

目录

1. 产品配网常见的问题	1
(1) 设备配网上线后, H5 页面显示的属性数值异常	1
(2) 设备配网上线后, H5 页面显示的属性数值与设备实际状态不同步	1
(3) 智慧生活 APP 添加设备时扫描不到设备, 但可以扫到模组发出的蓝牙广播	1
(4) 连接设备时注册失败, 提示该设备未经过华为官方认证, 该设备 MAC 非法	2
(5) 设备添加注册成功, 点击设备进入 H5 页面显示网络异常没有控制界面	3
2. SDK API 接口及相关功能使用问题	4
(1) PWM 相关接口的使用	4
(2) GPIO 模式配置及常见问题	4
(3) 重置配网接口 HILINK_BT_HardRevoke() 使用注意事项	5
(4) 操作用户区域 flash 相关 API 说明	5
(5) SDK 开启标准串口功能	7
3. 蓝牙、星闪相关问题	8
(1) 使用蓝牙注册 gatt 客户端时, 接口返回 80006006 异常码	8
(2) 启用蓝牙靠近发现是在什么地方, 如何修改靠近发现的距离?	8
(3) 不同品类设备我们的蓝牙名称有要求, 这些字段应该在哪改?	9
(4) 对蓝牙广播的时间有要求, 在哪里进行修改?	9
(5) 使用指令 AT+CONFIG 配置完成时, APP 搜索不到设备, 并且也扫描不到蓝牙广播	9
(6) 蓝牙使用自定义 PIN 码功能, 设置的通行码未生效, 只能弹出默认 PIN 码	9
(7) 在低功耗状态下, 模组做星闪/蓝牙从机时功耗偏高, 需要怎么解决	10
(8) 如何实时获取蓝牙的连接状态?	10
(9) 使用星闪连接时, MTU 大小有限制吗? 为什么设置时只允许 251 字节?	11
(10) 使用星闪保持长连接时, 进入低功耗状态, 在对应回调函数中增加连接间隔的数值来降低功耗, 功耗反正会增加, 是什么原因?	12
(11) 蓝牙在设置发包参数时返回异常码 0x80006007, 是什么原因?	12
4. 环境搭建和编译常见问题	13
(1) 编译报错编解码解析失败	13
5. SDK RAM 优化方案	14
(1) 方案简述	14
(2) 修改方法	14
(3) 修改验证	17

1. 产品配网常见的问题

(1) 设备配网上线后，H5 页面显示的属性数值异常

此情况是该属性的值未进行初始设置或设置的值不在取值范围内，对比检查 `BLE_SendCustomData()` 函数中上报的属性值是否符合 `Profile_XXXX.xlsx` 文件中的数据类型和取值范围，特别注意 `string` 类型的数据上报时需要加“”

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
设备类型	设备类型(中文)	服务 sid	服务(中文)	服务类型 ServiceType	属性	属性中文名称	属性英文名称	属性类型 Character Type	操作权限	数据类型	数据约束 (IT系统录入时使用)	取值范围	描述	
眼部按摩器			bt.connect 外部音频连接	bt.connection								0-连接		
					externalaud 外部音频连接	externalaud characteristic	bt GET/REPORT/PUT	enum		枚举-自定义枚举值	1-正在连接	2-新开		
childlockSw1	童锁开关	switch		on	童锁	childLockSw characteristic	on GET/REPORT/PUT	bool		枚举-固定枚举范围	0-关	1-开		
gear	按摩档位	gear			gear	档位	gear	characteristic.get GET/REPORT/PUT	enum	枚举-自定义枚举值	0-关	1-轻柔	2-舒缓	3-强力
globalVolume	全局音量	gear			globalVolum	全局音量	globalVolum characteristic	get GET/REPORT/PUT	enum	枚举-自定义枚举值	0-静音	1-一档	2-二档	3-三档
hotcompress1	热敷温度	gear			hotcompress	热敷温度	hotcompress characteristic	get GET/REPORT/PUT	enum	枚举-自定义枚举值	0-不加热	1-低温	2-中温	3-高温
node	模式	node			node	模式	Mode	characteristic.mode GET/REPORT/PUT	enum	枚举-自定义枚举值	0-轻度模式	1-中度模式	2-高度模式	3-暖宫健脾
switch	开关	switch	on	开关	on		characteristic.on	GET/REPORT/PUT	bool	枚举-固定枚举范围	0-关	1-开		

(2) 设备配网上线后，H5 页面显示的属性数值与设备实际状态不同步

此问题是模组没有及时上报相关数据，H5 连接成功时会下发 SID `allservices`，模组需要在接收到这个 `sid` 时来主动上报全量数据同步设备状态

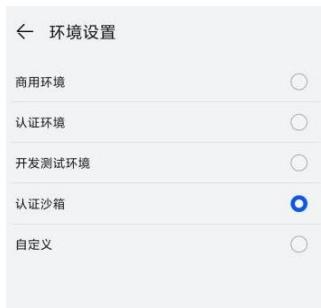
```
application > samples > hilink > ohos_connect > hilink_adapt > entry > C hilink_ble_main.c > set_CfgNetAdvCtrl_flag
306 static int BleRcvCustomData(unsigned char *buff, unsigned int len)
324 do {
331     if (!cJSON_IsString(sidItem) || (sidItem->valuestring == NULL)) {
333     }
334
335     if (strcmp(sidItem->valuestring, "allservices") == 0) {
336         /* H5 连接上时会給设备发送allservices，设备需要给H5同步全量状态 */
337         printf("sync dev status\r\n");
338         #ifdef _HSF_GENERAL_
339         // ReporAllSidStatus();
340         send_wifi_state_fun(HILINK_SERVER_CONNECT, "SERVER_CONNECT");
341         #endif
342
343         ret = 0;
344         break;
345     } else if (strcmp(sidItem->valuestring, "currentTime") == 0) {
346         // 参考格式 {"sid":"currentTime", "data":{"currentTime":1730692041}}
347         cJSON *item = cJSON_GetObjectItem(dataItem, "currentTime");
348         if (item != NULL) {
349             printf("sync time %d\r\n", item->valueint);
350         }
351         ret = 0;
352     }
353 }
```

(3) 智慧生活 APP 添加设备时扫描不到设备，但可以扫到模组发出的蓝牙广播

此现象通常有以下四种状况

a. 智慧生活 app 环境设置非认证沙箱环境

设置方法:我的-->设置-->关于-->环境设置



b. 平台信息没同步，刷新网页和清理智慧生活 APP 的缓存之后即可解决，这种情况通常出现在平台新建的产品上。

c. 模组内部有保存上次产品的配网信息，此次发送的蓝牙广播为不可被扫描到的广播类型，这个时候在烧录固件的时候选择 erase all，或者发送 AT+RESET 重置。

