3113

3114 **46.1.3 Return Value**

3115 **Type:** CelfMpStatus

3116 **Description:**

3117 celf\_mp\_cs\_get\_noise\_cancel() **shall** return one of the values defined:

3118 CELF\_MP\_STATUS\_OK: successful completion

3119 CELF\_MP\_STATUS\_ERR: Other unsuccessful completion  
3120

3121 **46.1.4 Include File**

3122 /usr/include/celf/mp\_cs.h  
3123

3124 **46.1.5 Functional Description**

3125 This function gets the noise canceller status.  
3126

3127

## 47.Set Noise Cancel

3128

### 47.1 Symbol: `celf_mp_cs_set_noise_cancel`

3129

#### 47.1.1 Syntax

3130

```
CelfMpStatus celf_mp_cs_set_noise_cancel (
```

3131

```
    CelfMpCsNoiseCancel    mode);
```

3132

3133

#### 47.1.2 Argument

3134

**Name:** mode

3135

**Type:** CelfMpCsNoiseCancel

3136

**I/O:** I

3137

**Description:**

3138

Reconnection tone to be set

3139

CELF\_MP\_CS\_ON: Noise canceller ON

3140

CELF\_MP\_CS\_OFF: Noise canceller OFF

3141

3142

#### 47.1.3 Return Value

3143

**Type:** CelfMpStatus

3144

**Description:**

3145

`celf_mp_cs_set_noise_cancel()` shall return one of the values defined:

3146

CELF\_MP\_STATUS\_OK: successful completion

3147

CELF\_MP\_STATUS\_ERR: Other unsuccessful completion.

3148

3149

#### 47.1.4 Include File

3150

`/usr/include/celf/mp_cs.h`

3151

3152

#### 47.1.5 Functional Description

3153

This function sets the noise canceller off or on.

3154

## 3155 48. Get Call Quality Alarm

### 3156 48.1 Symbol: `celf_mp_cs_get_call_quality_alarm`

#### 3157 48.1.1 Syntax

```
3158 CelfMpStatus celf_mp_cs_get_call_quality_alarm (  
3159 CelfMpCsQualAlarm result);  
3160
```

#### 3161 48.1.2 Argument

3162 **Name:** result

3163 **Type:** CelfMpCsQualAlarm

3164 **I/O:** O

3165 **Description:**

3166 returns one of the values defined:

3167 CELF\_MP\_CS\_QUALITY\_ALM\_OFF: Quality alarm OFF

3168 CELF\_MP\_CS\_QUALITY\_ALM\_LOW: Quality alarm ON low tone

3169 CELF\_MP\_CS\_QUALITY\_ALM\_HI: Quality alarm ON high tone  
3170

#### 3171 48.1.3 Return Value

3172 **Type:** CelfMpStatus

3173 **Description:**

3174 `celf_mp_cs_get_call_quality_alarm()` **shall** return one of the values defined:

3175 CELF\_MP\_STATUS\_OK: successful completion

3176 CELF\_MP\_STATUS\_ERR: Other unsuccessful completion.  
3177

#### 3178 48.1.4 Include File

3179 `/usr/include/celf/mp_cs.h`  
3180

#### 3181 48.1.5 Functional Description

3182 This function gets the status of the call quality alarm sound.

3183

## 49.Set Call Quality Alarm

3184

### 49.1 Symbol: `celf_mp_cs_set_call_quality_alarm`

3185

#### 49.1.1 Syntax

3186

```
CelfMpStatus celf_mp_cs_set_call_quality_alarm (
```

3187

```
    CelfMpCsQualAlarm mode);
```

3188

3189

#### 49.1.2 Argument

3190

**Name:** mode

3191

**Type:** CelfMpCsQualAlarm

3192

**I/O:** I

3193

**Description:**

3194

CELF\_MP\_CS\_QUALITY\_ALM\_OFF: Quality alarm OFF

3195

CELF\_MP\_CS\_QUALITY\_ALM\_LOW: Quality alarm ON low tone

3196

CELF\_MP\_CS\_QUALITY\_ALM\_HI: Quality alarm ON high tone

3197

3198

#### 49.1.3 Return Value

3199

**Type:** CelfMpStatus

3200

**Description:**

3201

`celf_mp_cs_set_call_quality_alarm()` **shall** return one of the values defined:

3202

CELF\_MP\_STATUS\_OK: successful completion

3203

CELF\_MP\_STATUS\_ERR: Other unsuccessful completion.

