EC2X&AG35-QuecOpen

ECALL API MANUAL

LTE Module Series

Rev. EC2X&AG35-QuecOpen_ECALL_API_guide_manual_V1.3

Date: 2018-04-04

Status: Temporary

Our aim is to provide customers with timely and comprehensive service. For any assistance, please contact our company headquarters:

Quectel Wireless Solutions Co., Ltd.
7th Floor, Hongye Building, No.1801 Hongmei Road, Xuhui District, Shanghai 200233, China
Tel: +86 21 5108 6236
Email: info@quectel.com

Or our local office. For more information, please visit:
http://quectel.com/support/sales.htm

For technical support, or to report documentation errors, please visit:
http://quectel.com/support/technical.htm
Or email to: support@quectel.com

GENERAL NOTES

QUECTEL OFFERS THE INFORMATION AS A SERVICE TO ITS CUSTOMERS. THE INFORMATION PROVIDED IS BASED UPON CUSTOMERS' REQUIREMENTS. QUECTEL MAKES EVERY EFFORT TO ENSURE THE QUALITY OF THE INFORMATION IT MAKES AVAILABLE. QUECTEL DOES NOT MAKE ANY WARRANTY AS TO THE INFORMATION CONTAINED HEREIN, AND DOES NOT ACCEPT ANY LIABILITY FOR ANY INJURY, LOSS OR DAMAGE OF ANY KIND INCURRED BY USE OF OR RELIANCE UPON THE INFORMATION. ALL INFORMATION SUPPLIED HEREIN IS SUBJECT TO CHANGE WITHOUT PRIOR NOTICE.

COPYRIGHT

About the Document

History

Revision	Date	Author	Description
1.0	2018-02-28	Laurence YIN	Initial
1.1	2018-03-17	Laurence YIN	Modification
1.2	2018-03-27	Laurence YIN	Updated
1.3	2018-04-04	Laurence YIN	Updated

Contents

About the Document **3**

Contents **4**

Figure Index **5**

1 Introduction **6**

2 ECALL API **8**

2.1. QL_Voice_Call_Client_Init 8
2.2. QL_Voice_Call_Client_Deinit 8
2.3. QL_Voice_Call_AddStateHandler 8
2.4. QL_Voice_Call_RemoveStateHandler 9
2.5. QL_Voice_Call_Ecall 9
2.6. QL_Voice_Call_Ecall_HangUp 10
2.7. QL_Voice_Call_AddCommonStateHandler 10
2.8. QL_Voice_Call_RemoveCommonStateHandler 10

3 Program Steps Of The Demo **11**

4 Execution of the demo **12**

4.1. Execute the command 12
4.2. Originating a ecall 12

5 eCall Compiling Introduction **14**

Figure Index

FIGURE 1: ECALL SYSTEM OVERVIEW 6
FIGURE 2: ECALL SYSTEM ARCHITECTURE 7

1 Introduction

This document mainly introduces how to use the eCall function of Quectel standard module. eCall function is only supported by the special software version.

eCall is defined as a manually or automatically initiated emergency call from a vehicle, supplemented with a minimum set of emergency related data (MSD), as defined under the EU Commission's eSafety initiative. It can be depicted by the figure below.