d. 蓝牙广播名称非法，设备无法被识别，需要特别注意 空格 也属于非法字符。

(4) 连接设备时注册失败，提示该设备未经过华为官方认证，该设备 MAC 非法



此现象为模组的 MAC 未录入到华为的 DP 平台，联系相关人员进行模组 MAC 录入即可解决。

(5) 设备添加注册成功，点击设备进入 H5 页面显示网络异常没有控制界面

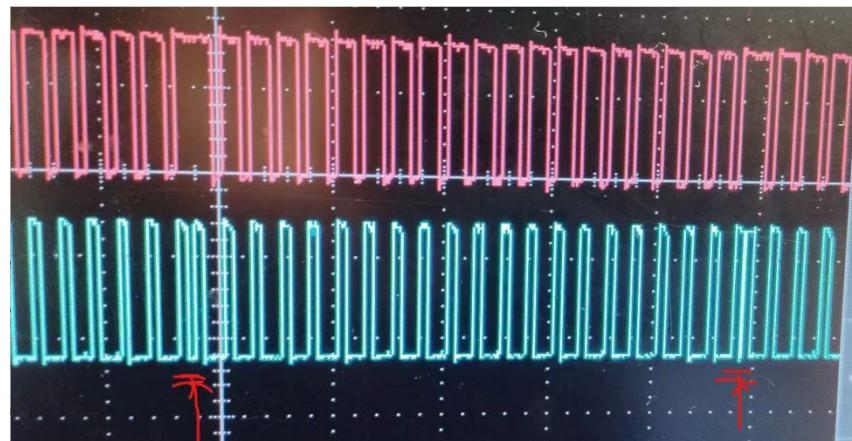


此现象常为平台数据未同步，多次刷新网页重复保存页面可以解决，清理智慧生活 APP 缓存。

2. SDK API 接口及相关功能使用问题

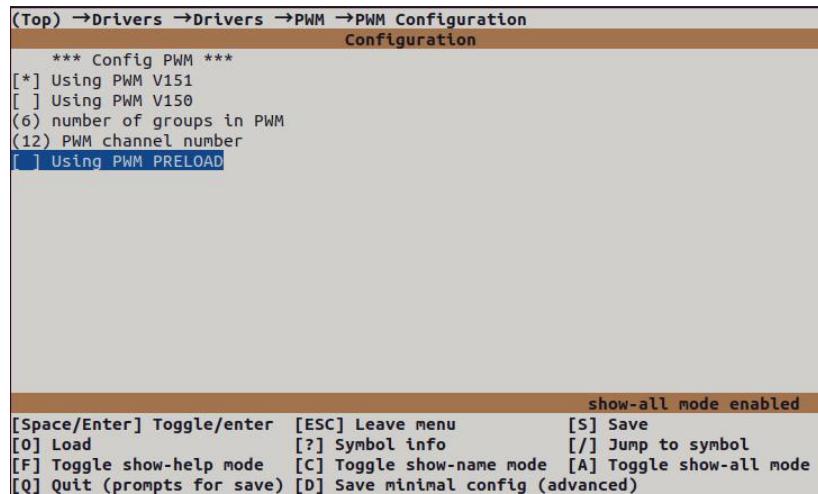
(1) PWM 相关接口的使用

Q: 当更新 PWM 占空比时, 就会破坏当前 PWM 周期, 现象是 LED 出现闪烁, 进行调光时有卡顿感。



A: 占空比动态调整可使用 PWM 预配置接口 `uapi_pwm_config_reload()`; 当上一个 PWM 周期完成时, 此配置会自动加载。

此功能默认关闭, 使用此接口需要添加该配置



(2) GPIO 模式配置及常见问题

Q: 复制例程中的 GPIO 引脚配置代码, 就改了引脚发现配置不生效。

A: 不同引脚的模式的复用信号对应功能不一定相同, 配置引脚模式请查看硬件用户指南。

Q: 如何用软件配置 GPIO 口复位功能。

A: 参考下图 demo 配置即可, 通常选择复用于 GPIO21, 只有当软件运行后配置才会生效

```
static bool pmu_control_pin_reset_enable(pin_t pin)
{
    uapi_pin_set_mode(pin, 0);
    uapi_pin_set_pull(pin, PIN_PULL_UP);
    uapi_pin_set_ie(pin, PIN_IE_1);
    reg16_setbit(0x5702C51C, 0x0);
    reg16_setbits(0x5702C51C, 0x4, 0x5, (uint8_t)pin);
    uapi_tcxo_delay_ms(5);
    reg16_clrbit(0x5702C51C, 0x0);
    return true;
}
```

(3) 重置配网接口 HILINK_BT_HardRevoke()使用注意事项

Q: 调用此接口之后模组打印下图中的日志，是什么原因？

A: 不允许在硬件中断里面调用这个接口，可在线程中去调用

(4) 操作用户区域 flash 相关 API 说明

1、hfuflash size

函数原型：

```
int HSF API hfuflash_size(void);
```

说明：

获取用户 flash 大小，单位字节

参数：

无

返回值：

返回用户 flash 大

2. hfuflasI

力原型：

int HSF API hfuf

（二）在本办法施行前，已经取得《医疗机构执业许可证》的医疗机构，应当按照本办法的规定，重新申请《医疗机构执业许可证》。

九九

删除用 `trash` 的类

参数:

addr: 用户 flash 逻辑地址

pages: 要擦除的 flash 页数

返回值:

成功返回 HF_SUCCESS, 失败返回 HF_FAIL

3、hfuflash_write

函数原型:

```
int HSF_API hfuflash_write(uint32_t addr, char *data, int len);
```

说明:

将数据写入到用户 flash 中的指定地址

参数:

addr: 用户 flash 逻辑地址

data: 指向待写入数据的指针

len: 待写入数据的长度

返回值:

如果小于零失败, 否则返回实际写入到 flash 的 Bytes 数;

4、hfuflash_read

函数原型:

```
int HSF_API hfuflash_read(uint32_t addr, char *data, int len);
```

说明:

从用户 flash 中的指定地址读取数据

参数:

addr: 用户 flash 逻辑地址

data: 指向存储读取数据的指针

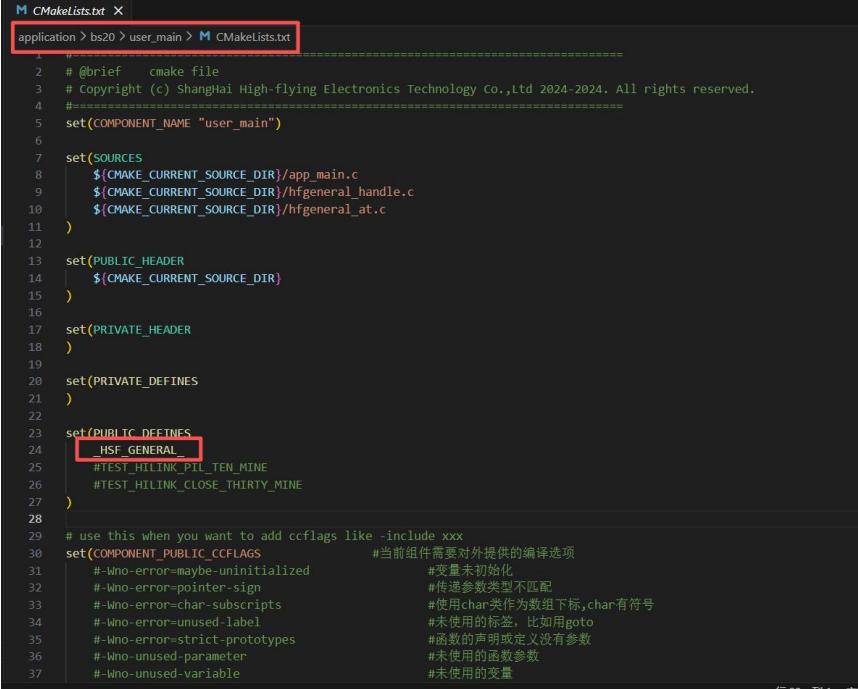
len: 待读取数据的长度

返回值:

如果小于零失败, 否则返回成功读取的 Bytes 数;