3204

3205

#### 49.1.4 Include File

3206

`/usr/include/celf/mp_cs.h`

3207

3208

#### 49.1.5 Functional Description

3209

This function sets the call quality alarm sound.



## 3210 **50. Get Noise Cancel Permit**

### 3211 **50.1 Symbol: celf\_mp\_cs\_get\_noise\_cancel\_permit**

#### 3212 **50.1.1 Syntax**

```
3213 CelfMpStatus celf_mp_cs_get_noise_cancel_permit (  
3214 CelfMpCsNoiseCancel result);
```

3215

#### 3216 **50.1.2 Argument**

3217 **Name:** result

3218 **Type:** CelfMpCsNoiseCancel

3219 **I/O:** O

3220 **Description:**

3221 returns one of the values defined:

3222 CELF\_MP\_CS\_ON: Noise canceller permission

3223 CELF\_MP\_CS\_OFF: Noise canceller non-permission

3224

#### 3225 **50.1.3 Return Value**

3226 **Type:** CelfMpStatus

3227 **Description:**

3228 `celf_mp_cs_get_noise_cancel_permit()` **shall** return one of the values defined:

3229 CELF\_MP\_STATUS\_OK: successful completion

3230 CELF\_MP\_STATUS\_ERR: Other unsuccessful completion.

3231

#### 3232 **50.1.4 Include File**

3233 `/usr/include/celf/mp_cs.h`

3234

#### 3235 **50.1.5 Functional Description**

3236 This function obtains whether noise canceller is permitted or not.

3237

## 51. Get High Priority communication mode

3238

### 51.1 Symbol: `celf_mp_cs_get_hi_prio_com`

3239

#### 51.1.1 Syntax

3240

```
CelfMpStatus celf_mp_cs_get_hi_prio_com (
```

3241

```
    CelfMpCsHiPrioCom mode);
```

3242

3243

#### 51.1.2 Argument

3244

**Name:** mode

3245

**Type:** CelfMpCsHiPrioCom

3246

**I/O:** O

3247

**Description:**

3248

Connection priority currently set:

3249

CELF\_MP\_CS\_COMPRI\_NONE: No setting

3250

CELF\_MP\_CS\_COMPRI\_VOICE: Voice

3251

CELF\_MP\_CS\_COMPRI\_PACKET: Packet

3252

3253

#### 51.1.3 Return Value

3254

**Type:** CelfMpStatus

3255

**Description:**

3256

`celf_mp_cs_get_hi_prio_com()` shall return one of the values defined:

3257

CELF\_MP\_STATUS\_OK: successful completion

3258

CELF\_MP\_STATUS\_ERR: Other unsuccessful completion.

3259

3260

#### 51.1.4 Include File

3261

`/usr/include/celf/mp_cs.h`

3262

3263

#### 51.1.5 Functional Description

3264

This function returns the high priority communication mode setting in either to voice

3265

communication or packet communication. The priority gives the mandatory order of the

3266

processing.

3267

## 3268 52.Set High Priority communication mode

### 3269 52.1 Symbol: `celf_mp_cs_set_hi_prio_com`

#### 3270 52.1.1 Syntax

```
3271 CelfMpStatus celf_mp_cs_set_hi_prio_com (  
3272 CelfMpCsHiPrioCom mode);  
3273
```

#### 3274 52.1.2 Argument

3275 **Name:** mode

3276 **Type:** CelfMpCsHiPrioCom

3277 **I/O:** I

3278 **Description:**

3279 Connection priority to be chosen.

3280 CELF\_MP\_CS\_COMPRI\_NONE: No setting

3281 CELF\_MP\_CS\_COMPRI\_VOICE: Voice

3282 CELF\_MP\_CS\_COMPRI\_PACKET: Packet

3283

#### 3284 52.1.3 Return Value

3285 **Type:** CelfMpStatus

3286 **Description:**

3287 `celf_mp_cs_set_hi_prio_com()` shall return one of the values defined:

3288 CELF\_MP\_STATUS\_OK: successful completion

3289 CELF\_MP\_STATUS\_ERR: Other unsuccessful completion.

3290

#### 3291 52.1.4 Include File

3292 `/usr/include/celf/mp_cs.h`

3293

#### 3294 52.1.5 Functional Description

3295 This function sets the high priority communication mode either to voice communication or  
3296 packet communication. The priority gives the mandatory order of the processing.

