0015-MSM RF Driver Configuration

注：本文参考项目路径和代码为SIM7600 LE20分支

1 原理

MSM/MDM+WTR RF Frontend(MIPI)结构

2 MIPI ASM Customization

Reference

80-NG377-1_A_MIPI_Device_Customization.pdf

添加或修改天线开关设备

2.1 Step1 ASM设备驱动

文件路径：`AMSS_LE20/modem_proc/rfdevice_asm/src`

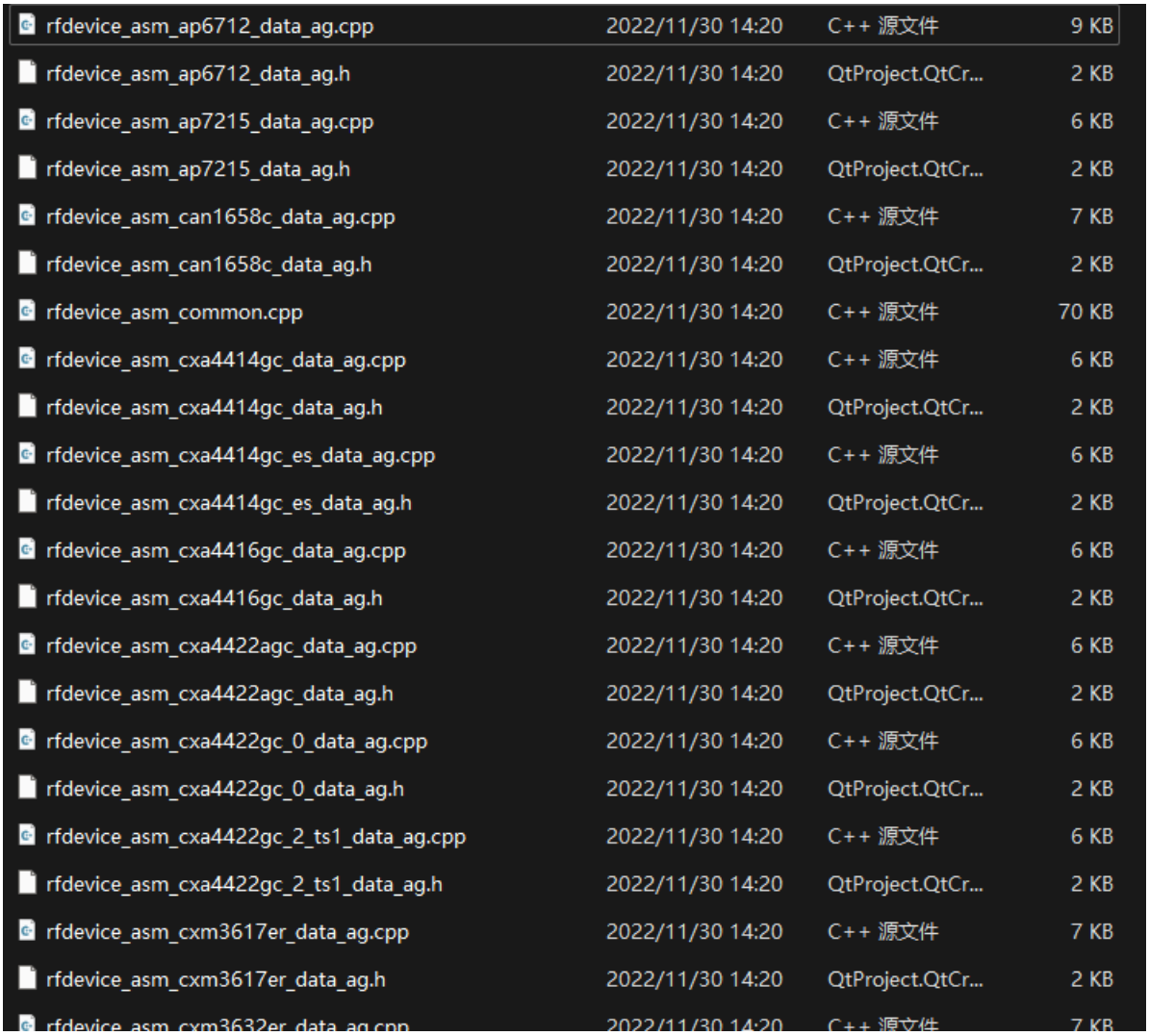
可以完成如下工作：

1. 为已存在的ASM设备更改配置

比如在`rfdevice_asm_cxa4416gc_data_ag.h`和`rfdevice_asm_cxa4416gc_data_ag.cpp`中为cxa4416gc修改配置。

2. 添加一个新的ASM设备

为一个新ASM设备添加.h和.cpp文件，.h和.cpp文件内容可以参考已经存在的其他设备的文件内容。



- 在.cpp文件中为ASM on/off/trigger操作定义寄存器配置

```
#define RFDEVICE_ASM_S5643_52_NUM_PORTS 6

#define RFDEVICE_ASM_S5643_52_ASM_ON_NUM_REGS 1
static uint8
rfdevice_asm_s5643_52_asm_on_regs[RFDEVICE_ASM_S5643_52_ASM_ON_NUM_REGS] =
{0x02, };
static int16 rfdevice_asm_s5643_52_asm_on_data[RFDEVICE_ASM_S5643_52_NUM_PORTS]
[RFDEVICE_ASM_S5643_52_ASM_ON_NUM_REGS] =
{
  { /* PORT NUM: 0 *//* HB1->HBRX2*/
    0x01,
  },
  { /* PORT NUM: 1 *//* HB2->HBRX2*/
    0x02,
  },
  { /* PORT NUM: 2 *//* HB3->HBRX1*/
    0x03,
  },
  { /* PORT NUM: 3 *//* HB4->HBRX2*/
    0x04,
  },
  { /* PORT NUM: 4 *//* Switch off*/
    0x00,
  },
```

```
  { /* PORT NUM: 5 *//* High Isolation*/
    0x00,
  },
};


#define RFDEVICE_ASM_S5643_52_ASM_OFF_NUM_REGS 1
static uint8
rfdevice_asm_s5643_52_asm_off_regs[RFDEVICE_ASM_S5643_52_ASM_OFF_NUM_REGS] =
{0x02, };
static int16 rfdevice_asm_s5643_52_asm_off_data[RFDEVICE_ASM_S5643_52_NUM_PORTS]
[RFDEVICE_ASM_S5643_52_ASM_OFF_NUM_REGS] =
{
  { /* PORT NUM: 0 */
    0x00,
  },
  { /* PORT NUM: 1 */
    0x00,
  },
  { /* PORT NUM: 2 */
    0x00,
  },
  { /* PORT NUM: 3 */
    0x00,
  },
  { /* PORT NUM: 4 */
    0x00,
  },
  { /* PORT NUM: 5 */
    0x00,
  },


};


#define RFDEVICE_ASM_S5643_52_ASM_TRIGGER_NUM_REGS 1
static uint8
rfdevice_asm_s5643_52_asm_trigger_regs[RFDEVICE_ASM_S5643_52_ASM_TRIGGER_NUM_REG
S] =  {0x1C, };
static int16
rfdevice_asm_s5643_52_asm_trigger_data[RFDEVICE_ASM_S5643_52_NUM_PORTS]
[RFDEVICE_ASM_S5643_52_ASM_TRIGGER_NUM_REGS] =
{
  { /* PORT NUM: 0 */
    0x07,
  },
  { /* PORT NUM: 1 */
    0x07,
  },
  { /* PORT NUM: 2 */
    0x07,
  },
  { /* PORT NUM: 3 */
    0x07,
```

```
  },
  { /* PORT NUM: 4 */
    0x07,
  },
  { /* PORT NUM: 5 */
    0x07,
  },
};
```

注意：

RFDEVICE_ASM_S5643_52_NUM_PORTS是端口的数量，不同的端口对应不同的频段开关。该数量与rfdevice_asm_s5643_52_asm_on_data列表中的寄存器数值是一致的。比如该值设置为6，那么与rfdevice_asm_s5643_52_asm_on_data肯定应该有6个数值。

表格1 S5643_52真值表

将这些值转换为16进制后，与代码modem_proc/rfdevice_asm/src/rfdevice_asm_s5643_52_data_ag.cpp中的rfdevice_asm_s5643_52_asm_on_data[]对应起来。

表格2 端口与真值对应关系表

这样在代码里面，就可以为gsm、wcdma、lte...，来选择ASM设备端口了。

表格3 ASM设备GSM配置表

- 在.cpp文件中为ASM设备配置正确的MID、PID和product revision

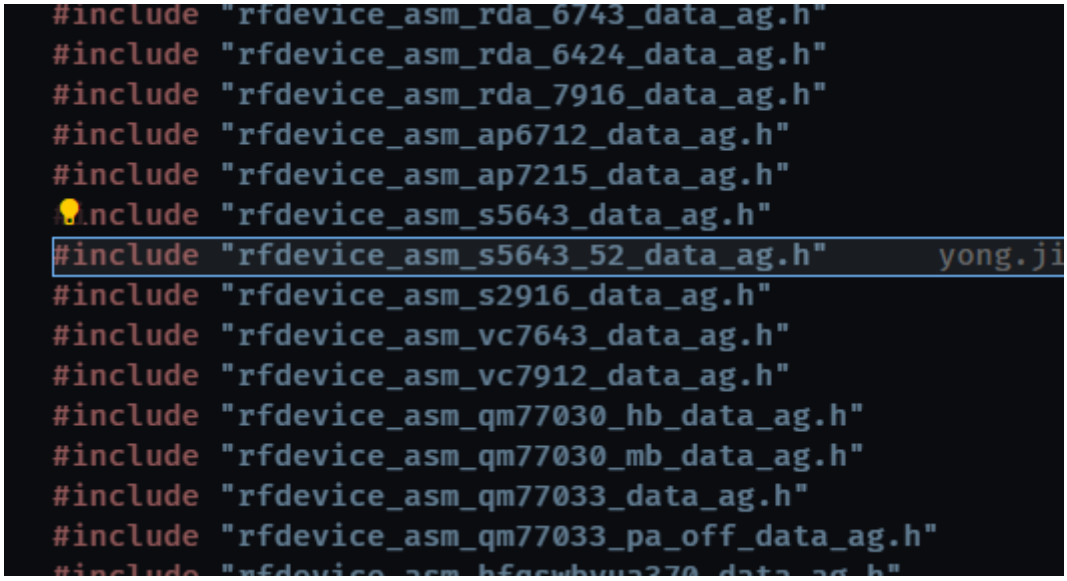
```
boolean rfdevice_asm_s5643_52_data_ag::device_info_get( rfdevice_asm_info_type
*asm_info )
{
  boolean ret_val = FALSE;

  if ( NULL == asm_info )
  {
    return FALSE;
  }
  else
  {
      asm_info->mfg_id = 0x02E9;
      asm_info->prd_id = 0x8A;
      asm_info->prd_rev = 0;
      asm_info->num_ports = RFDEVICE_ASM_S5643_52_NUM_PORTS;
      ret_val = TRUE;
  }
  return ret_val;
}
```