(5) SDK 开启标准串口功能

定义宏 `_HSF_GENERAL_` 即可打开，反之则关闭。

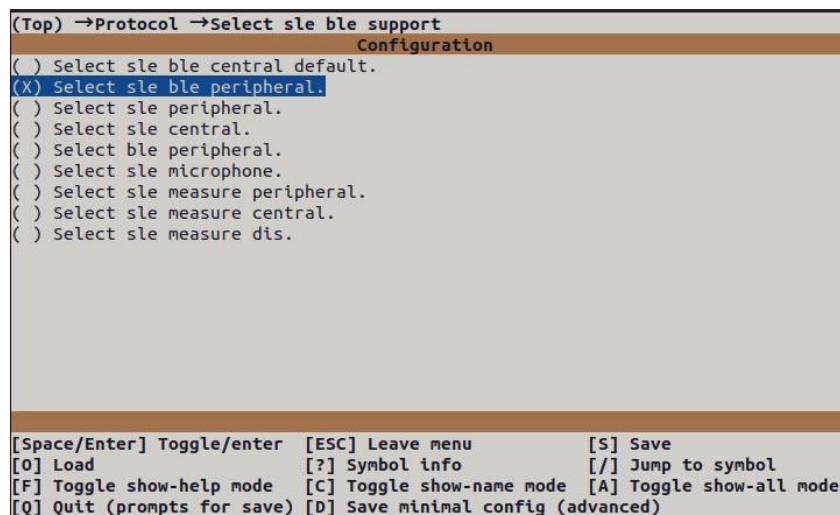


```
application > bs20 > user_main > CMakeLists.txt
1  #
2  # @brief  cmake file
3  # Copyright (c) Shanghai High-flying Electronics Technology Co.,Ltd 2024-2024. All rights reserved.
4  #=====
5  set(COMPONENT_NAME "user_main")
6
7  set(SOURCES
8      ${CMAKE_CURRENT_SOURCE_DIR}/app_main.c
9      ${CMAKE_CURRENT_SOURCE_DIR}/hfgeneral_handle.c
10     ${CMAKE_CURRENT_SOURCE_DIR}/hfgeneral_at.c
11 )
12
13 set(PUBLIC_HEADER
14     ${CMAKE_CURRENT_SOURCE_DIR}
15 )
16
17 set(PRIVATE_HEADER
18 )
19
20 set(PRIVATE_DEFINES
21 )
22
23 set(PUBLIC_DEFINES
24     _HSF_GENERAL_
25     #TEST_HILINK_PIL_TEN_MINE
26     #TEST_HILINK_CLOSE_THIRTY_MINE
27 )
28
29 # use this when you want to add ccflags like -include xxx
30 set(COMPONENT_PUBLIC_CFLAGS
31     # -Wno-error=maybe-uninitialized           #当前组件需要对外提供的编译选项
32     # -Wno-error=pointer-sign                 #变量未初始化
33     # -Wno-error=char-subscripts              #传递参数类型不匹配
34     # -Wno-error=unused-label                #使用char类作为数组下标,char有符号
35     # -Wno-error=strict-prototypes          #未使用的标签, 比如用goto
36     # -Wno-unused-parameter                 #函数的声明或定义没有参数
37     # -Wno-unused-variable                  #未使用的函数参数
38     # -Wno-unused-variable
```

3. 蓝牙、星闪相关问题

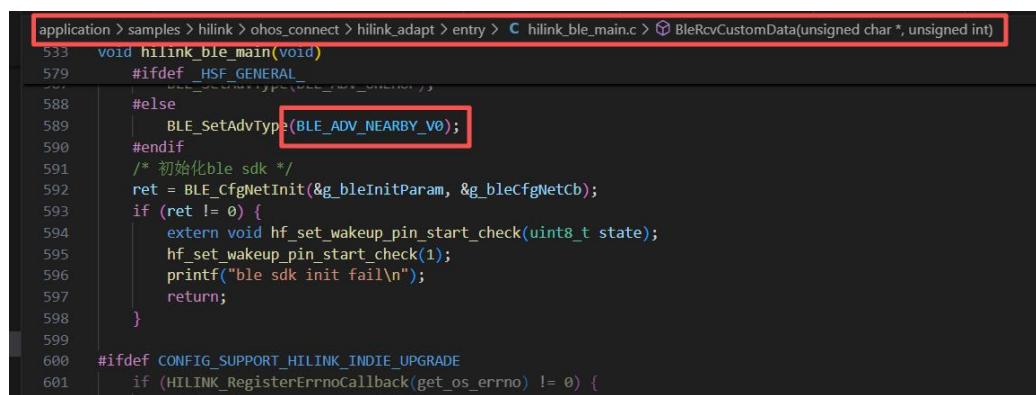
(1) 使用蓝牙注册 gatt 客户端时，接口返回 80006006 异常码

SDK 默认只开启 BLE、SLE 从机 lib>Select sle ble peripheral, 如果需要使用主机功能，需要使用 python3 build.py -ninja menuconfig 指令进入图形化配置页面开启 SLE、BLE 主从机 lib>Select sle ble central default)



(2) 启用蓝牙靠近发现是在什么地方，如何修改靠近发现的距离？

如图示路径下，通过 BLE_SetAdvType() 函数设置 hilink 广播类型，目前已经支持的广播类型有：拉取半模态卡片的靠近发现、蓝牙碰一碰以及常态广播三种。靠近发现的距离可通过修改广播的功率来修改，修改位置：文件 hilink_ble_main.c 中的 ADV_TX_POWER



```
application > samples > hilink > ohos_connect > hilink_adapt > entry > C hilink_ble_main.c > BleRcvCustomData(unsigned char *, unsigned int)
533 void hilink_ble_main(void)
534 {
535     #ifdef _HSF_GENERAL
536     /* 从机广播 */
537     BLE_SetAdvType(BLE_ADV_NEARBY_V0);
538     #else
539     /* 主机广播 */
540     BLE_SetAdvType(BLE_ADV_GENERAL);
541     #endif
542     /* 初始化ble sdk */
543     ret = BLE_CfgNetInit(&g_bleInitParam, &g_bleCfgNetCb);
544     if (ret != 0) {
545         extern void hf_set_wakeup_pin_start_check(uint8_t state);
546         hf_set_wakeup_pin_start_check(1);
547         printf("ble sdk init fail\n");
548         return;
549     }
550     #ifdef CONFIG_SUPPORT_HILINK_INDIE_UPGRADE
551         if (HILINK_RegisterErrnoCallback(get_os_errno) != 0) {
552             return;
553         }
554     #endif
555 }
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1778
1779
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1878
1879
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1888
1889
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1898
1899
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1928
1929
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1938
1939
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1948
1949
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1978
1979
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1988
1989
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1998
1999
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2078
2079
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2088
2089
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2098
2099
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2148
2149
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2178
2179
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2188
2189
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2198
2199
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2248
2249
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2278
2279
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2288
2289
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2298
2299
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2348
2349
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2378
2379
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2388
2389
2389
2390
2391
2392
2393
2394
2395
2396
2397
2397
2398
2398
2399
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2438
2439
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2448
2449
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2478
2479
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2488
2489
2489
2490
2491
2492
2493
2494
2495
2496
2497
2497
2498
2498
2499
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2548
2549
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2578
2579
2579
2580
2581
2582
2
```

```

65  /* BLE广播类型定义 */
66  typedef enum {
67      BLE_ADV_DEFAULT, //新蓝牙靠近发现
68      BLE_ADV_NEARBY_V0, //拉取半模态卡片的靠近发现
69      BLE_ADV_ONEHOP, //蓝牙碰一碰
70      BLE_ADV_LOCAL_NAME, //常态广播
71      BLE_ADV_CUSTOM
72  } BLE_AdvType;