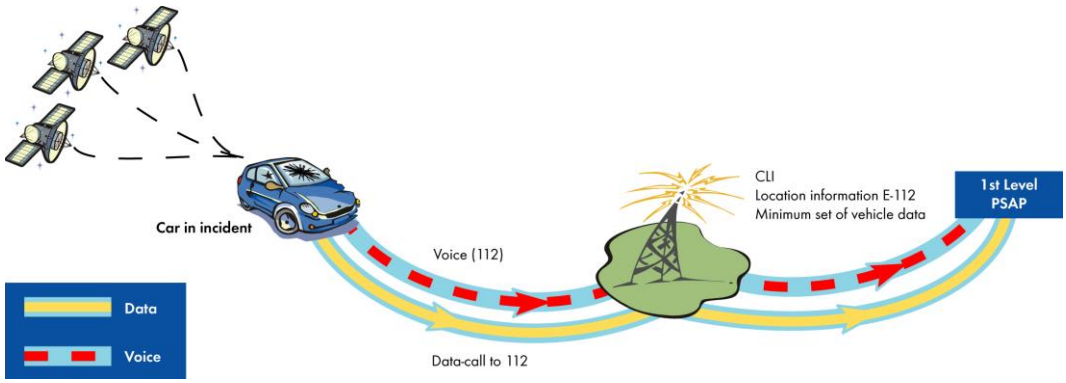


Figure 1: eCall System Overview

The architecture of eCall system is described in Figure 2. In Quectel test system, the module has the ability to act as IVS and also to simulate the PSAP. Thus, eCall testing can be easily performed by preparing two Quectel modules in the circumstance without access to a real PSAP. It will be described in the following chapters. Of course, if a real PSAP can be accessed, testing in the real environment is preferred.

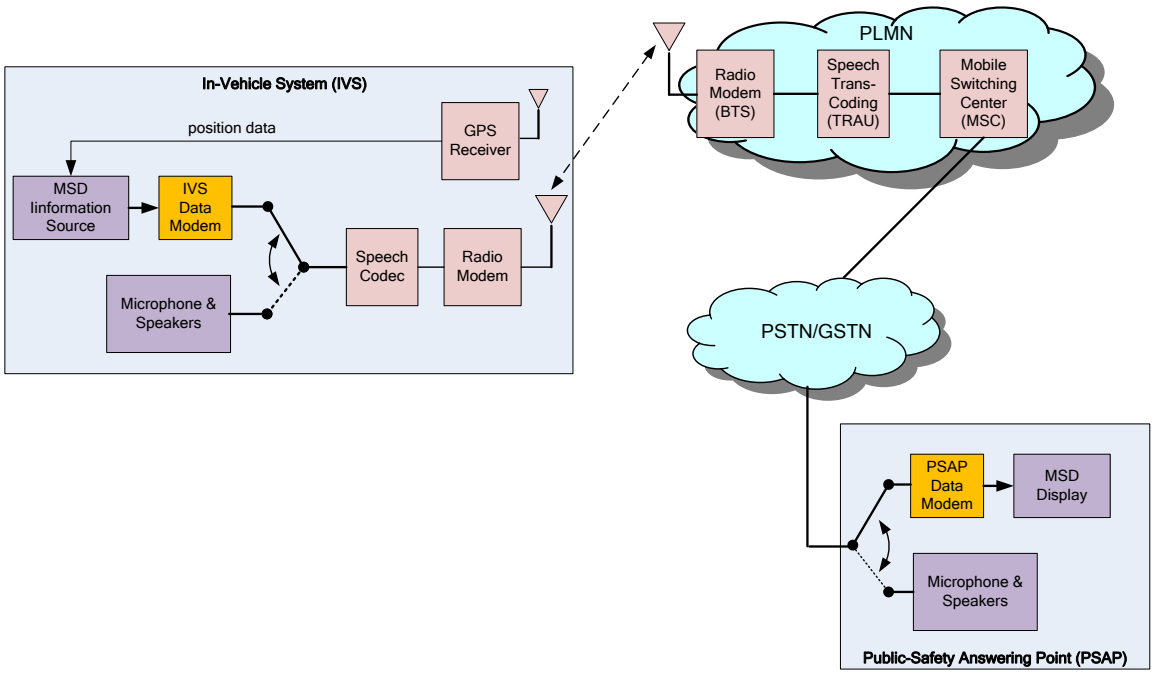


Figure 2: eCall System Architecture

2 ECALL API

2.1. QL_Voice_Call_Client_Init

(1) Function prototype:
int QL_Voice_Call_Client_Init(voice_client_handle_type *ph_voice);
(2) Parameter description:
1) ph_voice: OUT the pointer of voice handle
(3) Return description: int, 0-SUCCESS, Greater than 0- partial SUCCESS, Less than 0- FAILURE
(4) Functional description:
Init Voice function handle.