3297

## 53. Get Phone Answering Sound Activation

3298

### 53.1 Symbol: `celf_mp_cs_get_vm_sound_status`

3299

#### 53.1.1 Syntax

3300

```
CelfMpStatus celf_mp_cs_get_vm_sound_status (
```

3301

```
    CelfMpCsVmSound  result);
```

3302

3303

#### 53.1.2 Argument

3304

**Name:** result

3305

**Type:** CelfMpCsVmSound

3306

**I/O:** O

3307

**Description:**

3308

returns one of the values defined:

3309

CELF\_MP\_CS\_ON: Message sound ON

3310

CELF\_MP\_CS\_OFF: Message sound OFF

3311

3312

#### 53.1.3 Return Value

3313

**Type:** CelfMpStatus

3314

**Description:**

3315

`celf_mp_cs_get_vm_sound_status()` **shall** return one of the values defined:

3316

CELF\_MP\_STATUS\_OK: successful completion

3317

CELF\_MP\_STATUS\_ERR: Other unsuccessful completion.

3318

3319

#### 53.1.4 Include File

3320

`/usr/include/celf/mp_cs.h`

3321

3322

#### 53.1.5 Functional Description

3323

This function gets the setting status.

3324

If the CelfMpCsVmSound is ON, the phone sounds, when the number of voice mail system increases.

3325

3326

## 54.Set Phone Answering Sound Activation

3327

### 54.1 Symbol: `celf_mp_cs_set_vm_sound_status`

3328

#### 54.1.1 Syntax

3329

```
CelfMpStatus celf_mp_cs_set_vm_sound_status (
```

3330

```
    CelfMpCsVmSound mode);
```

3331

3332

#### 54.1.2 Argument

3333

**Name:** mode

3334

**Type:** CelfMpCsVmSound

3335

**I/O:** I

3336

**Description:**

3337

CELF\_MP\_CS\_ON: Message sound ON

3338

CELF\_MP\_CS\_OFF: Message sound OFF

3339

3340

#### 54.1.3 Return Value

3341

**Type:** CelfMpStatus

3342

**Description:**

3343

`celf_mp_cs_set_vm_sound_status()` shall return one of the values defined:

3344

CELF\_MP\_STATUS\_OK: successful completion

3345

CELF\_MP\_STATUS\_ERR: Other unsuccessful completion.

3346

3347

#### 54.1.4 Include File

3348

`/usr/include/celf/mp_cs.h`

3349

3350

#### 54.1.5 Functional Description

3351

This function sets the phone sounds status whether the phone sounds or not.

3352

## 55. Get Automatic Receive Status

3353

### 55.1 Symbol: `celf_mp_cs_get_auto_rcv_status`

3354

#### 55.1.1 Syntax

3355

```
CelfMpStatus celf_mp_cs_get_auto_rcv_status (
```

3356

```
    CelfMpCsAutoRcv    mode);
```

3357

3358

#### 55.1.2 Argument

3359

**Name:** mode

3360

**Type:** CelfMpCsAutoRcv

3361

**I/O:** O

3362

**Description:**

3363

returns one of the values defined:

3364

CELFP\_MP\_CS\_ON: Automatic incoming call ON

3365

CELFP\_MP\_CS\_OFF: Automatic incoming call OFF

3366

3367

#### 55.1.3 Return Value

3368

**Type:** CelfMpStatus

3369

**Description:**

3370

`celf_mp_cs_get_auto_rcv_status()` **shall** return one of the values defined:

3371

CELFP\_MP\_STATUS\_OK: successful completion

3372

CELFP\_MP\_STATUS\_ERR: Other unsuccessful completion.

3373

3374

#### 55.1.4 Include File

3375

`/usr/include/celf/mp_cs.h`

3376

3377

#### 55.1.5 Functional Description

3378

This function obtains the status of automatic incoming call.