```

(3) 不同品类设备我们的蓝牙名称有要求，这些字段应该在哪改？

蓝牙广播名称对应程序位置：

application/samples/hilink/ohos_connect/halink_adapt/entry/halink_ble_main.c



(4) 对蓝牙广播的时间有要求，在哪里进行修改？

更改 application/samples/hilink/ohos_connect/halink_adapt/entry/halink_ble_main.c 中 BLE_ADV_TIME 的值

```

application > samples > hilink > ohos_connect > hilink_adapt > entry > C hilink_ble_main.c > BLE_ADV_TIME
72  #else
73  #define FIRMWAREVER "1.0.5"
74  #define SOFTWAREVER "14.2.0.311"
75  #define HARDWAREVER "1.0.0"
76  /* 蓝牙sdk单独使用的定义 */
77  #define SUB_PRODUCT_ID          "00"
78  #define ADV_TX_POWER             0xF8
79  #define BLE_ADV_TIME             0x900B0 // 替换为0x900B0
80  /* 厂商自定义蓝牙广播，设备型号信息 */
81  #define BT_CUSTOM_INFO           "12345678"
82  #define DEVICE_MANU_ID           "01C"
83
84
85
86
87
88
89
90
91
92
93

```

(5) 使用指令 AT+CONFIG 配置完成时，APP 搜索不到设备，并且也扫描不到蓝牙广播

需要根据文档来检查参数是否有误，参数设置错误会造成蓝牙初始化失败

(6) 蓝牙使用自定义 PIN 码功能，设置的通行码未生效，只能弹出默认 PIN 码

hilink_ble_adapter.c 文件中 BleGattsSetEncryption() 回调中会关闭 PIN 码配对，需要修改对应代码来使能安全配对能力

```

application > samples > hilink > ohos_connect > hilink_adapt > adapter > C hilink_ble_adapter.c > BleGattSetEncryption(BdAddr, BleSecAct)
1087 int BleGattSetEncryption(BdAddr bdAddr, BleSecAct secAct)
1126 ,
1127 /*
1128 * @brief Set the encryption level of the data transmission channel during connection
1129 * @param[in] <bdAddr> remote address
1130 * @param[in] <secAct> BleSecAct
1131 * @return 0-success, other-fail
1132 */
1133 int BleGattSetEncryption(BdAddr bdAddr, BleSecAct secAct)
1134 {
1135     printf("%s BleGattSetEncryption enter, secAct:%d.\r\n", BLE_HILINK_SERVER_LOG, secAct);
1136
1137     gap_ble_sec_params_t sec_params = {0};
1138     sec_params.bondable = 1;
1139     sec_params.io_capability = g_io_cap_mode;
1140     sec_params.sc_enable = 0; // SC enable
1141     sec_params.sc_mode = secAct;
1142     int ret = gap_ble_set_sec_param(&sec_params);
1143     if (ret != 0) {
1144         printf("%s gap_ble_set_sec_param fail, ret:%d.\r\n", BLE_HILINK_SERVER_LOG, ret);
1145     }
1146     return 0;
1147 }
1148

```

(7) 在低功耗状态下，模组做星闪/蓝牙从机时功耗偏高，需要怎么解决

在星闪/蓝牙处于未连接状态时，需要在模组进入深睡状态时，在对应的回调函数中更新广播参数（增加广播间隔时间）来降低功耗，退出低功耗时恢复即可。处于长连接状态时，保连电流会受参数 `inteval` 和 `latency` 影响比较大，也需要在模组进入深睡状态时更新对应的连接参数

```

void hf_pm_low_power_entry(void)
{
    pm_state_trans_handler_t handler = {
        .work_to_standby = pm_state_work_to_standby,
        .standby_to_sleep = pm_state_standby_to_sleep,
        .standby_to_work = pm_state_standby_to_work,
        .sleep_to_work = pm_state_sleep_to_work,
    };
    uapi_pm_state_trans_handler_register(&handler);

    uapi_pm_work_state_reset();
    uapi_pm_set_state_trans_duration(0xFFFFFFFF, 0xFFFFFFFF);

}

```

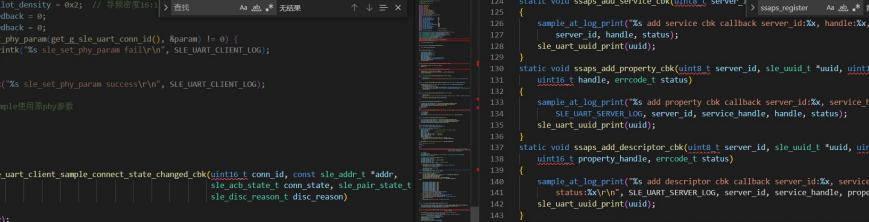
(8) 如何实时获取蓝牙的连接状态？

可在下图中标注的蓝牙连接状态回调函数中来进行获取

```
application > samples > hilink > hilink_connect > hilink_adapt > entry > C hilink_ble_main.c > ...
379     void HILINK_ReportSidState(char * sid_char,char * data_char)
420     }
421
422     static int BleCfgNetProcess(BLE_CfgNetStatus status)
423     {
424         printf("BleCfgNetProcess status[%d]\n", status);
425         #ifdef _HSF_GENERAL_
426         if(status == CFG_NET_SPEKE_SUCCESS)
427         {
428             send_wifi_state_fun(HILINK_SERVER_CONNECT,"SERVER_CONNECT");
429             int adv_type = hf_get_hilink_adv_type();
430             if ((adv_type == HF_BLE_ADV_NEARBY_NO_RETURN || adv_type == HF_BLE_ADV_LOCAL))
431             {
432                 extern void BLE_SetAdvType(int type);
433                 BLE_SetAdvType(BLE_ADV_LOCAL_NAME);
434             }
435             else if(adv_type == HF_BLE_ADV_NEARBY_V0)
436             {
437                 extern void BLE_SetAdvType(int type);
438                 BLE_SetAdvType(BLE_ADV_NEARBY_V0);
439                 hf_change_adv_timer();
440             }
441         }
442         else if(status == CFG_NET_BLE_DIS_CONNECT)
443             send_wifi_state_fun(HILINK_BLE_DISCONNECT,"BLE_DISCONNECT");
444         else if(status == CFG_NET_BLE_CONNECT)
445             send_wifi_state_fun(HILINK_BLE_CONNECT,"BLE_CONNECT");
446     #endif
447     return 0;
448     }
449 }
```

(9) 使用星闪连接时, MTU 大小有限制吗? 为什么设置时只允许 251 字节?