MID即MANUFACTURER ID，PID即PRODUCT ID，由芯片spec查到。

2.2 Step2 更新FTM中的ASM信息

在modem_proc/rfdevice_asm/src/rfdevice_asm_factory_ag.cpp中：

1. 添加新ASM设备的.h头文件
2. 为新添加的ASM设备更改或添加程序




```
rfdevice_asm_data* rfdevice_asm_data_create (uint16 mfg_id, uint8 prd_id, uint8
prd_rev)
{
  rfdevice_asm_data * asm_data = NULL;

  if ( mfg_id ==  0x01B0 && prd_id == 0x35  && prd_rev == 2)
  {
    asm_data = rfdevice_asm_cxa4422agc_data_ag::get_instance();
  }
.......
  else if ( mfg_id ==  0x02E9 && prd_id == 0x89  && prd_rev == 0)
  {
    asm_data = rfdevice_asm_s5643_data_ag::get_instance();
  }
  else if ( mfg_id ==  0x02E9 && prd_id == 0x8A  && prd_rev == 0)
  {
    asm_data = rfdevice_asm_s5643_52_data_ag::get_instance();
  }
......
  }
```

2.3 Step3 更新common devices list

在RFC common文件中，为ASM设备更新信息，比如：在modem_proc/rfc_jolokia/rf_card/rfc_wtr2965_non_ca2_4320_sim_0/common/src/rfc_wtr2965_non_ca2_4320_sim_0_cmn_ag.cpp中：

```
rfc_phy_device_info_type rfc_wtr2965_non_ca2_4320_sim_0_phy_devices_list[] =
```

```
{
......
  { /*Device: S5643 */
    GEN_DEVICE, /* PHY_DEVICE_NAME */
    1, /* PHY_DEVICE_INSTANCE */
    1, /* PHY_DEVICE_ALT_PART_NUM_OF_INSTANCE */
    RFDEVICE_COMM_PROTO_RFFE, /* PHY_DEVICE_COMM_PROTOCOL */
    RFDEVICE_COMM_PROTO_VERSION_DEFAULT, /* PHY_DEVICE_COMM_PROTOCOL_VERSION */
    {    0,0 /* 0 not specified */,}, /* PHY_DEVICE_COMM_BUS */
    0x02E9, /* PHY_DEVICE_MANUFACTURER_ID */
    0x89, /* PHY_DEVICE_PRODUCT_ID */
#ifdef FEATURE_NO_PA_DEBUG
    0 | RFC_SKIP_RFFE_DETECT_BIT_IND, /* PHY_DEVICE_PRODUCT_REV */
#else
    0, /* PHY_DEVICE_PRODUCT_REV */
#endif
    0x0F, /* DEFAULT USID RANGE START */
    0x0F, /* DEFAULT USID RANGE END */
    0x0F, /* PHY_DEVICE_ASSIGNED_USID */
    0 /*Warning: Not specified*/, /* RFFE_GROUP_ID */
    FALSE, /* INIT */
    RFC_INVALID_PARAM, /* ASSOCIATED_DAC */
  }, /* END - Device: S5643 */
  { /*Device: S5643-52 */
    GEN_DEVICE, /* PHY_DEVICE_NAME */
    1, /* PHY_DEVICE_INSTANCE */
    2, /* PHY_DEVICE_ALT_PART_NUM_OF_INSTANCE */
    RFDEVICE_COMM_PROTO_RFFE, /* PHY_DEVICE_COMM_PROTOCOL */
    RFDEVICE_COMM_PROTO_VERSION_DEFAULT, /* PHY_DEVICE_COMM_PROTOCOL_VERSION */
    {    0,0 /* 0 not specified */,}, /* PHY_DEVICE_COMM_BUS */
    0x02E9, /* PHY_DEVICE_MANUFACTURER_ID */
    0x8A, /* PHY_DEVICE_PRODUCT_ID */
#ifdef FEATURE_NO_PA_DEBUG
    0 | RFC_SKIP_RFFE_DETECT_BIT_IND, /* PHY_DEVICE_PRODUCT_REV */
#else
    0, /* PHY_DEVICE_PRODUCT_REV */
#endif
    0x0F, /* DEFAULT USID RANGE START */
    0x0F, /* DEFAULT USID RANGE END */
    0x0F, /* PHY_DEVICE_ASSIGNED_USID */
    0 /*Warning: Not specified*/, /* RFFE_GROUP_ID */
    FALSE, /* INIT */
    RFC_INVALID_PARAM, /* ASSOCIATED_DAC */
  }, /* END - Device: S5643 */
 ......
};
```

2.4 Step 匹配ASM端口

在rfc_wtr2965_non_ca2_4320_sim_0config_data_ag.c文件中，为不同Tech/Mode/Band匹配对应的ASM端口。

例如在 modem_proc/rfc_jolokia/rf_card/rfc_wtr2965_non_ca2_4320_sim_0/cdma/src/rfc_wtr2965_non_ca2_4320_sim_0_cdma_config_data_ag.c中，为cdma 4320 rx配置

查询S5643_52得到band真值表：

查询设备驱动：modem_proc/rfdevice_asm/src/rfdevice_asm_s5643_52_data_ag.cpp得到真值与port对应表

修改代码：

3 MIPI PA Customization

Reference

80-NG377-1_A_MIPI_Device_Customization.pdf

添加或者修改PA设备。

3.1 Step1 PA设备驱动

文件路径/home/wm/items/SIM7600/AMSS_LE20/modem_proc/rfdevice_pa/src

可以完成如下工作：

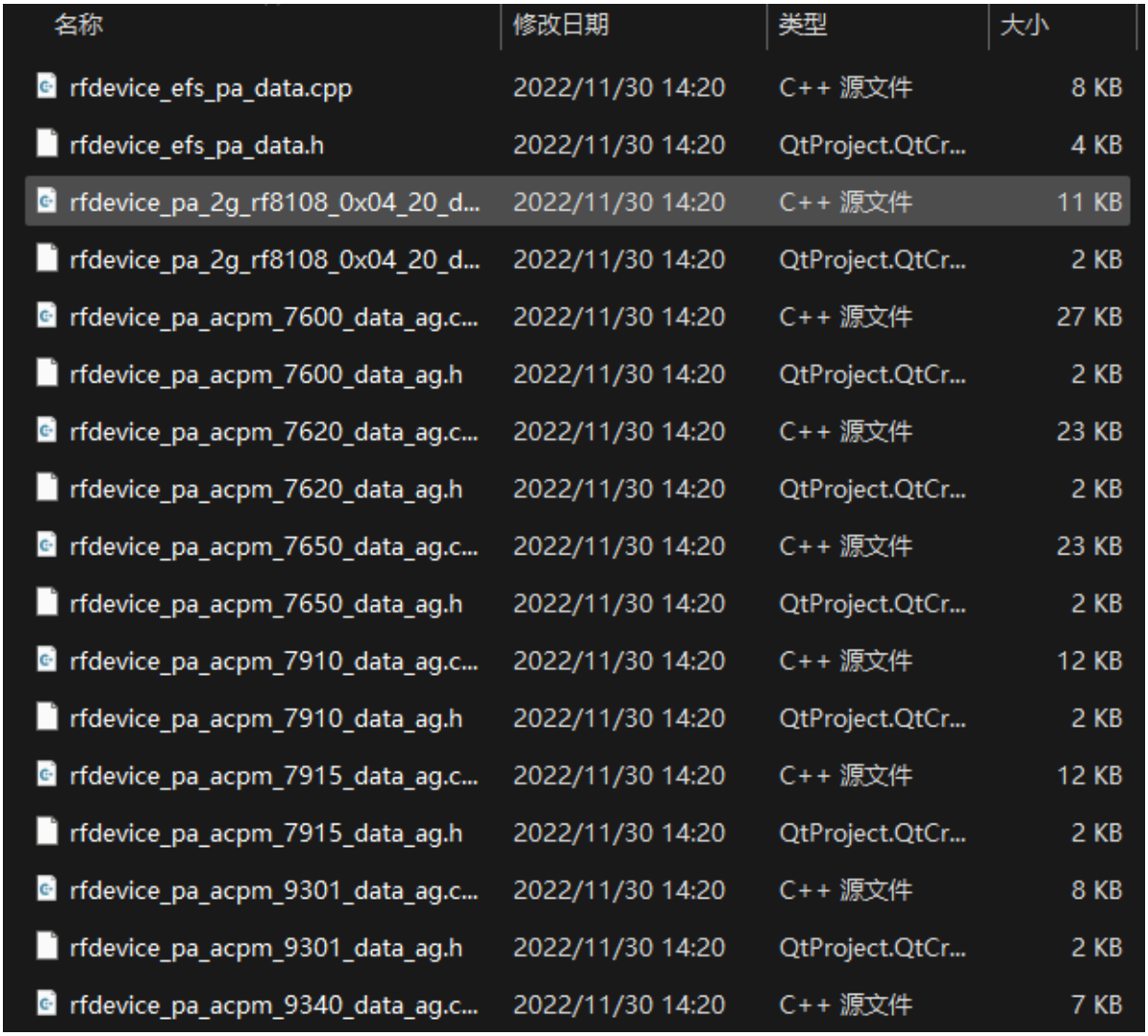
1. 为已经存在的PA设备更改配置，在其对应文件中修改

如：

modem_proc/rfdevice_pa/src/rfdevice_pa_s5643_52_data_ag.cpp和
modem_proc/rfdevice_pa/src/rfdevice_pa_s5643_52_data_ag.h