## 3379 56.Set Automatic Receive Status

### 3380 56.1 Symbol: `celf_mp_cs_set_auto_rcv_status`

#### 3381 56.1.1 Syntax

```
3382 CelfMpStatus celf_mp_cs_set_auto_rcv_status (  
3383 CelfMpCsAutoRcv mode);  
3384
```

#### 3385 56.1.2 Argument

3386 **Name:** mode

3387 **Type:** CelfMpCsAutoRcv

3388 **I/O:** I

3389 **Description:**

3390 CELF\_MP\_CS\_ON: Automatic incoming call ON

3391 CELF\_MP\_CS\_OFF: Automatic incoming call OFF

3392

#### 3393 56.1.3 Return Value

3394 **Type:** CelfMpStatus

3395 **Description:**

3396 `celf_mp_cs_set_auto_rcv_status()` **shall** return one of the values defined:

3397 CELF\_MP\_STATUS\_OK: successful completion

3398 CELF\_MP\_STATUS\_ERR: Other unsuccessful completion.

3399

#### 3400 56.1.4 Include File

3401 `/usr/include/celf/mp_cs.h`

3402

#### 3403 56.1.5 Functional Description

3404 This function sets the automatic incoming call status.

## 3405 57. Get Automatic Timer

### 3406 57.1 Symbol: `celf_mp_cs_get_auto_timer`

#### 3407 57.1.1 Syntax

```
3408 CelfMpStatus celf_mp_cs_get_auto_timer(  
3409 CelfMpCsTimer result);  
3410
```

#### 3411 57.1.2 Argument

3412 **Name:** result

3413 **Type:** CelfMpCsTimer

3414 **I/O:** O

3415 **Description:**

3416 returns one of the values defined:

3417 1 to 120 seconds  
3418

#### 3419 57.1.3 Return Value

3420 **Type:** CelfMpStatus

3421 **Description:**

3422 `celf_mp_cs_get_auto_timer()` shall return one of the values defined:

3423 CELF\_MP\_STATUS\_OK: successful completion

3424 CELF\_MP\_STATUS\_ERR: Other unsuccessful completion.  
3425

#### 3426 57.1.4 Include File

3427 `/usr/include/celf/mp_cs.h`  
3428

#### 3429 57.1.5 Functional Description

3430 This function obtains the timer value of the automatic incoming call.

3431 The timer value is the duration of sounding of the ring alert.



3432 **58.Set Automatic Timer**

3433 **58.1 Symbol: celf\_mp\_cs\_set\_auto\_timer**

3434 **58.1.1 Syntax**

3435 CelfMpStatus celf\_mp\_cs\_set\_auto\_timer (  
3436 CelfMpCsTimer time);  
3437

3438 **58.1.2 Argument**

3439 **Name:** time  
3440 **Type:** CelfMpCsTimer  
3441 **I/O:** I  
3442 **Description:**  
3443 1 to 120 seconds  
3444

3445 **58.1.3 Return Value**

3446 **Type:** CelfMpStatus  
3447 **Description:**  
3448 celf\_mp\_cs\_set\_auto\_timer() **shall** return one of the values defined:  
3449 CELF\_MP\_STATUS\_OK: successful completion  
3450 CELF\_MP\_STATUS\_ERR: Other unsuccessful completion.  
3451

3452 **58.1.4 Include File**

3453 /usr/include/celf/mp\_cs.h  
3454

3455 **58.1.5 Functional Description**

3456 This function sets the timer value of the automatic incoming call.  
3457  
3458  
3459  
3460

3461

## 59. Get Reset Date

3462

### 59.1 Symbol: `celf_mp_cs_get_reset_date`

3463

#### 59.1.1 Syntax

3464

```
CelfMpStatus celf_mp_cs_get_reset_date(  
3465
```

3466

```
    CelfMpCsDate *    reset_date);
```

3467

#### 59.1.2 Argument

3468

**Name:** `reset_date`

3469

**Type:** `CelfMpCsDate`

3470

**I/O:** `O`

3471

**Description:**

3472

Accumulated date record

3473

See section 1. for details.

3474

3475

3476

#### 59.1.3 Return Value

3477

**Type:** `CelfMpStatus`

3478

**Description:**

3479

`celf_mp_cs_get_reset_date()` **shall** return one of the values defined:

3480

`CELF_MP_STATUS_OK:` successful completion

3481

`CELF_MP_STATUS_ERR:` Other unsuccessful completion.