MTU 可以设置 251-520 字节，连接前要先通过对外接口 `ssaps_set_info` 服务端设置 MTU 自定义最大值，例如 500，客户端再根据需要设置成想要的值，最后协商结果均为客户端设置的值。



```
C:\cie_uart\client\cie_uart\cie_uart_client.c 9+ 工程设置
cie_uart_client.c @ cie_uart_client.sample.connect_state_changed_cb(uint16_t const_sle_addr_t *sle_acb_state_t, sle_pair_state_t, sle_disc_reason_t)
145     param_rx_pilot_density = 0x2; // 低功耗模式
146     param_g_feedback = 0;
147     param_t_feedback = 0;
148     if (!sle_set_phy_param_get_p_sle_uart_conn_id(), &param) { 0 }
149     osal_prinl("%s sle_set_phy_param fail\n", SLE_UART_CLIENT_LOG);
150     return;
151 }
152 osal_prinl("%s sle_set_phy_param success\n", SLE_UART_CLIENT_LOG);
153 else
154 {
155     // 单串行sample使用phy参数
156     #endif
157     return;
158 }
159 #endif
160
161 static void cie_uart_client_sample_connect_state_changed_cb(uint16_t conn_id, const_sle_addr_t *addr,
162                 const_sle_acb_state_t conn_state, const_sle_pair_state_t
163                 const_sle_disc_reason_t disc_reason)
164 {
165     unused(addr);
166     unused(pair_state);
167     osal_prinl("%s conn state changed disc_reason:0x%04x\n", SLE_UART_CLIENT_LOG, disc_reason);
168     g_sle_uart_conn_id = conn_id;
169
170     sles_connection_params_update(&con_params) = { 0 };
171     con_params.conn_id = conn_id;
172     con_params.interval_max = 48; // 每个slot 0.125ms
173     con_params.interval_min = 48; // 每个slot 0.125ms
174     con_params.latency = 0;
175     con_params.supervision_timeout = 5000; // 设置连接延时500ms
176
177     if (conn_state == SLE_ACB_STATE_CONNECTED) {
178         osal_prinl("%s SLE_ACB_STATE_CONNECTED\n", SLE_UART_CLIENT_LOG);
179         sle_update_connect_params(&con_params); // 4
180
181         ssaps_exchange_info_t info = { 0 };
182         info.nmtu_size = 500;
183         info.version = 1;
184         ssaps_exchange_info_req(0, conn_id, &info);
185
186 #ifdef CONFIG_SAMPLE_SUPPORT_LOW_LATENCY_TYPE
187     sle_low_latency_rx_enable();
188     sle_low_latency_set_get_p_sle_uart_conn_id(), true, SLE_UART_LOW_LATENCY_2K;
189 #endif
190     sle_uart_client_sample_set_phy_params();
191     osal_prinl("%s sle_set_phy_params success\n", SLE_UART_CLIENT_LOG);
192     sles_set_message_p_sle_uart_conn_id(), SLE_UART_QPSK_MCS1;
193     osal_prinl("%s sle_low_latency_rx_enable\n", SLE_UART_CLIENT_LOG);
194 }
```



```
C:\cie_uart\server\cie_uart_server.c 9+ x
cie_uart_server.c @ cie_uart_server.sample.connect_state_changed_cb(uint16_t const_sle_addr_t *sle_acb_state_t, sle_pair_state_t, sle_disc_reason_t)
124     static void ssaps_add_service_cb(uint8_t server_id) { ssaps_register_cb(ssaps_read_request_callback, ssaps_write_request_callback, server_id, handle, status); }
125     {
126         sample_at_log_print("%s add service cbk callback server_id:%x, handle:%x, status:%x\n", SLE_UART_SERVER_LOG, server_id, handle, status);
127         sle_uart_uuid.print(uuid);
128     }
129 }
130 static void ssaps_add_property_cb(uint8_t server_id, sle_uart_uuid_t *uuid, uint16_t service_handle,
131                                 uint16_t handle, encode_t status)
132 {
133     sample_at_log_print("%s add property cbk callback server_id:%x, service_handle:%x, handle:%x, status:%x\n", SLE_UART_SERVER_LOG, server_id, service_handle, handle, status);
134     sle_uart_uuid.print(uuid);
135 }
136 static void ssaps_add_descriptor_cb(uint8_t server_id, sle_uart_uuid_t *uuid, uint16_t service_handle,
137                                   uint16_t property_handle, encode_t status)
138 {
139     sample_at_log_print("%s add descriptor cbk callback server_id:%x, service_handle:%x, property_handle:%x, status:%x\n", SLE_UART_SERVER_LOG, server_id, service_handle, property_handle, status);
140     sle_uart_uuid.print(uuid);
141 }
142 }
143 static void ssaps_delete_all_service_cb(uint8_t server_id, encode_t status)
144 {
145     sample_at_log_print("%s delete all service callback server_id:%x, status:%x\n", SLE_UART_SERVER_LOG, server_id, status);
146 }
147 static encode_t ssaps_register_cbk(ssaps_read_request_callback ssaps_read_callback, ssaps_write_request_callback
148                                 ssaps_write_callback)
149 {
150     encode_t ret;
151     ssaps_callbacks_t cbk = { 0 };
152     ssaps_cbk.add_service_cb = ssaps_add_service_cb;
153     ssaps_cbk.add_property_cb = ssaps_add_property_cb;
154     ssaps_cbk.add_descriptor_cb = ssaps_add_descriptor_cb;
155     ssaps_cbk.start_service_cb = ssaps_start_service_cb;
156     ssaps_cbk.delete_all_service_cb = ssaps_delete_all_service_cb;
157
158     ssaps_exchange_info_t info = { 0 };
159     info.nmtu_size = 500;
160     info.version = 1;
161     ssaps_set_info(0, &info);
162
163     ssaps_cbk.info_changed_cb = ssaps_info_changed_cb;
164     ssaps_cbk.read_request_cb = ssaps_read_callback;
165     ssaps_cbk.write_request_cb = ssaps_write_callback;
166     ret = ssaps_register_callbacks(&ssaps_cbk);
167
168     if (ret != ERRCODE_SILE_SUCCESS) {
169         sample_at_log_print("%s ssaps_register_cbk,ssaps_register_callbacks fail:%x\n", SLE_UART_SERVER_LOG, ret);
170     }
171 }
```

(10) 使用星闪保持长连接时，进入低功耗状态，在对应回调函数中增加连接间隔的数值来降低功耗，功耗反正会增加，是什么原因？

需要满足该公式，才能正常更新连接参数：

supervision_timeout > 2 * (1+max_latency)*(interval)

(11) 蓝牙在设置发包参数时返回异常码 0x80006007，是什么原因？

需要排查连接句柄是否有误，可使用回调函数中传入的 conn_id

```
static void ble_uart_mtu_changed_cbk(uint8_t server_id, uint16_t conn_id, uint16_t mtu_size, errcode_t status)
{
    osal_printk("%s MtuChanged--server_id:%d conn_id:%d\n", BLE_UART_SERVER_LOG, server_id, conn_id);
    osal_printk("%s mtusize:%d, status:%d\n", BLE_UART_SERVER_LOG, mtu_size, status);
    gap_le_set_data_length_t param = {0};
    param.conn_handle = conn_id;
    param.maxtxoctets = MAXTXOCTETS;
    param.maxtxtime = MAXTXTIME;
    errcode_t ret = gap_ble_set_data_length(&param);
    if (ret != ERRCODE_BT_SUCCESS) {
        osal_printk("%s set data length failed ret = %d\n", BLE_UART_SERVER_ERROR, ret);
    }
    else
        osal_printk("MTU setup successful ret = %d\n", ret);
}
```

4. 环境搭建和编译常见问题

(1) 编译报错编解码解析失败

Q: 编译报错 UnicodeDecodeError: 'utf-8' codec can't decode byte...