2. 添加一个新的PA设备

为一个新PA设备添加.h和.cpp文件，.h和.cpp文件内容可以参考已经存在的其他设备的文件。



名称	修改日期	类型	大小
rfdevice_efs_pa_data.cpp	2022/11/30 14:20	C++ 源文件	8 KB
rfdevice_efs_pa_data.h	2022/11/30 14:20	QtProject.QtCr...	4 KB
rfdevice_pa_2g_rf8108_0x04_20_d...	2022/11/30 14:20	C++ 源文件	11 KB
rfdevice_pa_2g_rf8108_0x04_20_d...	2022/11/30 14:20	QtProject.QtCr...	2 KB
rfdevice_pa_acpm_7600_data_ag.c...	2022/11/30 14:20	C++ 源文件	27 KB
rfdevice_pa_acpm_7600_data_ag.h	2022/11/30 14:20	QtProject.QtCr...	2 KB
rfdevice_pa_acpm_7620_data_ag.c...	2022/11/30 14:20	C++ 源文件	23 KB
rfdevice_pa_acpm_7620_data_ag.h	2022/11/30 14:20	QtProject.QtCr...	2 KB
rfdevice_pa_acpm_7650_data_ag.c...	2022/11/30 14:20	C++ 源文件	23 KB
rfdevice_pa_acpm_7650_data_ag.h	2022/11/30 14:20	QtProject.QtCr...	2 KB
rfdevice_pa_acpm_7910_data_ag.c...	2022/11/30 14:20	C++ 源文件	12 KB
rfdevice_pa_acpm_7910_data_ag.h	2022/11/30 14:20	QtProject.QtCr...	2 KB
rfdevice_pa_acpm_7915_data_ag.c...	2022/11/30 14:20	C++ 源文件	12 KB
rfdevice_pa_acpm_7915_data_ag.h	2022/11/30 14:20	QtProject.QtCr...	2 KB
rfdevice_pa_acpm_9301_data_ag.c...	2022/11/30 14:20	C++ 源文件	8 KB
rfdevice_pa_acpm_9301_data_ag.h	2022/11/30 14:20	QtProject.QtCr...	2 KB
rfdevice_pa_acpm_9340_data_ag.c...	2022/11/30 14:20	C++ 源文件	7 KB

在.cpp文件中为PA bias/range/on/off/trigger操作定义寄存器配置

在.cpp文件中为你的PA设备配置正确的MID、PID和product revision、PA范围

MID、PID可以从spec查到

3.2 Step2更新FTM中的PA信息

在文件rfdebice_pa_factory_ag.cpp中：

1、为新添加的PA设备include进.h文件

2、为新添加的PA设备更改或添加程序

3.3 Step3更新common devices list

在RFC common文件中，为你的PA设备更新信息

3.4 Step4匹配PA端口

在rfc_wtr2965_non_ca2_4320_sim_0config_data_ag.c文件中，为不同Tech/Mode/Band匹配对应的PA端口。

例如在 modem_proc/rfc_jolokia/rf_card/rfc_wtr2965_non_ca2_4320_sim_0/cdma/src/rfc_wtr2965_non_ca2_4320_sim_0_cdma_config_data_ag.c中，为wcdma b1 tx0配置

查询spec得到band真值表：

查询设备驱动：modem_proc/rfdevice_pa/src/rfdevice_pa_s5643_52_data_ag.cpp得到真值与port对应表

修改代码：

4 MSM8974/MDM9x25 RFC Code Checklist

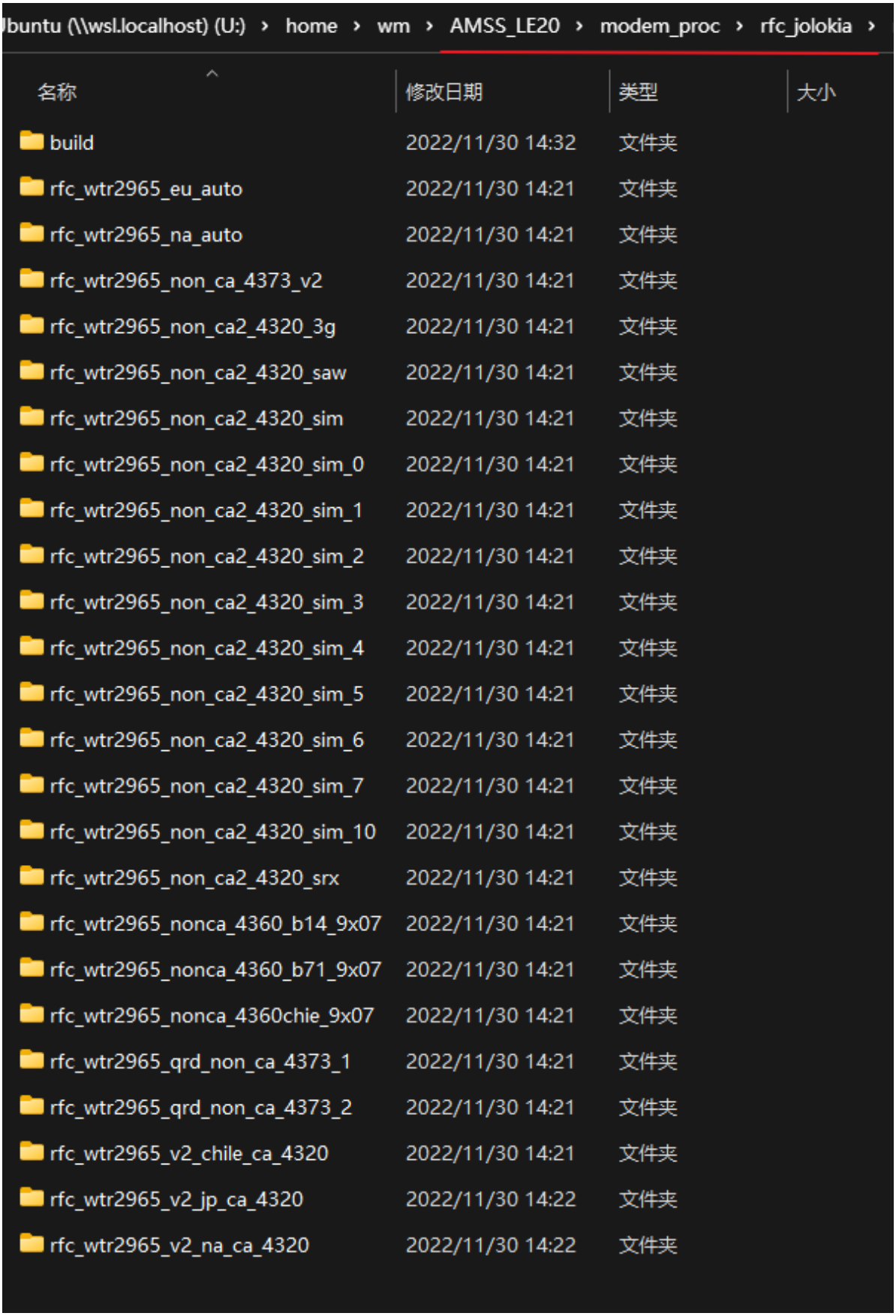
References

80-NA157-179_A_MSM8974_MDM9x25_RFC_Code_Customization_Checklist.pdf

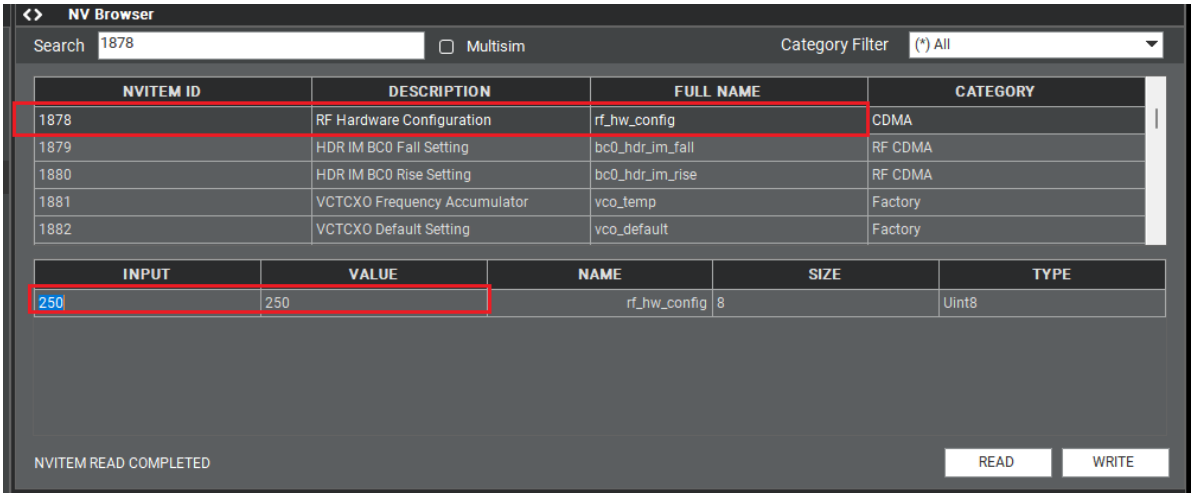
4.1 rf_card类型选择

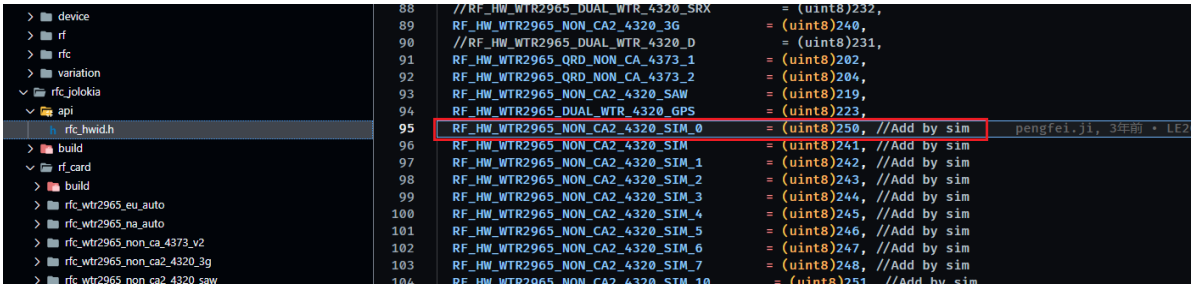
RF卡有许多类型，不同的RF卡对应不同的device list

rf_card文件夹路径`AMSS_LE20/modem_proc/rfc_jolokia/rf_card`，该路径下包含了所有用到的RF卡类型：



在代码编译时，所有的RF cards文件都会被编译，modem使用`NV:1878`来决定实际使用哪个卡。
NV1878数值与RF card类型的对应关系表在文件Rfc_hwid.h。





在AMSS_LE20项目中，读取NV1878值为250，所以RF卡类型应为`RF_HW_WTR2965_NON_CA2_4320_SIM_0`，对应源文件路径`AMSS_LE20/modem_proc/rfc_jolokia/api/rfc_hwid.h`。

注：NA是北美、EU是欧洲。

```
/* ------------------------------------------------------------
** The RF Card Id used in the target
** Note: The Id needs to be sequential
** ------------------------------------------------------------ */
typedef enum {
  RF_HW_UNDEFINED                       = (uint8)0,
  RF_TARGET_NONE                        = RF_HW_UNDEFINED,
  ......
  RF_HW_WTR2965_NON_CA2_4320_SAW        = (uint8)219,
  RF_HW_WTR2965_DUAL_WTR_4320_GPS       = (uint8)223,
  RF_HW_WTR2965_NON_CA2_4320_SIM_0      = (uint8)250, //Add by sim
  RF_HW_WTR2965_NON_CA2_4320_SIM        = (uint8)241, //Add by sim
  ......
} rf_hw_type;
```