2.2. QL_Voice_Call_Client_Deinit

(1) Function prototype:
int QL_Voice_Call_Client_Deinit (voice_client_handle_type h_voice);
(2) Parameter description:
1) h_voice: IN voice handle
(3) Return description:int,0-SUCCESS, Greater than 0- partial SUCCESS,Less than 0- FAILURE
(4) Functional description:
Destroy related Voice feature resources

2.3. QL_Voice_Call_AddStateHandler

(1) Function prototype:
int QL_Voice_Call_AddStateHandler(voice_client_handle_type h_voice,
QL_VoiceCall_StateHandlerFunc_t handlerPtr,
void* contextPtr)
(2) Parameter description:
1) h_voice: IN voice handle
2) handlerPtr: IN voice callback function

3) contextPtr IN (the content of incoming call_id)

(3) Return description: int,0-SUCCESS, Greater than 0- partial SUCCESS, Less than 0- FAILURE

(4) Functional description:

Register the callback function to receive the incoming voice;

2.4. QL_Voice_Call_RemoveStateHandler

(1) Function prototype:

int QL_Voice_Call_RemoveStateHandler(voice_client_handle_type h_voice)

(2) Parameter description:

1) h_voice: IN voice handle

(3) Return description: int,0-SUCCESS, Greater than 0- partial SUCCESS, Less than 0- FAILURE

(4) Functional description:

Destroy the registered the callback function;

2.5. QL_Voice_Call_Ecall

(1) Function prototype:

```
int QL_Voice_Call_Ecall(voice_client_handle_type      h_voice,
                        E_QL_VCALL_ID_T               simId,
                        char*                         phone_number,
                        char*                         ecall_msd,
                        int                           manual,
                        E_QL_MCM_ECALL_VARIANT_T      eCallModeType,
                        int                           *call_id);
```

(2) Parameter description:

1) h_voice: IN voice handle
2) simId: INslot ID（**dumped params**）
3) phone_number: IN The called number
4) ecall_msd IN msd info
5) manual IN ecall voice type,0-auto;1-manual
6) eCallModeType IN ecall voice mode, E_QL_MCM_ECALL_TEST-ecall test mode; E_QL_MCM_ECALL_EMERGENCY-ecall normal; E_QL_MCM_ECALL_RECONFIG-no work
7) call_id: OUT call ID

(3) Return description: int,0-SUCCESS, Greater than 0- partial SUCCESS, Less than 0- FAILURE

(4) Functional description:

Originating-caller make a ecall；

2.6. QL_Voice_Call_Ecall_HangUp

(1) Function prototype:
int QL_Voice_Call_Ecall_HangUp (voice_client_handle_type h_voice)
(2) Parameter description:
1) h_voice: IN voice handle
(3) Return description: int,0-SUCCESS, Greater than 0- partial SUCCESS,Less than 0- FAILURE
(4) Functional description:
Hang up；

2.7. QL_Voice_Call_AddCommonStateHandler

(1) Function prototype:
int QL_Voice_Call_AddCommonStateHandler(voice_client_handle_type h_voice,
QL_VoiceCall_CommonStateHandlerFunc_t handlerPtr);

(2) Parameter description:
1) h_voice: IN voice handle
2) handlerPtr IN
(3) Return description: int,0-SUCCESS, Greater than 0- partial SUCCESS, Less than 0- FAILURE
(4) Functional description:
Register the callback function to receive the ind info;

2.8. QL_Voice_Call_RemoveCommonStateHandler

(1) Function prototype:
int QL_Voice_Call_RemoveCommonStateHandler(voice_client_handle_type h_voice);
(2) Parameter description:
1) h_voice: IN voice handle
(3) Return description: int,0-SUCCESS, Greater than 0- partial SUCCESS, Less than 0- FAILURE
(4) Functional description:
Destroy the registered the callback function;

3 Program Steps Of The Demo

Please refer to **example/ecall/example_ecall.c**

Description:

step1:QL_Voice_Call_Client_Init------ register voice client

step2:QL_Voice_Call_AddCommonStateHandler ----- register callback

step3: Communication

step4:QL_Voice_Call_RemoveCommonStateHandler ----destroy callback

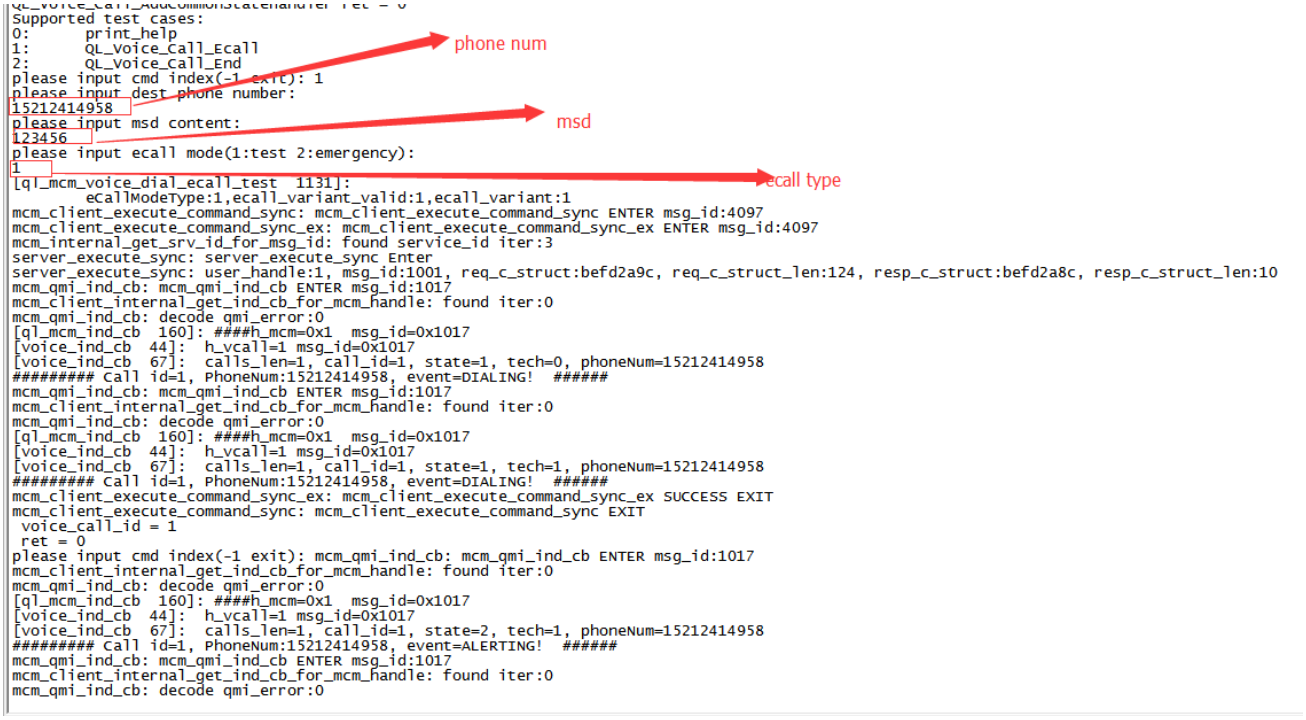
step5:QL_Voice_Call_Client_Deinit----- destroy client