A: 尝试如下方法:

方法 1:

终端命令行输入: `sudo gedit ~/.bashrc`

最后一行输入 `export LC_ALL=C.UTF-8`

保存退出, 重启虚拟机

方法 2:

终端命令行输入:

`sudo apt-get update`

`sudo apt-get install locales`

安装/修复完成后输入 `locale -a` 查看安装结果

重启虚拟机

5. SDK RAM 优化方案

(1) 方案简述

将系统栈 DTCM 开头移动到 ITCM 结尾，空出的 DTCM 部分作为 RAM 的补充，可增加 4KB 内存空间

(2) 修改方法

共五张修改参考图

图 1：

```
#ifdef SUPPORT_STACK_IN_ITCM
/* stack for nmi 0.5k */

#define APP_NMI_STACK_LIMIT (APP_RAMTEXT_ORIGIN + APP_ITCM_LENGTH)
#define APP_NMI_STACK_LEN 0x200
#define APP_NMI_STACK_BASEADDR (APP_NMI_STACK_LIMIT - APP_NMI_STACK_LEN)

/* stack for exception 0.5k */
#define APP_EXCP_STACK_LIMIT (APP_NMI_STACK_BASEADDR)
#define APP_EXCP_STACK_LEN 0x200
#define APP_EXCP_STACK_BASEADDR (APP_EXCP_STACK_LIMIT - APP_EXCP_STACK_LEN)

/* stack for irq 2k */
#define APP_IRQ_STACK_LIMIT (APP_EXCP_STACK_BASEADDR)
#define APP_IRQ_STACK_LEN 0x800
#define APP_IRQ_STACK_BASEADDR (APP_IRQ_STACK_LIMIT - APP_IRQ_STACK_LEN)

/* stack for normal 1.5k */
#define APP_USER_STACK_LIMIT (APP_IRQ_STACK_BASEADDR)
#define APP_USER_STACK_LEN 0x600
#define APP_USER_STACK_BASEADDR (APP_USER_STACK_LIMIT - APP_USER_STACK_LEN)

#define APP_STACK_TOTAL_LEN (APP_NMI_STACK_LEN + APP_EXCP_STACK_LEN + APP_IRQ_STACK_LEN +
APP_USER_STACK_LEN)
#else
```

```

drivers > chips > bs20 > board > memory_config > include > C memory_config_common.h > {} #ifndef MEMORY_CONFIG_COMMON_H
  156 #define APP_RAHTEXT_LENGTH (APP_ITCM_LENGTH)
  157
  158 /* 64K DTCM for APP core data */
  159
  160 #ifndef MEMORY_CONFIG_COMMON_H
  161 #define APP_NMI_STACK_IN_ITCM
  162 #define APP_NMI_STACK_LINIT (APP_RAHTEXT_ORIGIN + APP_ITCM_LENGTH)
  163 #define APP_NMI_STACK_BASEADDR (APP_NMI_STACK_LINIT - APP_NMI_STACK_LEN)
  164
  165 /* stack for exception 0.5k */
  166 #define APP_EXCP_STACK_LINIT (APP_NMI_STACK_BASEADDR)
  167 #define APP_EXCP_STACK_LEN 0x200
  168 #define APP_EXCP_STACK_BASEADDR (APP_EXCP_STACK_LINIT - APP_EXCP_STACK_LEN)
  169
  170 /* stack for normal 1.5k */
  171 #define APP_USER_STACK_BASEADDR APP_DTCM_ORIGIN
  172 #define APP_USER_STACK_LEN 0x800
  173 #define APP_USER_STACK_LINIT (APP_USER_STACK_BASEADDR + APP_USER_STACK_LEN)
  174
  175 /* stack for normal 1.5k */
  176 #define APP_IRO_STACK_BASEADDR APP_USER_STACK_LINIT
  177 #define APP_IRO_STACK_LEN 0x800
  178 #define APP_IRO_STACK_LINIT (APP_IRO_STACK_BASEADDR + APP_IRO_STACK_LEN)
  179
  180 /* stack for exception 0.5k */
  181 #define APP_EXCP_STACK_BASEADDR APP_DTCM_ORIGIN
  182 #define APP_EXCP_STACK_LEN 0x800
  183 #define APP_EXCP_STACK_LINIT (APP_EXCP_STACK_BASEADDR + APP_USER_STACK_LEN)
  184
  185 /* stack for normal 1.5k */
  186 #define APP_IRO_STACK_BASEADDR APP_USER_STACK_LINIT
  187 #define APP_IRO_STACK_LEN 0x800
  188 #define APP_IRO_STACK_LINIT (APP_IRO_STACK_BASEADDR + APP_IRO_STACK_LEN)
  189
  190 /* stack for normal 1.5k */
  191 #define APP_EXCP_STACK_BASEADDR APP_IRO_STACK_LINIT
  192 #define APP_EXCP_STACK_LEN 0x200
  193 #define APP_EXCP_STACK_LINIT (APP_EXCP_STACK_BASEADDR + APP_EXCP_STACK_LEN)
  194
  195 /* stack for exception 0.5k */
  196 #define APP_EXCP_STACK_BASEADDR APP_IRO_STACK_LINIT
  197 #define APP_EXCP_STACK_LEN 0x200
  198 #define APP_EXCP_STACK_LINIT (APP_EXCP_STACK_BASEADDR + APP_EXCP_STACK_LEN)
  199
  200 /* stack for normal 1.5k */
  201 #define APP_NMI_STACK_BASEADDR APP_IRO_STACK_LINIT
  202 #define APP_NMI_STACK_LEN 0x200
  203 #define APP_NMI_STACK_LINIT (APP_NMI_STACK_BASEADDR + APP_NMI_STACK_LEN)
  204
  205 /* 12*N bytes for cpu trace, trace line is 12 bytes, 1020 bytes */
  206
  207 /* 12*N bytes for cpu trace, trace line is 12 bytes, 1020 bytes */
  208 /* 12*N bytes for cpu trace, trace line is 12 bytes, 1020 bytes */
  209 /* 12*N bytes for cpu trace, trace line is 12 bytes, 1020 bytes */
  210 /* 12*N bytes for cpu trace, trace line is 12 bytes, 1020 bytes */
  211 /* 12*N bytes for cpu trace, trace line is 12 bytes, 1020 bytes */
  212 /* 12*N bytes for cpu trace, trace line is 12 bytes, 1020 bytes */
  213 /* 12*N bytes for cpu trace, trace line is 12 bytes, 1020 bytes */
  214 /* 12*N bytes for cpu trace, trace line is 12 bytes, 1020 bytes */
  215 /* 12*N bytes for cpu trace, trace line is 12 bytes, 1020 bytes */
  216 /* 12*N bytes for cpu trace, trace line is 12 bytes, 1020 bytes */
  217 /* 12*N bytes for cpu trace, trace line is 12 bytes, 1020 bytes */
  218 /* 12*N bytes for cpu trace, trace line is 12 bytes, 1020 bytes */
  219
  220 /* 12*N bytes for cpu trace, trace line is 12 bytes, 1020 bytes */
  221
  222 /* 12*N bytes for cpu trace, trace line is 12 bytes, 1020 bytes */
  223
  224 /* 12*N bytes for cpu trace, trace line is 12 bytes, 1020 bytes */
  225
  226 /* 12*N bytes for cpu trace, trace line is 12 bytes, 1020 bytes */
  227
  228 /* 12*N bytes for cpu trace, trace line is 12 bytes, 1020 bytes */
  229
  230 /* 12*N bytes for cpu trace, trace line is 12 bytes, 1020 bytes */
  231
  232 /* 12*N bytes for cpu trace, trace line is 12 bytes, 1020 bytes */
  233
  234 /* 12*N bytes for cpu trace, trace line is 12 bytes, 1020 bytes */
  235
  236 /* 12*N bytes for cpu trace, trace line is 12 bytes, 1020 bytes */
  237
  238 /* 12*N bytes for cpu trace, trace line is 12 bytes, 1020 bytes */
  239
  240 /* 12*N bytes for cpu trace, trace line is 12 bytes, 1020 bytes */
  241
  242 /* 12*N bytes for cpu trace, trace line is 12 bytes, 1020 bytes */
  243
  244 /* 12*N bytes for cpu trace, trace line is 12 bytes, 1020 bytes */
  245
  246 /* 12*N bytes for cpu trace, trace line is 12 bytes, 1020 bytes */
  247
  248 /* 12*N bytes for cpu trace, trace line is 12 bytes, 1020 bytes */
  249
  250 /* 12*N bytes for cpu trace, trace line is 12 bytes, 1020 bytes */
  251
  252 /* 12*N bytes for cpu trace, trace line is 12 bytes, 1020 bytes */
  253
  254
  255 /* 12*N bytes for cpu trace, trace line is 12 bytes, 1020 bytes */
  256
  257
  258
  259
  260
  261
  262
  263
  264
  265
  266
  267
  268
  269
  270
  271
  272
  273
  274
  275
  276
  277
  278
  279
  280
  281
  282
  283
  284
  285
  286
  287
  288
  289
  290
  291
  292
  293
  294
  295
  296
  297
  298
  299
  300
  301
  302
  303
  304
  305
  306
  307
  308
  309
  310
  311
  312
  313
  314
  315
  316
  317
  318
  319
  320
  321
  322
  323
  324
  325
  326
  327
  328
  329
  330
  331
  332
  333
  334
  335
  336
  337
  338
  339
  340
  341
  342
  343
  344
  345
  346
  347
  348
  349
  350
  351
  352
  353
  354
  355
  356
  357
  358
  359
  360
  361
  362
  363
  364
  365
  366
  367
  368
  369
  370
  371
  372
  373
  374
  375
  376
  377
  378
  379
  380
  381
  382
  383
  384
  385
  386
  387
  388
  389
  390
  391
  392
  393
  394
  395
  396
  397
  398
  399
  400
  401
  402
  403
  404
  405
  406
  407
  408
  409
  410
  411
  412
  413
  414
  415
  416
  417
  418
  419
  420
  421
  422
  423
  424
  425
  426
  427
  428
  429
  430
  431
  432
  433
  434
  435
  436
  437
  438
  439
  440
  441
  442
  443
  444
  445
  446
  447
  448
  449
  450
  451
  452
  453
  454
  455
  456
  457
  458
  459
  460
  461
  462
  463
  464
  465
  466
  467
  468
  469
  470
  471
  472
  473
  474
  475
  476
  477
  478
  479
  480
  481
  482
  483
  484
  485
  486
  487
  488
  489
  490
  491
  492
  493
  494
  495
  496
  497
  498
  499
  500
  501
  502
  503
  504
  505
  506
  507
  508
  509
  510
  511
  512
  513
  514
  515
  516
  517
  518
  519
  520
  521
  522
  523
  524
  525
  526
  527
  528
  529
  530
  531
  532
  533
  534
  535
  536
  537
  538
  539
  540