4.2 MID、PID、USID

`MANUFACTURER_ID`、`PRODUCT_ID`、`default USID`是不同器件的编号，根据该ID可以区分不同的器件。

在芯片驱动、`rfc_wtr1625_naeu_cmn_devices_list`、`rfc_wtr1625_naeu_<tech>_config_data_ag.c`中都需要为各个芯片设置。

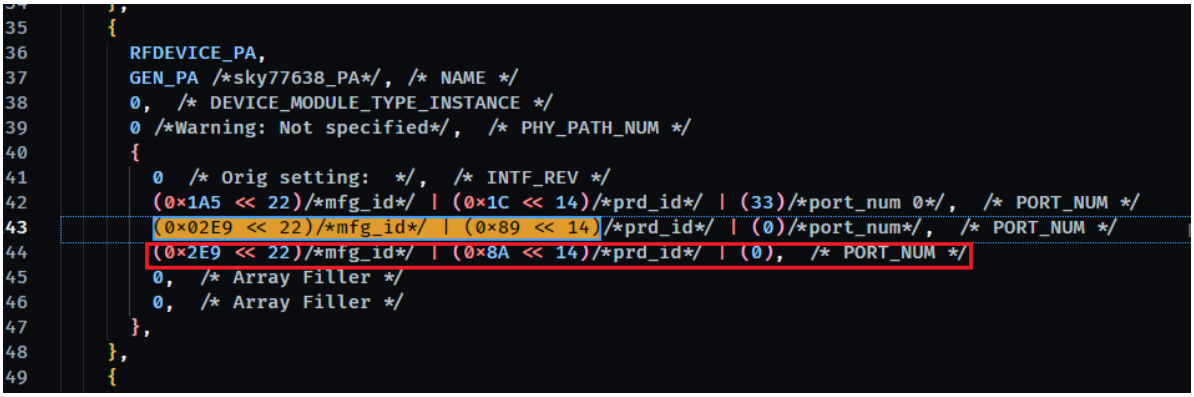
例如AMSS_LE20，在`AMSS_LE20/modem_proc/rfdevice_pa/src/rfdevice_pa_s5643_52_data_ag.cpp`中，`device_info_get()`：

```
931  boolean rfdevice_pa_s5643_52_data_ag::device_info_get( rfdevice_pa_info_type *pa_info )
932  {
933    boolean ret_val = FALSE;
934
935    if ( NULL == pa_info )
936    {
937      return FALSE;
938    }
939    else
940    {
941      pa_info->mfg_id = 0x02E9;
942      pa_info->prd_id = 0x8A;
943      pa_info->prd_rev = 0;
944      pa_info->num_ports = RFDEVICE_PA_S5643_52_NUM_PORTS;
945      pa_info->num_pa_ranges = 4;
946      ret_val = TRUE;
947    }
948    return ret_val;
949  }
```

在`AMSS_LE20/modem_proc/rfc_jolokia/rf_card/rfc_wtr2965_non_ca2_4320_sim_0/common/src/rfc_wtr2965_non_ca2_4320_sim_0_cmn_ag.cpp`中，在`rfc_wtr2965_non_ca2_4320_sim_0_phy_devices_list[]`

```
23     { /*Device: S5643-52 */
24       GEN_DEVICE, /* PHY_DEVICE_NAME */
25       1, /* PHY_DEVICE_INSTANCE */
26       2, /* PHY_DEVICE_ALT_PART_NUM_OF_INSTANCE */
27       RFDEVICE_COMM_PROTO_RFFE, /* PHY_DEVICE_COMM_PROTOCOL */
28     RFDEVICE_COMM_PROTO_VERSION_DEFAULT, /* PHY_DEVICE_COMM_PROTOC
29       {    0,0 /* 0 not specified */,}, /* PHY_DEVICE_COMM_BUS */
30       0x02E9, /* PHY_DEVICE_MANUFACTURER_ID */
31       0x8A, /* PHY_DEVICE_PRODUCT_ID */
32   #ifdef FEATURE_NO_PA_DEBUG
33       0 | RFC_SKIP_RFFE_DETECT_BIT_IND, /* PHY_DEVICE_PRODUCT_REV
34   #else
35       0, /* PHY_DEVICE_PRODUCT_REV */
36   #endif
37       0x0F, /* DEFAULT USID RANGE START */
38       0x0F, /* DEFAULT USID RANGE END */
39       0x0F, /* PHY_DEVICE_ASSIGNED_USID */
40       0 /*Warning: Not specified*/, /* RFFE_GROUP_ID */
41       FALSE, /* INIT */
42       RFC_INVALID_PARAM, /* ASSOCIATED_DAC */
43     }, /* END - Device: S5643 */
```

在`AMSS_LE20/modem_proc/rfc_jolokia/rf_card/rfc_wtr2965_non_ca2_4320_sim_0/lte/src/rfc_wtr2965_non_ca2_4320_sim_0_lte_config_data_ag.c`中：



注意确保rfc_wtr1625_naeu_cmn_ag.cpp中`rfc_wtr2965_non_ca2_4320_sim_0_phy_devices_list[]`器件列表的器件与实际硬件电路设计一致。MIPI device信息，如MANUFACTURER_ID、PRODUCT_ID、default USID、ASSIGNED_USID需要根据实际使用的器件改动。

详细方法可以参考《80-NG377-1 Presentation:MIPI Device Customization》。

注意：

1. MANUFACTURER_ID、PRODUCT_ID从芯片的spec中查到
2. ASSIGNED_USID为研发自己设定，需要注意相同MANUFACTURER_ID的不同Device，其PRODUCT_ID和DEVICE_TYPE_INSTANCE不同

4.3 DEVICE_TYPE_INSTANCE

`DEVICE_TYPE_INSTANCE`参数用来标明电路板上相同类型设备的不同元器件。

如果板子上相同类型设备元器件的数目超过一个，比如PA、ASM、天线调节器。。。就用不同的ID来标记他们，如0、1、2...，用来作为他们的`DEVICE_TYPE_INSTANCE`。

注意：

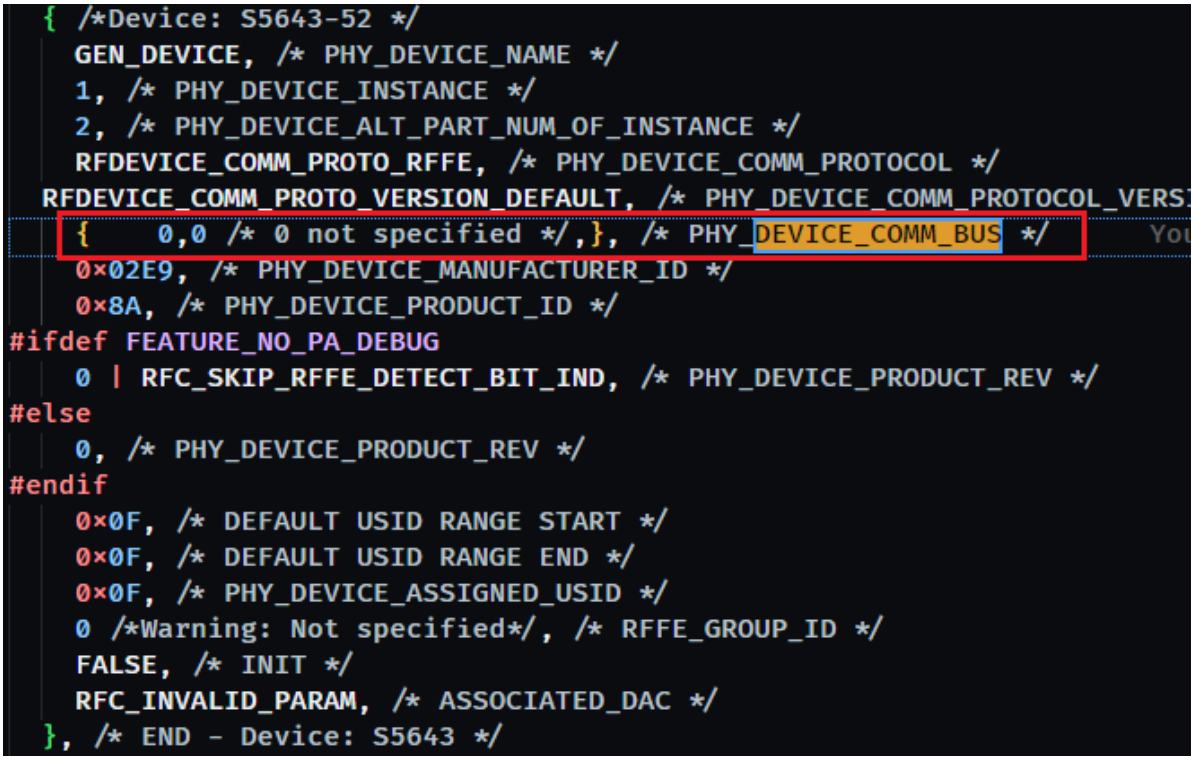
> 同一元器件的DEVICE_TYPE_INSTANCE在`rfc_wtr2965_non_ca2_4320_sim_0_phy_devices_list[]`和`rfc_wtr2965_non_ca2_4320_sim_0_lte_config_data_ag.c`中要一样。

例如：

在`rfc_wtr2965_non_ca2_4320_sim_0_phy_devices_list[]`中DEVICE_TYPE_INSTANCE值为1，在`rfc_wtr2965_non_ca2_4320_sim_0_lte_config_data_ag.c`中DEVICE_TYPE_INSTANCE值也为1

4.4 DEVICE_COMM_BUS

DEVICE_COMM_BUS的第一个参数用来指定连接到的MIPI device的MIPI RFFE bus.