3482

3483

#### 59.1.4 Include File

3484

`/usr/include/celf/mp_cs.h`

3485

3486

#### 59.1.5 Functional Description

3487

This function obtains the date and time when the accumulated call duration was reset.

3488

The value is obtained from non-volatile memory.

3489

## 60.Set Reset Date

3490

### 60.1 Symbol: `celf_mp_cs_set_reset_date`

3491

#### 60.1.1 Syntax

3492

```
CelfMpStatus celf_mp_cs_set_reset_date(
```

3493

```
void);
```

3494

3495

#### 60.1.2 Argument

3496

None.

3497

3498

#### 60.1.3 Return Value

3499

**Type:** CelfMpStatus

3500

**Description:**

3501

`celf_mp_cs_set_reset_date()` **shall** return one of the values defined:

3502

CELF\_MP\_STATUS\_OK: successful completion

3503

CELF\_MP\_STATUS\_ERR: Other unsuccessful completion.

3504

3505

#### 60.1.4 Include File

3506

`/usr/include/celf/mp_cs.h`

3507

3508

#### 60.1.5 Functional Description

3509

This function sets the current date and time as the reset date and time of the accumulated call duration.

3510

3511

The value set to non-volatile memory.

3512

## 61.Get Call Silent Time

3513

### 61.1 Symbol: `celf_mp_cs_get_call_silent_time`

3514

#### 61.1.1 Syntax

3515

```
CelfMpStatus celf_mp_cs_get_call_silent_time(  
3516
```

3516

```
    CelfMpTime  time);  
3517
```

3517

3518

#### 61.1.2 Argument

3519

**Name:** time

3520

**Type:** CelfMpTime

3521

**I/O:** O

3522

**Description:**

3523

returns one of the values defined:

3524

0 to 99 seconds

3525

3526

#### 61.1.3 Return Value

3527

**Type:** CelfMpStatus

3528

**Description:**

3529

`celf_mp_cs_get_call_silent_time()` **shall** return one of the values defined:

3530

CELF\_MP\_STATUS\_OK: successful completion

3531

CELF\_MP\_STATUS\_ERR: Other unsuccessful completion.

3532

3533

#### 61.1.4 Include File

3534

`/usr/include/celf/mp_cs.h`

3535

3536

#### 61.1.5 Functional Description

3537

This function gets the duration between the arrival of incoming call and the start of sounding of the ring alert. This duration is called the silent time.

3538

3539

This function is effective that the number of this incoming call is unregistered with the phone book.

3540

3541 **62.Set Call Silent Time**

3542 **62.1 Symbol: celf\_mp\_cs\_set\_call\_silent\_time**

3543 **62.1.1 Syntax**

3544 CelfMpStatus celf\_mp\_cs\_set\_call\_silent\_time(  
3545 CelfMpCsTimer time);  
3546

3547 **62.1.2 Argument**

3548 **Name:** time  
3549 **Type:** CelfMpCsTimer  
3550 **I/O:** I  
3551 **Description:**  
3552 0 to 99 seconds  
3553

3554 **62.1.3 Return Value**

3555 **Type:** CelfMpStatus  
3556 **Description:**  
3557 `celf_mp_cs_set_call_silent_time()` **shall** return one of the values defined:  
3558 CELF\_MP\_STATUS\_OK: successful completion  
3559 CELF\_MP\_STATUS\_ERR: Other unsuccessful completion.  
3560

3561 **62.1.4 Include File**

3562 `/usr/include/celf/mp_cs.h`  
3563

3564 **62.1.5 Functional Description**

3565 This function sets the silent time.  
3566 Refer to get calling operation start time.

3567

## 63. Get Call Recorded

3568

### 63.1 Symbol: `celf_mp_cs_get_call_recorded`

3569

#### 63.1.1 Syntax

3570

```
CelfMpStatus celf_mp_cs_get_call_recorded(
```

3571

```
    CelfMpSetting    mode);
```

3572

3573

#### 63.1.2 Argument

3574

**Name:** mode

3575

**Type:** CelfMpSetting

3576

**I/O:** O

3577

**Description:**

3578

returns one of the values defined:

3579

CELF\_MP\_CS\_ON: Setting ON

3580

CELF\_MP\_CS\_OFF: Setting OFF

3581

3582

#### 63.1.3 Return Value

3583

**Type:** CelfMpStatus

3584

**Description:**

3585

`celf_mp_cs_get_call_recorded()` **shall** return one of the values defined:

3586

CELF\_MP\_STATUS\_OK: successful completion

3587

CELF\_MP\_STATUS\_ERR: Other unsuccessful completion.