  541
  542
  543
  544
  545
  546
  547
  548
  549
  550
  551
  552
  553
  554
  555
  556
  557
  558
  559
  560
  561
  562
  563
  564
  565
  566
  567
  568
  569
  570
  571
  572
  573
  574
  575
  576
  577
  578
  579
  580
  581
  582
  583
  584
  585
  586
  587
  588
  589
  590
  591
  592
  593
  594
  595
  596
  597
  598
  599
  600
  601
  602
  603
  604
  605
  606
  607
  608
  609
  610
  611
  612
  613
  614
  615
  616
  617
  618
  619
  620
  621
  622
  623
  624
  625
  626
  627
  628
  629
  630
  631
  632
  633
  634
  635
  636
  637
  638
  639
  640
  641
  642
  643
  644
  645
  646
  647
  648
  649
  650
  651
  652
  653
  654
  655
  656
  657
  658
  659
  660
  661
  662
  663
  664
  665
  666
  667
  668
  669
  670
  671
  672
  673
  674
  675
  676
  677
  678
  679
  680
  681
  682
  683
  684
  685
  686
  687
  688
  689
  690
  691
  692
  693
  694
  695
  696
  697
  698
  699
  700
  701
  702
  703
  704
  705
  706
  707
  708
  709
  710
  711
  712
  713
  714
  715
  716
  717
  718
  719
  720
  721
  722
  723
  724
  725
  726
  727
  728
  729
  730
  731
  732
  733
  734
  735
  736
  737
  738
  739
  740
  741
  742
  743
  744
  745
  746
  747
  748
  749
  750
  751
  752
  753
  754
  755
  756
  757
  758
  759
  760
  761
  762
  763
  764
  765
  766
  767
  768
  769
  770
  771
  772
  773
  774
  775
  776
  777
  778
  779
  780
  781
  782
  783
  784
  785
  786
  787
  788
  789
  790
  791
  792
  793
  794
  795
  796
  797
  798
  799
  800
  801
  802
  803
  804
  805
  806
  807
  808
  809
  810
  811
  812
  813
  814
  815
  816
  817
  818
  819
  820
  821
  822
  823
  824
  825
  826
  827
  828
  829
  830
  831
  832
  833
  834
  835
  836
  837
  838
  839
  840
  841
  842
  843
  844
  845
  846
  847
  848
  849
  850
  851
  852
  853
  854
  855
  856
  857
  858
  859
  860
  861
  862
  863
  864
  865
  866
  867
  868
  869
  870
  871
  872
  873
  874
  875
  876
  877
  878
  879
  880
  881
  882
  883
  884
  885
  886
  887
  888
  889
  890
  891
  892
  893
  894
  895
  896
  897
  898
  899
  900
  901
  902
  903
  904
  905
  906
  907
  908
  909
  910
  911
  912
  913
  914
  915
  916
  917
  918
  919
  920
  921
  922
  923
  924
  925
  926
  927
  928
  929
  930
  931
  932
  933
  934
  935
  936
  937
  938
  939
  940
  941
  942
  943
  944
  945
  946
  947
  948
  949
  950
  951
  952
  953
  954
  955
  956
  957
  958
  959
  960
  961
  962
  963
  964
  965
  966
  967
  968
  969
  970
  971
  972
  973
  974
  975
  976
  977
  978
  979
  980
  981
  982
  983
  984
  985
  986
  987
  988
  989
  990
  991
  992
  993
  994
  995
  996
  997
  998
  999
  1000
  1001
  1002
  1003
  1004
  1005
  1006
  1007
  1008
  1009
  1010
  1011
  1012
  1013
  1014
  1015
  1016
  1017
  1018
  1019
  1020
  1021
  1022
  1023
  1024
  1025
  1026
  1027
  1028
  1029
  1030
  1031
  1032
  1033
  1034
  1035
  1036
  1037
  1038
  1039
  1040
  1041
  1042
  1043
  1044
  1045
  1046
  1047
  1048
  1049
  1050
  1051
  1052
  1053
  1054
  1055
  1056
  1057
  1058
  1059
  1060
  1061
  1062
  1063
  1064
  1065
  1066
  1067
  1068
  1069
  1070
  1071
  1072
  1073
  1074
  1075
  1076
  1077
  1078
  1079
  1080
  1081
  1082
  1083
  1084
  1085
  1086
  1087
  1088
  1089
  1090
  1091
  1092
  1093
  1094
  1095
  1096
  1097
  1098
  1099
  1100
  1101
  1102
  1103
  1104
  1105
  1106
  1107
  1108
  1109
  1110
  1111
  1112
  1113
  1114
  1115
  1116
  1117
  1118
  1119
  1120
  1121
  1122
  1123
  1124
  1125
  1126
  1127
  1128
  1129
  1130
  1131
  1132
  1133
  1134
  1135
  1136
  1137
  1138
  1139
  1140
  1141
  1142
  1143
  1144
  1145
  1146
  1147
  1148
  1149
  1150
  1151
  1152
  1153
  1154
  1155
  1156
  1157
  1158
  1159
  1160
  1161
  1162
  1163
  1164
  1165
  1166
  1167
  1168
  1169
  1170
  1171
  1172
  1173
  1174
  1175
  1176
  1177
  1178
  1179
  1180
  1181
  1182
  1183
  1184
  1185
  1186
  1187
  1188
  1189
  1190
  1191
  1192
  1193
  1194
  1195
  1196
  1197
  1198
  1199
  1200
  1201
  1202
  1203
  1204
  1205
  1206
  1207
  1208
  1209
  1210
  1211
  1212
  1213
  1214
  1215
  1216
  1217
  1218
  1219
  1220
  1221
  1222
  1223
  1224
  1225
  1226
  1227
  1228
  1229
  1230
  1231
  1232
  1233
  1234
  1235
  1236
  1237
  1238
  1239
  1240
  1241
  1242
  1243
  1244
  1245
  1246
  1247
  1248
  1249
  1250
  1251
  1252
  1253
  1254
  1255
  1256
  1257
  1258
  1259
  1260
  1261
  1262
  1263
  1264
  1265
  1266
  1267
  1268
  1269
  1270
  1271
  1272
  1273
  1274
  1275
  1276
  1277
  1278
  1279
  1280
  1281
  1282
  1283
  1284
  1285
  1286
  1287
  1288
  1289
  1290
  1291
  1292
  1293
  1294
  1295
  1296
  1297
  1298
  1299
  1300
  1301
  1302
  1303
  1304
  1305
  1306
  1307
  1308
  1309
  1310
  1311
  1312
  1313
  1314
  1315
  1316
  1317
  1318
  1319
  1320
  1321
  1322
  1323
  1324
  1325
  1326
  1327
  1328
  1329
  1330
  1331
  1332
  1333
  1334
  1335
  1336
  1337
  1338
  1339
  1340
  1341
  1342
  1343
  1344
  1345
  1346
  1347
  1348
  1349
  1350
  1351
  1352
  1353
  1354
  1355
  1356
  1357
  1358
  1359
  1360
  1361
  1362
  1363
  1364
  1365
  1366
  1367
  1368
  1369
  1370
  1371
  1372
  1373
  1374
  1375
  1376
  1377
  1378
  1379
  1380
  1381
  1382
  1383
  1384
  1385
  1386
  1387
  1388
  1389
  1390
  1391
  1392
  1393
  1394
  1395
  1396
  1397
  1398
  1399
  1400
  1401
  1402
  1403
  1404
  1405
  1406
  1407
  1408
  1409
  1410
  1411
  1412
  1413
  1414
  1415
  1416
  1417
  1418
  1419
  1420
  1421
  1422
  1423
  1424
  1425
  1426
  1427
  1428
  1429
  1430
  1431
  1432
  1433
  1434
  1435
  1436
  1437
  1438
  1439
  1440
  1441
  1442
  1443
  1444
  1445
  1446
  1447
  1448
  1449
  1450
  1451
  1452
  1453
  1454
  1455
  1456
  1457
  1458
  1459
  1460
  1461
  1462
  1463
  1464
  1465
  1466
  1467
  1468
  1469
  1470
  1471
  1472
  1473
  1474
  1475
  1476
  1477
  1478
  1479
  1480
  1481
  1482
  1483
  1484
  1485
  1486
  1487
  1488
  1489
  1490
  1491
  1492
  1493
  1494
  1495
  1496
  1497
  1498
  1499
  1500
  1501
  1502
  1503
  1504
  1505
  1506
  1507
  1508
  1509
  1510
  1511
  1512
  1513
  1514
  1515
  1516
  1517
  1518
  1519
  1520
  1521
  1522
  1523
  1524
  1525
  1526
  1527
  1528
  1529
  1530
  1531
  1532
  1533
  1534
  1535
  1536
  1537
  1538
  1539
  1540
  1541
  1542
  1543
  1544
  1545
  1546
  1547
  1548
  1549
  1550
  1551
  1552
  1553
  1554
  1555
  1556
  1557
  1558
  1559
  1560
  1561
  1562
  1563
  1564
  1565
  1566
  1567
  1568
  1569
  1570
  1571
  1572
  1573
  1574
  1575
  1576
  1577
  1578
  1579
  1580
  1581
  1582
  1583
  1584
  1585
  1586
  1587
  1588
  1589
  1590
  1591
  1592
  1593
  1594
  1595
  1596
  1597
  1598
  1599
  1600
  1601
  1602
  1603
  1604
  1605
  1606
  1607
  1608
  1609
  1610
  1611
  1612
  1613
  1614
  1615
  1616
  1617
  1618
  1619
  1620
  1621
  1622
  1623
  1624
  1625
  1626
  1627
  1628
  1629
  1630
  1631
  1632
  1633
  1634
  1635
  1636
  1637
  1638
  1639
  1640
  1641
  1642
  1643
  1644
  1645
  1646
  1647
  1648
  1649
  1650
  1651
  1652
  1653
  1654
  1655
  1656
  1657
  1658
  1659
  1660
  1661
  1662
  1663
  1664
  1665
  1666
  1667
  1668
  1669
  1670
  1671
  1672
  1673
  1674
  1675
  1676
  1677
  1678
  1679
  1680
  1681
  16
```