0表示第一个RFFE bug，而2表示第二个, 该数值根据MIPI device的实际连线来确定。

4.5 删除用不到的device

如果QFE device(QFE1100/1101、QFE1510)在设计中没有使用，那么就将他们从device list中删除。其它device也一样，如果没有用到，就删除掉。比如PA、ASM等等。

同样，从CDMA/GSM/LTE/WCDMA每个band的配置文件中删除没有用到的device。比如，在文件`rfc_wtr2965_non_ca2_4320_sim_0_cdma_config_data_ag.c`中，

当从device list中增加或者删除device时候，确保相应修改NUM_DEVICES_TO_CONFIGURE参数。该参数应该根据具体方案的设计来设定。

此时`NUM_DEVICES_TO_CONFIGURE`为**6**，如果删除一个device的话，则`NUM_DEVICES_TO_CONFIGURE`应该改成**5**。

```
199  rfc_device_info_type rf_card_wtr2965_non_ca2_4320_sim_0_tx0_wcdma_b1_device_info =
200  {
201    RFC_ENCODED_REVISION,
202    RFC_TX_MODEM_CHAIN_0,    /* Modem Chain */
203    0,    /* NV Container */
204    0,    /* Antenna */
205    5,  /* NUM_DEVICES_TO_CONFIGURE */     pengfei.ji, 22个月前 • [CR][NO][RFC] add 7600G RFC …
206    {
207      { (enum <unnamed>)RFDEVICE_TRANSCEIVER = 0
208        RFDEVICE_TRANSCEIVER,
209        WTR2965, /* NAME */
210        0,    /* DEVICE_MODULE_TYPE_INSTANCE */
211        0,    /* PHY_PATH_NUM */
212        {
213          0 /* Warning: Not specified */,    /* INTF_REV */
214          (int)WTR2965_WCDMA_TX_BAND1_THMLB4,    /* PORT */
215          ( RFDEVICE_PA_LUT_MAPPING_VALID | WTR2965_LP_LUT_TYPE << RFDEVICE_PA_STATE_0_BSHFT | WTR2965_HP_LUT_TY
216          FALSE,    /* TXAGC_LUT */
217          WTR2965_FBRX_LOW_ATTN_MODE,    /* FBRX_ATTN_STATE */
218          0,  /* Array Filler */
219        },
220      },
221      {
222        RFDEVICE_ASM,
223        GEN_ASM /*sky77916_ASM_with_gsm_pa*/, /* NAME */
224        0,  /* DEVICE_MODULE_TYPE_INSTANCE */
225        0 /*Warning: Not specified*/,  /* PHY_PATH_NUM */
226        {
227          0  /* Orig setting:  */,  /* INTF_REV */
228          (0x1A5 << 22)/*mfg_id*/ | (0x96 << 14)/*prd_id*/ | (15)/*port_num(11)*/,  /* PORT_NUM */
229          (0x02E9 << 22)/*mfg_id*/ | (0x29 << 14)/*prd_id*/ | (15)/*port_num*/,  /* PORT_NUM */
230          0,  /* Array Filler */
231          0,  /* Array Filler */
232          0,  /* Array Filler */
233        },
234      },
```

4.6 端口匹配

确保端口匹配。不同technology中每个band的port软件设定应与硬件设计一致。例如：

`rf_card_wtr2965_non_ca2_4320_sim_0_rx0_wcdma_b1_sig_cfg` 定义如下：

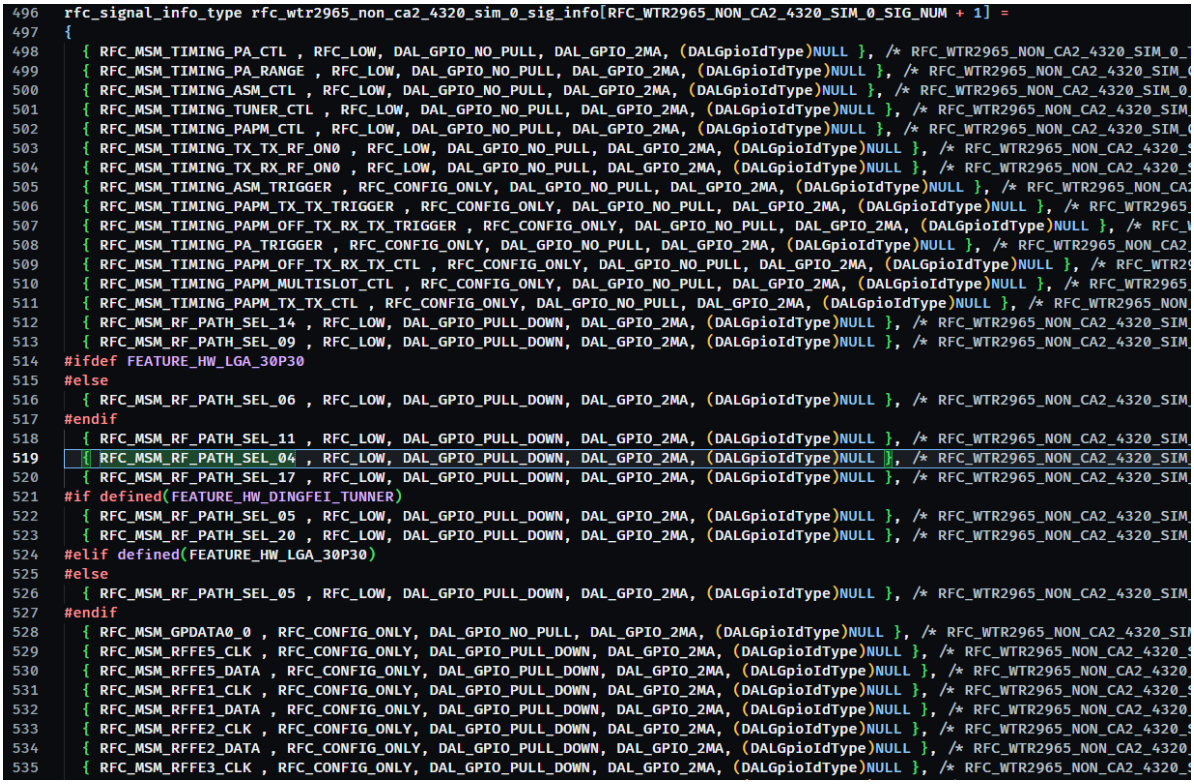
`modem_proc/rfc_jolokia/rf_card/rfc_wtr2965_non_ca2_4320_sim_0/wcdma/src/rfc_wtr2965_non_ca2_4320_sim_0_wcdma_config_data_ag.c`

```
145  rfc_sig_info_type rf_card_wtr2965_non_ca2_4320_sim_0_rx0_wcdma_b1_sig_cfg =
146  {
147    RFC_ENCODED_REVISION,
148    {
149  #ifdef FEATURE_HW_DINGFEI_TUNNER
150      { (int)RFC_WTR2965_NON_CA2_4320_SIM_0_RF_PATH_SEL_05,    { RFC_LOW, -0 }, {RFC_LOW, 0 }  },
151      { (int)RFC_WTR2965_NON_CA2_4320_SIM_0_RF_PATH_SEL_20,    { RFC_LOW, -0 }, {RFC_LOW, 0 }  },
152  #endif
153      { (int)RFC_WTR2965_NON_CA2_4320_SIM_0_RF_PATH_SEL_04,    { RFC_LOW, 0 }, {RFC_LOW, 0 }  },
154      { (int)RFC_SIG_LIST_END,    { RFC_LOW, 0 }, {RFC_LOW, 0 } }
155    },
156  };
```

`rf_card_wtr2965_non_ca2_4320_sim_0_rx0_wcdma_b1_sig_cfg.cfg_sig_list[0]` 的sig_name之所以选择 `RFC_WTR2965_NON_CA2_4320_SIM_0_RF_PATH_SEL_04` 的原因：