3588

3589

#### 63.1.4 Include File

3590

`/usr/include/celf/mp_cs.h`

3591

3592

#### 63.1.5 Functional Description

3593

This function gets the setting condition of whether the silent call is recorded in the absent incoming call log, or not.

3594

3595

The absent incoming call log is the log that records no-responded incoming call.

3596

The silent call is the incoming call, which disconnects within the silent time.

3597

Refer to "Get calling operation start time".

3598

3599

3600

3601

## 64.Set Call Recorded

3602

### 64.1 Symbol: `celf_mp_cs_set_call_recorded`

3603

#### 64.1.1 Syntax

3604

```
CelfMpStatus celf_mp_cs_set_call_recorded(  
3605
```

3606

```
    CelfMpCsSetting    mode);
```

3606

3607

#### 64.1.2 Argument

3608

**Type:** CelfMpCsSetting

3609

**I/O:** I

3610

**Description:**

3611

CELF\_MP\_CS\_ON: Setting ON

3612

CELF\_MP\_CS\_OFF: Setting OFF

3613

3614

#### 64.1.3 Return Value

3615

**Type:** CelfMpStatus

3616

**Description:**

3617

`celf_mp_cs_set_call_recorded()` **shall** return one of the values defined:

3618

CELF\_MP\_STATUS\_OK: successful completion

3619

CELF\_MP\_STATUS\_ERR: Other unsuccessful completion.

3620

3621

#### 64.1.4 Include File

3622

`/usr/include/celf/mp_cs.h`

3623

3624

#### 64.1.5 Functional Description

3625

This function sets the setting condition of whether the silent call is recorded in the absent incoming call log or not.

3626

3627

Refer to "Get recording condition to absent incoming call log".

3628

## 65.Set Radio

3629

### 65.1 Symbol: `celf_mp_cs_set_radio`

3630

#### 65.1.1 Syntax

3631

```
CelfMpStatus celf_mp_cs_set_radio(  
3632     CelfMpCsSetting    mode);  
3633
```

3634

#### 65.1.2 Argument

3635

**Type:** CelfMpCsSetting

3636

**I/O:** I

3637

**Description:**

3638

CELF\_MP\_CS\_ON: Setting ON

3639

CELF\_MP\_CS\_OFF: Setting OFF

3640

3641

#### 65.1.3 Return Value

3642

**Type:** CelfMpStatus

3643

**Description:**

3644

`celf_mp_cs_set_radio()` shall return one of the values defined:

3645

CELF\_MP\_STATUS\_OK: successful completion

3646

CELF\_MP\_STATUS\_ERR: Other unsuccessful completion.

3647

3648

#### 65.1.4 Include File

3649

`/usr/include/celf/mp_cs.h`

3650

3651

#### 65.1.5 Functional Description

3652

This function sets the setting for deactivating or reactivating the Radio RF.



3653 **66.Get Radio Status**

3654 **66.1 Symbol: celf\_mp\_cs\_get\_radio**

3655 **66.1.1 Syntax**

3656 CelfMpStatus celf\_mp\_cs\_get\_radio(  
3657 CelfMpCsSetting mode);  
3658

3659 **66.1.2 Argument**

3660 **Type:** CelfMpCsSetting

3661 **I/O:** O

3662 **Description:**

3663 CELF\_MP\_CS\_ON: Setting ON

3664 CELF\_MP\_CS\_OFF: Setting OFF  
3665

3666 **66.1.3 Return Value**

3667 **Type:** CelfMpStatus

3668 **Description:**

3669 celf\_mp\_cs\_get\_radio() **shall** return one of the values defined:

3670 CELF\_MP\_STATUS\_OK: successful completion

3671 CELF\_MP\_STATUS\_ERR: Other unsuccessful completion.  
3672

3673 **66.1.4 Include File**

3674 /usr/include/celf/mp\_cs.h  
3675

3676 **66.1.5 Functional Description**

3677 This function gets the status the Radio RF.  
3678  
3679