图 4:

```
drivers > chips > bs20 > board > linker > standard > linker.prelds
78     {
98     {
105     . = ALIGN(16);
106     __irq_stack_top__ = .;
107     . += APP_EXCP_STACK_LEN;
108     . = ALIGN(16);
109     __exc_stack_top__ = .;
110     __exc_stack_top__ = .;
111     . += APP_NMI_STACK_LEN;
112     . = ALIGN(16);
113     __nmi_stack_top__ = .;
114     __nmi_stack_top__ = .;
115     g_system_stack_end = .;
116 #ifdef SUPPORT_STACK_IN_ITCM
117     } > ITCM_STACK
118 #else
119 // #ifdef CONFIG_SUPPORT_LOG_THREAD
120 //     . += LOGGING_REGION_LENGTH;
121 //     . = ALIGN(16);
122 // #endif
123     } > DTCM
124 #endif
125     g_system_stack_size = g_system_stack_end - g_system_stack_begin;
126     g_stack_system = g_system_stack_size;
127 }
```

图 5:

```
build > config > target_config > bs20 > config.py
1  #!/usr/bin/env python3
2  # encoding=utf-8
3  # =====
4  # @brief  Target Definitions File
5  # Copyright CompanyNameMagicTag 2022-2022. All rights reserved.
6  # =====
7
8  target = {
9      'standard-bs20-n1200': {
10         'base_target_name': 'target_bs20_application_template',
11         'pkg_chip': 'bs20-n1200',
12         'defines': [
13             'FLASH_1M', 'USE_RSA3072_SIGN', 'BGLE_TASK_EXIST', 'SUPPORT_MULTI_LIBS',
14             'SW_UART_DEBUG', 'XO_32M_CALI', 'SUPPORT_CHIP_N1200', 'SUPPORT_STACK_IN_ITCM',
15             '#, 'SUPPORT_CFBB_UPG' 'AT_COMMAND', 'OS_DFX_SUPPORT',
16             ],
17     }
```

(3) 修改验证

参考下图：多出新的区域 ITCM_STACK 且 DTCM 的使用为空

```
Generating /opt/liangjunwei/workspace/BSL600/drivers/chips/bs2i
Memory region      Used Size  Region Size %age Used
  ROM:          0 GB    169 KB    0.00%
  ITCM:        73656 B  74240 B  99.21%
  ITCM STACK:  4608 B   4608 B 100.00%
  DTCM:        0 GB   65280 B    0.00%
  PRESERVE:    252 B    256 B   98.44%
  FLASH_STARTUP: 488 B    768 B   63.54%
  FLASH_PROGRAM: 525186 B 611584 B  85.87%
  CPUTRACE_RAM: 1020 B   1020 B 100.00%
  SYMBOL_UNREACH: 220 B    512 B   42.97%
```