倒查如下，

1. 在文件 `modem_proc/rfc_jolokia/rf_card/rfc_wtr2965_non_ca2_4320_sim_0/common/src/rfc_wtr2965_non_ca2_4320_sim_0_cmn_ag.cpp` 中，



```
rfc_signal_info_type rfc_wtr2965_non_ca2_4320_sim_0_sig_info[RFC_WTR2965_NON_CA2_4320_SIM_0_SIG_NUM + 1] =
{
  { RFC_MSM_TIMING_PA_CTL , RFC_LOW, DAL_GPIO_NO_PULL, DAL_GPIO_2MA, (DALGpioIdType)NULL }, /* RFC_WTR2965_NON_CA2_4320_SIM_0_
  { RFC_MSM_TIMING_PA_RANGE , RFC_LOW, DAL_GPIO_NO_PULL, DAL_GPIO_2MA, (DALGpioIdType)NULL }, /* RFC_WTR2965_NON_CA2_4320_SIM_0
  { RFC_MSM_TIMING_ASM_CTL , RFC_LOW, DAL_GPIO_NO_PULL, DAL_GPIO_2MA, (DALGpioIdType)NULL }, /* RFC_WTR2965_NON_CA2_4320_SIM_0_
  { RFC_MSM_TIMING_TUNER_CTL , RFC_LOW, DAL_GPIO_NO_PULL, DAL_GPIO_2MA, (DALGpioIdType)NULL }, /* RFC_WTR2965_NON_CA2_4320_SIM_
  { RFC_MSM_TIMING_PAPM_CTL , RFC_LOW, DAL_GPIO_NO_PULL, DAL_GPIO_2MA, (DALGpioIdType)NULL }, /* RFC_WTR2965_NON_CA2_4320_SIM_0
  { RFC_MSM_TIMING_TX_TX_RF_ON0 , RFC_LOW, DAL_GPIO_NO_PULL, DAL_GPIO_2MA, (DALGpioIdType)NULL }, /* RFC_WTR2965_NON_CA2_4320_S
  { RFC_MSM_TIMING_TX_RX_RF_ON0 , RFC_LOW, DAL_GPIO_NO_PULL, DAL_GPIO_2MA, (DALGpioIdType)NULL }, /* RFC_WTR2965_NON_CA2_4320_S
  { RFC_MSM_TIMING_ASM_TRIGGER , RFC_CONFIG_ONLY, DAL_GPIO_NO_PULL, DAL_GPIO_2MA, (DALGpioIdType)NULL }, /* RFC_WTR2965_NON_CA2
  { RFC_MSM_TIMING_PAPM_TX_TX_TRIGGER , RFC_CONFIG_ONLY, DAL_GPIO_NO_PULL, DAL_GPIO_2MA, (DALGpioIdType)NULL }, /* RFC_WTR2965_
  { RFC_MSM_TIMING_PAPM_OFF_TX_RX_TX_TRIGGER , RFC_CONFIG_ONLY, DAL_GPIO_NO_PULL, DAL_GPIO_2MA, (DALGpioIdType)NULL }, /* RFC_W
  { RFC_MSM_TIMING_PA_TRIGGER , RFC_CONFIG_ONLY, DAL_GPIO_NO_PULL, DAL_GPIO_2MA, (DALGpioIdType)NULL }, /* RFC_WTR2965_NON_CA2_
  { RFC_MSM_TIMING_PAPM_OFF_TX_RX_TX_CTL , RFC_CONFIG_ONLY, DAL_GPIO_NO_PULL, DAL_GPIO_2MA, (DALGpioIdType)NULL }, /* RFC_WTR2965_
  { RFC_MSM_TIMING_PAPM_MULTISLOT_CTL , RFC_CONFIG_ONLY, DAL_GPIO_NO_PULL, DAL_GPIO_2MA, (DALGpioIdType)NULL }, /* RFC_WTR2965_
  { RFC_MSM_TIMING_PAPM_TX_TX_CTL , RFC_CONFIG_ONLY, DAL_GPIO_NO_PULL, DAL_GPIO_2MA, (DALGpioIdType)NULL }, /* RFC_WTR2965_NON_
  { RFC_MSM_RF_PATH_SEL_14 , RFC_LOW, DAL_GPIO_PULL_DOWN, DAL_GPIO_2MA, (DALGpioIdType)NULL }, /* RFC_WTR2965_NON_CA2_4320_SIM_
  { RFC_MSM_RF_PATH_SEL_09 , RFC_LOW, DAL_GPIO_PULL_DOWN, DAL_GPIO_2MA, (DALGpioIdType)NULL }, /* RFC_WTR2965_NON_CA2_4320_SIM_
#ifdef FEATURE_HW_LGA_30P30
#else
  { RFC_MSM_RF_PATH_SEL_06 , RFC_LOW, DAL_GPIO_PULL_DOWN, DAL_GPIO_2MA, (DALGpioIdType)NULL }, /* RFC_WTR2965_NON_CA2_4320_SIM_
#endif
  { RFC_MSM_RF_PATH_SEL_11 , RFC_LOW, DAL_GPIO_PULL_DOWN, DAL_GPIO_2MA, (DALGpioIdType)NULL }, /* RFC_WTR2965_NON_CA2_4320_SIM_
  { RFC_MSM_RF_PATH_SEL_04 , RFC_LOW, DAL_GPIO_PULL_DOWN, DAL_GPIO_2MA, (DALGpioIdType)NULL }, /* RFC_WTR2965_NON_CA2_4320_SIM_
  { RFC_MSM_RF_PATH_SEL_17 , RFC_LOW, DAL_GPIO_PULL_DOWN, DAL_GPIO_2MA, (DALGpioIdType)NULL }, /* RFC_WTR2965_NON_CA2_4320_SIM_
#if defined(FEATURE_HW_DINGFEI_TUNNER)
  { RFC_MSM_RF_PATH_SEL_05 , RFC_LOW, DAL_GPIO_PULL_DOWN, DAL_GPIO_2MA, (DALGpioIdType)NULL }, /* RFC_WTR2965_NON_CA2_4320_SIM_
  { RFC_MSM_RF_PATH_SEL_20 , RFC_LOW, DAL_GPIO_PULL_DOWN, DAL_GPIO_2MA, (DALGpioIdType)NULL }, /* RFC_WTR2965_NON_CA2_4320_SIM_
#elif defined(FEATURE_HW_LGA_30P30)
#else
  { RFC_MSM_RF_PATH_SEL_05 , RFC_LOW, DAL_GPIO_PULL_DOWN, DAL_GPIO_2MA, (DALGpioIdType)NULL }, /* RFC_WTR2965_NON_CA2_4320_SIM_
#endif
  { RFC_MSM_GPDATA0_0 , RFC_CONFIG_ONLY, DAL_GPIO_NO_PULL, DAL_GPIO_2MA, (DALGpioIdType)NULL }, /* RFC_WTR2965_NON_CA2_4320_SI
  { RFC_MSM_RFFE5_CLK , RFC_CONFIG_ONLY, DAL_GPIO_PULL_DOWN, DAL_GPIO_2MA, (DALGpioIdType)NULL }, /* RFC_WTR2965_NON_CA2_4320_S
  { RFC_MSM_RFFE5_DATA , RFC_CONFIG_ONLY, DAL_GPIO_PULL_DOWN, DAL_GPIO_2MA, (DALGpioIdType)NULL }, /* RFC_WTR2965_NON_CA2_4320_
  { RFC_MSM_RFFE1_CLK , RFC_CONFIG_ONLY, DAL_GPIO_PULL_DOWN, DAL_GPIO_2MA, (DALGpioIdType)NULL }, /* RFC_WTR2965_NON_CA2_4320_S
  { RFC_MSM_RFFE1_DATA , RFC_CONFIG_ONLY, DAL_GPIO_PULL_DOWN, DAL_GPIO_2MA, (DALGpioIdType)NULL }, /* RFC_WTR2965_NON_CA2_4320_
  { RFC_MSM_RFFE2_CLK , RFC_CONFIG_ONLY, DAL_GPIO_PULL_DOWN, DAL_GPIO_2MA, (DALGpioIdType)NULL }, /* RFC_WTR2965_NON_CA2_4320_S
  { RFC_MSM_RFFE2_DATA , RFC_CONFIG_ONLY, DAL_GPIO_PULL_DOWN, DAL_GPIO_2MA, (DALGpioIdType)NULL }, /* RFC_WTR2965_NON_CA2_4320_
  { RFC_MSM_RFFE3_CLK , RFC_CONFIG_ONLY, DAL_GPIO_PULL_DOWN, DAL_GPIO_2MA, (DALGpioIdType)NULL }, /* RFC_WTR2965_NON_CA2_4320_S
```

在`rfc_wtr2965_non_ca2_4320_sim_0_sig_info[RFC_WTR2965_NON_CA2_4320_SIM_0_SIG_NUM + 1]`列表中`RFC_MSM_RF_PATH_SEL_04`排序是第18，而在`AMSS_LE20/modem_proc/rfc_jolokia/rf_card/rfc_wtr2965_non_ca2_4320_sim_0/common/inc/rfc_wtr2965_non_ca2_4320_sim_0_cmn_ag.h`中的wtr1625_naeu_sig_type的定义中`RFC_WTR2965_NON_CA2_4320_SIM_0_RF_PATH_SEL_04`排序也是第18，这样RFC_WTR2965_NON_CA2_4320_SIM_0_RF_PATH_SEL_04和RFC_MSM_RF_PATH_SEL_04就对应起来了。

```
39  typedef enum
40  {
41    RFC_WTR2965_NON_CA2_4320_SIM_0_TIMING_PA_CTL,
42    RFC_WTR2965_NON_CA2_4320_SIM_0_TIMING_PA_RANGE,
43    RFC_WTR2965_NON_CA2_4320_SIM_0_TIMING_ASM_CTL,
44    RFC_WTR2965_NON_CA2_4320_SIM_0_TIMING_TUNER_CTL,
45    RFC_WTR2965_NON_CA2_4320_SIM_0_TIMING_PAPM_CTL,
46    RFC_WTR2965_NON_CA2_4320_SIM_0_TIMING_TX_TX_RF_ON0,
47    RFC_WTR2965_NON_CA2_4320_SIM_0_TIMING_TX_RX_RF_ON0,
48    RFC_WTR2965_NON_CA2_4320_SIM_0_TIMING_ASM_TRIGGER,
49    RFC_WTR2965_NON_CA2_4320_SIM_0_TIMING_PAPM_TX_TX_TRIGGER,
50    RFC_WTR2965_NON_CA2_4320_SIM_0_TIMING_PAPM_OFF_TX_RX_TX_TRIGGER,
51    RFC_WTR2965_NON_CA2_4320_SIM_0_TIMING_PA_TRIGGER,
52    RFC_WTR2965_NON_CA2_4320_SIM_0_TIMING_PAPM_OFF_TX_RX_TX_CTL,
53    RFC_WTR2965_NON_CA2_4320_SIM_0_TIMING_PAPM_MULTISLOT_CTL,
54    RFC_WTR2965_NON_CA2_4320_SIM_0_TIMING_PAPM_TX_TX_CTL,
55    RFC_WTR2965_NON_CA2_4320_SIM_0_RF_PATH_SEL_14,
56    RFC_WTR2965_NON_CA2_4320_SIM_0_RF_PATH_SEL_09,
57  #ifdef FEATURE_HW_LGA_30P30
58    /*GPIO52 was used for status in the 30*30 PCB*/
59  #else
60    RFC_WTR2965_NON_CA2_4320_SIM_0_RF_PATH_SEL_06,
61  #endif
62    RFC_WTR2965_NON_CA2_4320_SIM_0_RF_PATH_SEL_11,
63    RFC_WTR2965_NON_CA2_4320_SIM_0_RF_PATH_SEL_04,      pengfei.ji, 3年前 • LE20-440:S
64    RFC_WTR2965_NON_CA2_4320_SIM_0_RF_PATH_SEL_17,
65  #if defined(FEATURE_HW_DINGFEI_TUNNER)
66    RFC_WTR2965_NON_CA2_4320_SIM_0_RF_PATH_SEL_05,
67    RFC_WTR2965_NON_CA2_4320_SIM_0_RF_PATH_SEL_20,
68  #elif defined(FEATURE_HW_LGA_30P30)
69    /*GPIO50 was used for ring in the 30*30 PCB*/
70  #else
71    RFC_WTR2965_NON_CA2_4320_SIM_0_RF_PATH_SEL_05,
72  #endif
73    RFC_WTR2965_NON_CA2_4320_SIM_0_GPDATA0_0,
74    RFC_WTR2965_NON_CA2_4320_SIM_0_RFFE5_CLK,
75    RFC_WTR2965_NON_CA2_4320_SIM_0_RFFE5_DATA,
```

2. 在文件`modem_proc/rfc_jolokia/target/mdm9607/src/rfc_msm_signal_info_ag.c`中

```
rfc_msm_signal_info_type rfc_mdm9607_signal_info[RFC_MSM_SIG_NUM] =
{
......
    {  RFC_ANT_SEL, 47, 4, RFC_GRFC, 1, DAL_GPIO_OUTPUT, "grfc[4]"},  /* Signal:
RFC_MSM_RF_PATH_SEL_04, MSM Pin Name: GPIO_47*/
......
};
```

在`rfc_mdm9607_signal_info[RFC_MSM_SIG_NUM]`列表中，`grfc number`也是4，而`GPIO`口对应的是47口。

rfc_msm_signal_info_type的定义如下：

```
typedef struct
{
  rfc_signal_type signal_type;
  uint32 msm_gpio;
  uint8 grfc_num;
  rfc_gpio_grfc_type output_type;
  uint8 function_select;
  DALGpioDirectionType direction;
  char* tlmm_gpio_name;
} rfc_msm_signal_info_type;
```

即`GPIO 47`在硬件设计中是配置为RFC_WTR2965_NON_CA2_4320_SIM_0_RF_PATH_SEL_04的。

5 RFC wtr2965 ca2 config

下面以rfc wtr2965 ca2 config为例，来介绍rfc wtr2965 ca2 config代码

codes

> modem_proc/rfc_jolokia/rf_card/rfc_wtr2965_non_ca2_4320_sim_0/wcdma/inc/rfc_wtr2965_non_ca2_4320_sim_0_wcdma_config_ag.h
>
> modem_proc/rfc_jolokia/rf_card/rfc_wtr2965_non_ca2_4320_sim_0/wcdma/src/rfc_wtr2965_non_ca2_4320_sim_0_wcdma_config_ag.cpp
>
> modem_proc/rfc_jolokia/rf_card/rfc_wtr2965_non_ca2_4320_sim_0/wcdma/src/rfc_wtr2965_non_ca2_4320_sim_0_wcdma_config_data_ag.c