4 Execution of the demo

4.1. Execute the command

```
/usrdata # ./example_ecall
```

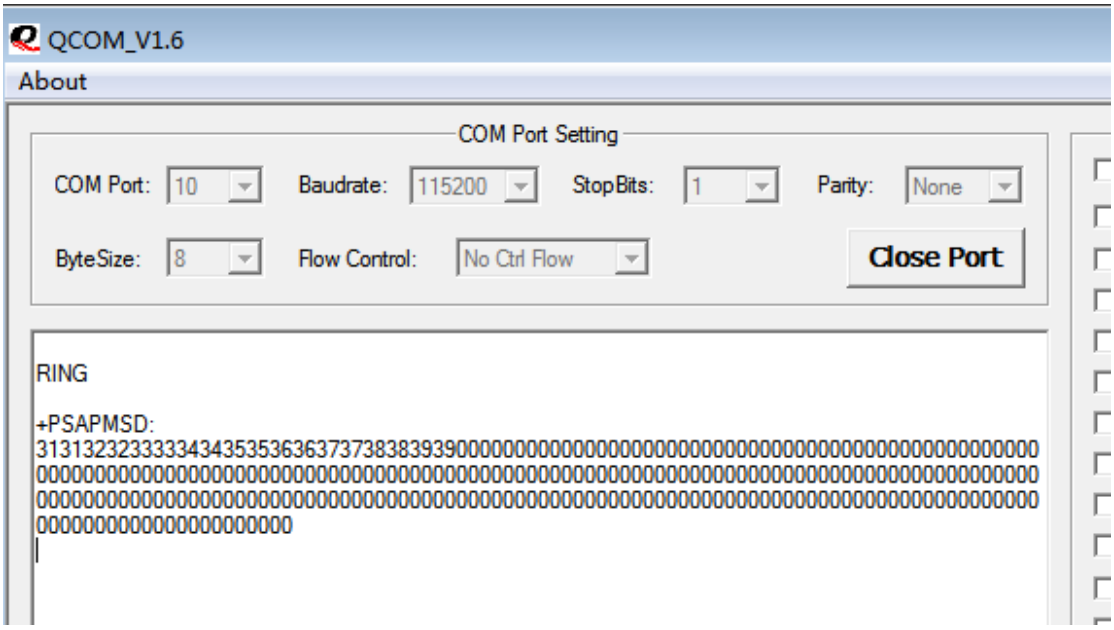
4.2. Originating a ecall

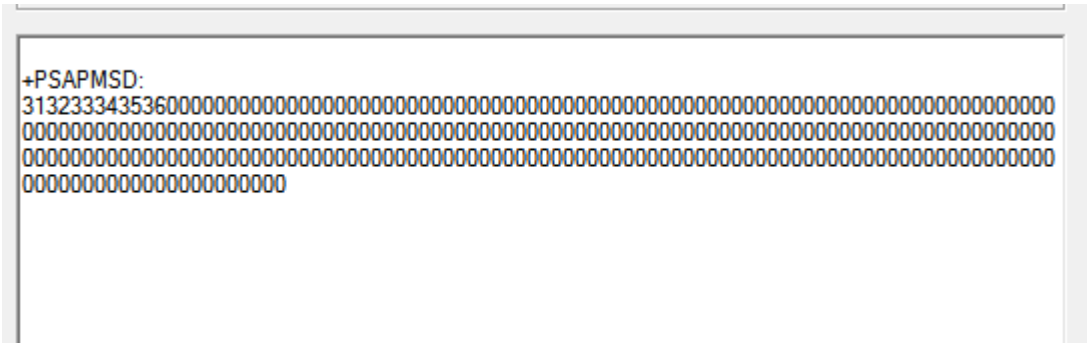



```
Supported test cases:
0:      print_help
1:      QL_Voice_Call_Ecall
2:      QL_Voice_Call_End
please input cmd index(-1 exit): 1
please input dest phone number:
15212414958
please input msd content:
123456
please input ecall mode(1:test 2:emergency):
1
[ql_mcm_voice_dial_ecall_test  1131]:
        eCallModeType:1,ecall_variant_valid:1,ecall_variant:1
mcm_client_execute_command_sync: mcm_client_execute_command_sync ENTER msg_id:4097
mcm_client_execute_command_sync_ex: mcm_client_execute_command_sync_ex ENTER msg_id:4097
mcm_internal_get_srv_id_for_msg_id: found service_id iter:3
server_execute_sync: server_execute_sync Enter
server_execute_sync: user_handle:1, msg_id:1001, req_c_struct:befd2a9c, req_c_struct_len:124, resp_c_struct:befd2a8c, resp_c_struct_len:10
mcm_qmi_ind_cb: mcm_qmi_ind_cb ENTER msg_id:1017
mcm_client_internal_get_ind_cb_for_mcm_handle: found iter:0
mcm_qmi_ind_cb: decode qmi_error:0
[ql_mcm_ind_cb  160]: ####h_mcm=0x1  msg_id=0x1017
[voice_ind_cb  44]:  h_vcall=1 msg_id=0x1017
[voice_ind_cb  67]:  calls_len=1, call_id=1, state=1, tech=0, phoneNum=15212414958
######### Call id=1, PhoneNum:15212414958, event=DIALING!  ######
mcm_qmi_ind_cb: mcm_qmi_ind_cb ENTER msg_id:1017
mcm_client_internal_get_ind_cb_for_mcm_handle: found iter:0
mcm_qmi_ind_cb: decode qmi_error:0
[ql_mcm_ind_cb  160]: ####h_mcm=0x1  msg_id=0x1017
[voice_ind_cb  44]:  h_vcall=1 msg_id=0x1017
[voice_ind_cb  67]:  calls_len=1, call_id=1, state=1, tech=1, phoneNum=15212414958
######### Call id=1, PhoneNum:15212414958, event=DIALING!  ######
mcm_client_execute_command_sync_ex: mcm_client_execute_command_sync_ex SUCCESS EXIT
mcm_client_execute_command_sync: mcm_client_execute_command_sync EXIT
 voice_call_id = 1
 ret = 0
please input cmd index(-1 exit): mcm_qmi_ind_cb: mcm_qmi_ind_cb ENTER msg_id:1017
mcm_client_internal_get_ind_cb_for_mcm_handle: found iter:0
mcm_qmi_ind_cb: decode qmi_error:0
[ql_mcm_ind_cb  160]: ####h_mcm=0x1  msg_id=0x1017
[voice_ind_cb  44]:  h_vcall=1 msg_id=0x1017