5.1 获取signal config data

signal config data通过函数sig_cfg_data_get()获取，该函数所在路径为：

`modem_proc/rfc_jolokia/rf_card/rfc_wtr2965_non_ca2_4320_sim_0/wcdma/src/rfc_wtr2965_non_ca2_4320_sim_0_wcdma_config_ag.cpp`

函数定义如下：

```
boolean rfc_wtr2965_non_ca2_4320_sim_0_wcdma_ag::sig_cfg_data_get(
rfc_cfg_params_type *cfg, rfc_sig_cfg_type **ptr )
{
  ......
  if ( ( cfg->rx_tx == RFC_CONFIG_RX ) && ( cfg->logical_device == RFM_DEVICE_0
) && ( cfg->alternate_path == 0 /*Warning: not specified*/ ) && ( cfg->band ==
(int)RFCOM_BAND_IMT ) && ( cfg->req == RFC_REQ_DEFAULT_GET_DATA ) && !ret_val )
  { *ptr = &
(rf_card_wtr2965_non_ca2_4320_sim_0_rx0_wcdma_b1_sig_cfg.cfg_sig_list[0]);
 ret_val = TRUE; }

  if ( ( cfg->rx_tx == RFC_CONFIG_RX ) && ( cfg->logical_device == RFM_DEVICE_1
) && ( cfg->alternate_path == 0 /*Warning: not specified*/ ) && ( cfg->band ==
(int)RFCOM_BAND_IMT ) && ( cfg->req == RFC_REQ_DEFAULT_GET_DATA ) && !ret_val )
  { *ptr = &
(rf_card_wtr2965_non_ca2_4320_sim_0_rx1_wcdma_b1_sig_cfg.cfg_sig_list[0]);
 ret_val = TRUE; }

  if ( ( cfg->rx_tx == RFC_CONFIG_TX ) && ( cfg->logical_device == RFM_DEVICE_0
) && ( cfg->alternate_path == 0 /*Warning: not specified*/ ) && ( cfg->band ==
(int)RFCOM_BAND_IMT ) && ( cfg->req == RFC_REQ_DEFAULT_GET_DATA ) && !ret_val )
  { *ptr = &
(rf_card_wtr2965_non_ca2_4320_sim_0_tx0_wcdma_b1_sig_cfg.cfg_sig_list[0]);
 ret_val = TRUE; }
  ......
}
```

Band class定义如下：

`AMSS_LE20/modem_proc/rfa/api/cdma/rfm_cdma_band_types.h`

```
/*!
  @brief
  Band class definitions as specified by 3GPP2 C.S0057.
*/
typedef enum rfm_band_class_e
{
  RFM_CDMA_BC0  = 0,  /*!< 800 MHz Band                                */
  RFM_CDMA_BC1  = 1,  /*!< 1900 MHz Band                               */
  RFM_CDMA_BC2  = 2,  /*!< TACS Band                                   */
  RFM_CDMA_BC3  = 3,  /*!< JTACS Band                                  */
  RFM_CDMA_BC4  = 4,  /*!< Korean PCS Band                             */
  RFM_CDMA_BC5  = 5,  /*!< 450 MHz Band                                */
  RFM_CDMA_BC6  = 6,  /*!< 2 GHz Band                                  */
  RFM_CDMA_BC7  = 7,  /*!< Upper 700 MHz Band                          */
  RFM_CDMA_BC8  = 8,  /*!< 1800 MHz Band                               */
  RFM_CDMA_BC9  = 9,  /*!< 900 MHz Band                                */
  RFM_CDMA_BC10 = 10, /*!< Secondary 800 MHz Band                      */
  RFM_CDMA_BC11 = 11, /*!< 400 MHz European PAMR Band                  */
  RFM_CDMA_BC12 = 12, /*!< 800 MHz PAMR Band                           */
  RFM_CDMA_BC13 = 13, /*!< 2.5 GHz IMT-2000 Extension Band             */
  RFM_CDMA_BC14 = 14, /*!< US PCS 1.9GHz Band                          */
  RFM_CDMA_BC15 = 15, /*!< AWS Band                                    */
  RFM_CDMA_BC16 = 16, /*!< US 2.5GHz Band                              */
  RFM_CDMA_BC17 = 17, /*!< US 2.5GHz Forward Link Only Band            */
  RFM_CDMA_BC18 = 18, /*!< 700 MHz Public Safety Band                  */
  RFM_CDMA_BC19 = 19, /*!< Lower 700 MHz Band                          */
  RFM_CDMA_BC20 = 20, /*!< L-Band                                      */

  RFM_CDMA_MAX_BAND   /*!< Terminal value for the enum, not a valid
                           band                                        */
} rfm_cdma_band_class_type;
```

以rf_card_wtr2965_non_ca2_4320_sim_0_rx0_wcdma_b1_sig_cfg、rf_card_wtr2965_non_ca2_4320_sim_0_rx1_wcdma_b1_sig_cfg和rf_card_wtr2965_non_ca2_4320_sim_0_tx0_wcdma_b1_sig_cfg为例，介绍signal config data的配置。

1、rf_card_wtr2965_non_ca2_4320_sim_0_rx0_wcdma_b1_sig_cfg

rf_card_wtr2965_non_ca2_4320_sim_0_rx0_wcdma_b1_sig_cfg定义如下：

modem_proc/rfc_jolokia/rf_card/rfc_wtr2965_non_ca2_4320_sim_0/wcdma/src/rfc_wtr2965_non_ca2_4320_sim_0_wcdma_config_data_ag.c

```
rfc_sig_info_type rf_card_wtr2965_non_ca2_4320_sim_0_rx0_wcdma_b1_sig_cfg =
{
  RFC_ENCODED_REVISION,
  {
#ifdef FEATURE_HW_DINGFEI_TUNNER
    { (int)RFC_WTR2965_NON_CA2_4320_SIM_0_RF_PATH_SEL_05,   { RFC_LOW, -0 },
{RFC_LOW, 0 }  },
    { (int)RFC_WTR2965_NON_CA2_4320_SIM_0_RF_PATH_SEL_20,   { RFC_LOW, -0 },
{RFC_LOW, 0 }  },
#endif
    { (int)RFC_WTR2965_NON_CA2_4320_SIM_0_RF_PATH_SEL_04,   { RFC_LOW, 0 },
{RFC_LOW, 0 }  },
    { (int)RFC_SIG_LIST_END,   { RFC_LOW, 0 }, {RFC_LOW, 0 } }
  },
};
```

rfc_sig_info_type定义为:

```
typedef struct
{
  uint32 rfc_revision;
  rfc_sig_cfg_type cfg_sig_list[];
} rfc_sig_info_type;
```

rfc_sig_cfg_type定义为:

```
typedef struct
{
  int sig_name;
  rfc_sig_timing_info_type start;
  rfc_sig_timing_info_type stop;
} rfc_sig_cfg_type;
```

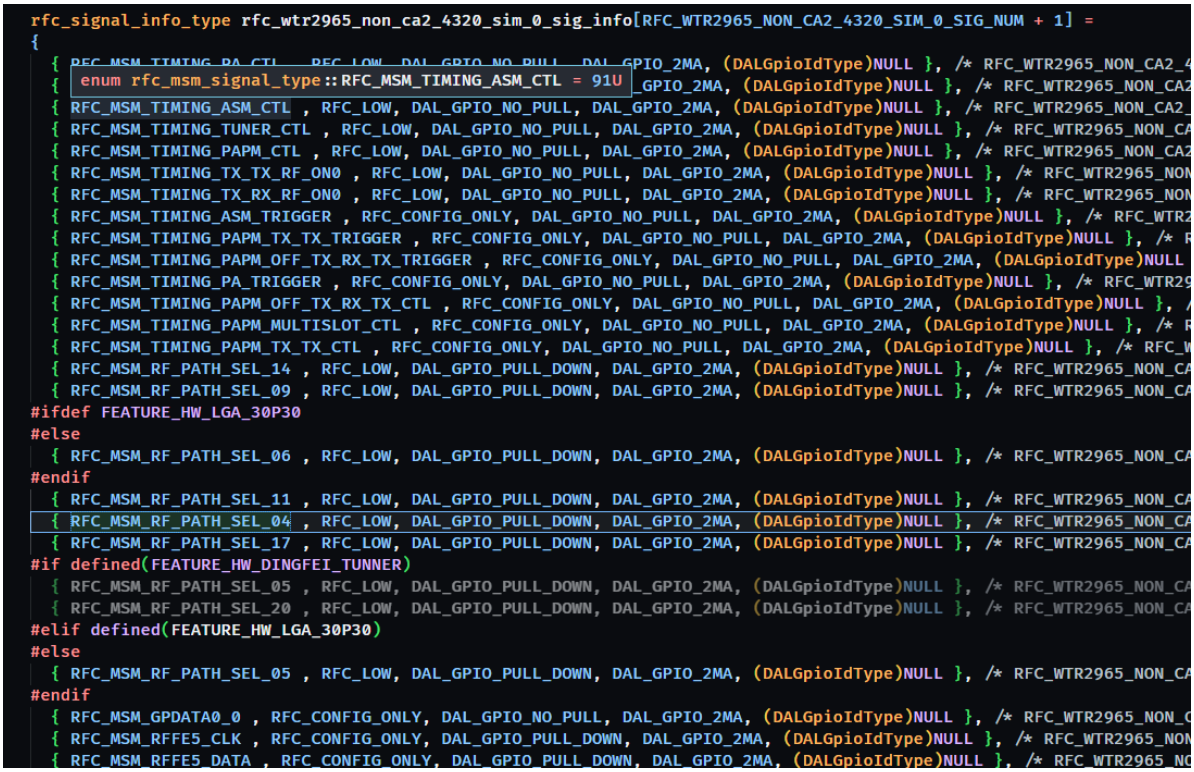
sig_name定义为:

```
typedef enum
{
  RFC_WTR2965_NON_CA2_4320_SIM_0_TIMING_PA_CTL,
  RFC_WTR2965_NON_CA2_4320_SIM_0_TIMING_PA_RANGE,
  RFC_WTR2965_NON_CA2_4320_SIM_0_TIMING_ASM_CTL,
  RFC_WTR2965_NON_CA2_4320_SIM_0_TIMING_TUNER_CTL,
  RFC_WTR2965_NON_CA2_4320_SIM_0_TIMING_PAPM_CTL,
  RFC_WTR2965_NON_CA2_4320_SIM_0_TIMING_TX_TX_RF_ON0,
  RFC_WTR2965_NON_CA2_4320_SIM_0_TIMING_TX_RX_RF_ON0,
  RFC_WTR2965_NON_CA2_4320_SIM_0_TIMING_ASM_TRIGGER,
  RFC_WTR2965_NON_CA2_4320_SIM_0_TIMING_PAPM_TX_TX_TRIGGER,
  RFC_WTR2965_NON_CA2_4320_SIM_0_TIMING_PAPM_OFF_TX_RX_TX_TRIGGER,
  RFC_WTR2965_NON_CA2_4320_SIM_0_TIMING_PA_TRIGGER,
  RFC_WTR2965_NON_CA2_4320_SIM_0_TIMING_PAPM_OFF_TX_RX_TX_CTL,
  RFC_WTR2965_NON_CA2_4320_SIM_0_TIMING_PAPM_MULTISLOT_CTL,
  RFC_WTR2965_NON_CA2_4320_SIM_0_TIMING_PAPM_TX_TX_CTL,
  RFC_WTR2965_NON_CA2_4320_SIM_0_RF_PATH_SEL_14,
  RFC_WTR2965_NON_CA2_4320_SIM_0_RF_PATH_SEL_09,
```

```
#ifdef FEATURE_HW_LGA_30P30
  /*GPIO52 was used for status in the 30*30 PCB*/
#else
  RFC_WTR2965_NON_CA2_4320_SIM_0_RF_PATH_SEL_06,
#endif
  RFC_WTR2965_NON_CA2_4320_SIM_0_RF_PATH_SEL_11,
  RFC_WTR2965_NON_CA2_4320_SIM_0_RF_PATH_SEL_04,
  RFC_WTR2965_NON_CA2_4320_SIM_0_RF_PATH_SEL_17,
#if defined(FEATURE_HW_DINGFEI_TUNNER)
  RFC_WTR2965_NON_CA2_4320_SIM_0_RF_PATH_SEL_05,
  RFC_WTR2965_NON_CA2_4320_SIM_0_RF_PATH_SEL_20,
#elif defined(FEATURE_HW_LGA_30P30)
  /*GPIO50 was used for ring in the 30*30 PCB*/
#else
  RFC_WTR2965_NON_CA2_4320_SIM_0_RF_PATH_SEL_05,
#endif
  RFC_WTR2965_NON_CA2_4320_SIM_0_GPDATA0_0,
  RFC_WTR2965_NON_CA2_4320_SIM_0_RFFE5_CLK,
  RFC_WTR2965_NON_CA2_4320_SIM_0_RFFE5_DATA,
  RFC_WTR2965_NON_CA2_4320_SIM_0_RFFE1_CLK,
  RFC_WTR2965_NON_CA2_4320_SIM_0_RFFE1_DATA,
  RFC_WTR2965_NON_CA2_4320_SIM_0_RFFE2_CLK,
  RFC_WTR2965_NON_CA2_4320_SIM_0_RFFE2_DATA,
  RFC_WTR2965_NON_CA2_4320_SIM_0_RFFE3_CLK,
  RFC_WTR2965_NON_CA2_4320_SIM_0_RFFE3_DATA,
  RFC_WTR2965_NON_CA2_4320_SIM_0_INTERNAL_GNSS_BLANK,
  RFC_WTR2965_NON_CA2_4320_SIM_0_INTERNAL_GNSS_BLANK_CONCURRENCY,
  RFC_WTR2965_NON_CA2_4320_SIM_0_TX_GTR_TH,
#ifdef FEATURE_HW_LGA_30P30
  /*GPIO51 was used for DCD in the 30*30 PCB*/
#else
  RFC_WTR2965_NON_CA2_4320_SIM_0_PA_IND,
#endif
#ifdef FEATURE_HW_LGA_30P30
  RFC_WTR2965_NON_CA2_4320_SIM_0_RF_PATH_SEL_02, //GPIO45
  RFC_WTR2965_NON_CA2_4320_SIM_0_RF_PATH_SEL_15,//GPIO49
#endif
  RFC_WTR2965_NON_CA2_4320_SIM_0_SIG_NUM,
  RFC_WTR2965_NON_CA2_4320_SIM_0_SIG_INVALID,
}wtr2965_non_ca2_4320_sim_0_sig_type;
```