[voice_ind_cb  67]:  calls_len=1, call_id=1, state=2, tech=1, phoneNum=15212414958
######### Call id=1, PhoneNum:15212414958, event=ALERTING!  ######
mcm_qmi_ind_cb: mcm_qmi_ind_cb ENTER msg_id:1017
mcm_client_internal_get_ind_cb_for_mcm_handle: found iter:0
mcm_qmi_ind_cb: decode qmi_error:0
```

```
[voice_ind_cb  67]:  calls_len=1, call_id=1, state=1, tech=1, phoneNum=15212414958
######### Call id=1, PhoneNum:15212414958, event=DIALING!  ######
mcm_client_execute_command_sync_ex: mcm_client_execute_command_sync_ex SUCCESS EXIT
mcm_client_execute_command_sync: mcm_client_execute_command_sync EXIT
 voice_call_id = 1
 ret = 0
please input cmd index(-1 exit): mcm_qmi_ind_cb: mcm_qmi_ind_cb ENTER msg_id:1017
mcm_client_internal_get_ind_cb_for_mcm_handle: found iter:0
mcm_qmi_ind_cb: decode qmi_error:0
[ql_mcm_ind_cb  160]: ####h_mcm=0x1  msg_id=0x1017
[voice_ind_cb  44]:  h_vcall=1 msg_id=0x1017
[voice_ind_cb  67]:  calls_len=1, call_id=1, state=2, tech=1, phoneNum=15212414958
######### Call id=1, PhoneNum:15212414958, event=ALERTING!  ######
mcm_qmi_ind_cb: mcm_qmi_ind_cb ENTER msg_id:1017
mcm_client_internal_get_ind_cb_for_mcm_handle: found iter:0
mcm_qmi_ind_cb: decode qmi_error:0
[ql_mcm_ind_cb  160]: ####h_mcm=0x1  msg_id=0x1017
[voice_ind_cb  44]:  h_vcall=1 msg_id=0x1017
[voice_ind_cb  67]:  calls_len=1, call_id=1, state=3, tech=1, phoneNum=15212414958
######### Call id=1, PhoneNum:15212414958, event=ACTIVE!  ######
mcm_qmi_ind_cb: mcm_qmi_ind_cb ENTER msg_id:10a1
mcm_client_internal_get_ind_cb_for_mcm_handle: found iter:0
mcm_qmi_ind_cb: decode qmi_error:0
[ql_mcm_ind_cb  160]: ####h_mcm=0x1  msg_id=0x10a1
[voice_ind_cb  44]:  h_vcall=1 msg_id=0x10a1
[voice_ind_cb  188]:  Ecall status  call_id =1 ,   ecall msd tx success.
========== Ecall status  call_id =1 ,   ecall msd tx success.
```

PSAP receive msd success





5 eCall Compiling Introduction

This chapter is the introductions of compiling single example_ecall.c.

1. Unzip tar -jxvf ql-ol-sdk.tar.bz2: tar -jxvf ql-ol-sdk.tar.bz2
2. Enter ql-ol-sdk: cd cd ql-ol-sdk
3. EnsureSDK version is same as firmware version: source ql-ol-crosstool/ql-ol-crosstool-env-init
4. Execute: cd ql-ol-extsdk/example/ecall
5. Execute: make clean;make