rf_card_wtr2965_non_ca2_4320_sim_0_rx0_wcdma_b1_sig_cfg.cfg_sig_list[0].signame之所以选择，RFC_WTR2965_NON_CA2_4320_SIM_0_RF_PATH_SEL_04的原因倒查如下：

1. 在文件 `modem_proc/rfc_jolokia/rf_card/rfc_wtr2965_non_ca2_4320_sim_0/common/src/rfc_wtr2965_non_ca2_4320_sim_0_cmn_ag.cpp` 中，



注意:

在rfc_wtr2965_non_ca2_4320_sim_0_sig_info[RFC_WTR2965_NON_CA2_4320_SIM_0_SIG_NUM + 1]列表中RFC_MSM_RF_PATH_SEL_04排序是第18位，而在wtr2965_non_ca2_4320_sim_0_sig_type中RFC_WTR2965_NON_CA2_4320_SIM_0_RF_PATH_SEL_04也是第18位，这样RFC_MSM_RF_PATH_SEL_04和RFC_WTR2965_NON_CA2_4320_SIM_0_RF_PATH_SEL_04就对应起来了。

2. 在文件`modem_proc/rfc_jolokia/target/mdm9607/src/rfc_msm_signal_info_ag.c`中

```
rfc_msm_signal_info_type rfc_mdm9607_signal_info[RFC_MSM_SIG_NUM] =
{
......
    {  RFC_ANT_SEL, 47, 4, RFC_GRFC, 1, DAL_GPIO_OUTPUT, "grfc[4]"},  /* Signal:
RFC_MSM_RF_PATH_SEL_04, MSM Pin Name: GPIO_47*/
......
};
```

注意，在 rfc_mdm9607_signal_info[RFC_MSM_SIG_NUM]列表中，grfc number是4，而GPIO口对应的是39口。

rfc_msm_signal_info_type的定义如下:

```
typedef struct
{
  rfc_signal_type signal_type;
  uint32 msm_gpio;
  uint8 grfc_num;
  rfc_gpio_grfc_type output_type;
  uint8 function_select;
  DALGpioDirectionType direction;
  char* tlmm_gpio_name;
} rfc_msm_signal_info_type;
```

即GPIO39在硬件设计中是配置为RFC_WTR2965_NON_CA2_4320_SIM_0_RF_PATH_SEL_04的。

2、rf_card_wtr2965_non_ca2_4320_sim_0_tx0_wcdma_b1_sig_cfg

rf_card_wtr2965_non_ca2_4320_sim_0_tx0_wcdma_b1_sig_cfg配置如下：

```
rfc_sig_info_type rf_card_wtr2965_non_ca2_4320_sim_0_tx0_wcdma_b1_sig_cfg =
{
  RFC_ENCODED_REVISION,
  {
    { (int)RFC_WTR2965_NON_CA2_4320_SIM_0_RF_PATH_SEL_17,   { RFC_LOW,
0/*Warning: Not specified*/ }, {RFC_LOW, 0/*Warning: Not specified*/ }  },
    { (int)RFC_WTR2965_NON_CA2_4320_SIM_0_TX_GTR_TH,   { RFC_CONFIG_ONLY,
0/*Warning: Not specified*/ }, {RFC_LOW, 0/*Warning: Not specified*/ }  },
    { (int)RFC_SIG_LIST_END,   { RFC_LOW, 0 }, {RFC_LOW, 0 } }
  },
};
```

rfc_sig_info_type的定义已经做过介绍。

5.2 获取device config data

device config data通过函数 `devices_cfg_data_get()` 获取。该函数所在路径为：

modem_proc/rfc_jolokia/rf_card/rfc_wtr2965_non_ca2_4320_sim_0/wcdma/src/rfc_wtr2965_non_ca2_4320_sim_0_wcdma_config_ag.cpp

定义如下：

```
boolean rfc_wtr2965_non_ca2_4320_sim_0_wcdma_ag::devices_cfg_data_get(
rfc_cfg_params_type *cfg, rfc_device_info_type **ptr )
{
......
  if ( ( cfg->rx_tx == RFC_CONFIG_RX ) && ( cfg->logical_device == RFM_DEVICE_0
) && ( cfg->alternate_path == 0 /*Warning: not specified*/ ) && ( cfg->band ==
(int)RFCOM_BAND_IMT ) && ( cfg->req == RFC_REQ_DEFAULT_GET_DATA ) && !ret_val )
  { *ptr = &(rf_card_wtr2965_non_ca2_4320_sim_0_rx0_wcdma_b1_device_info);
 ret_val = TRUE; }

  if ( ( cfg->rx_tx == RFC_CONFIG_RX ) && ( cfg->logical_device == RFM_DEVICE_1
) && ( cfg->alternate_path == 0 /*Warning: not specified*/ ) && ( cfg->band ==
(int)RFCOM_BAND_IMT ) && ( cfg->req == RFC_REQ_DEFAULT_GET_DATA ) && !ret_val )
  { *ptr = &(rf_card_wtr2965_non_ca2_4320_sim_0_rx1_wcdma_b1_device_info);
 ret_val = TRUE; }

  if ( ( cfg->rx_tx == RFC_CONFIG_TX ) && ( cfg->logical_device == RFM_DEVICE_0
) && ( cfg->alternate_path == 0 /*Warning: not specified*/ ) && ( cfg->band ==
(int)RFCOM_BAND_IMT ) && ( cfg->req == RFC_REQ_DEFAULT_GET_DATA ) && !ret_val )
  { *ptr = &(rf_card_wtr2965_non_ca2_4320_sim_0_tx0_wcdma_b1_device_info);
 ret_val = TRUE; }
......
  return ret_val;
}
```

以rf_card_wtr2965_non_ca2_4320_sim_0_rx0_wcdma_b1_device_info和rf_card_wtr2965_non_ca2_4320_sim_0_tx0_wcdma_b1_device_info为例，介绍device config data的配置。

1、rf_card_wtr2965_non_ca2_4320_sim_0_rx0_wcdma_b1_device_info

rf_card_wtr2965_non_ca2_4320_sim_0_rx0_wcdma_b1_device_info配置如下：

```
rfc_device_info_type rf_card_wtr2965_non_ca2_4320_sim_0_rx0_wcdma_b1_device_info
=
{
  RFC_ENCODED_REVISION,
  RFC_RX_MODEM_CHAIN_0,   /* Modem Chain */
  0,   /* NV Container */
  0,   /* Antenna */
  2,  /* NUM_DEVICES_TO_CONFIGURE 需要配置的device数量 */
  {
    {
      RFDEVICE_TRANSCEIVER, // RF设备类型
      WTR2965, /* NAME */  // RF设备名称
      0,   /* DEVICE_MODULE_TYPE_INSTANCE */
      0,   /* PHY_PATH_NUM */
      {
        0 /* Warning: Not specified */,   /* INTF_REV */
        (int)WTR2965_WCDMA_PRXLGY1_BAND1_PMB3,   /* PORT 端口 */
        ( RFDEVICE_PA_LUT_MAPPING_INVALID ),   /* RF_ASIC_BAND_AGC_LUT_MAPPING
*/
        FALSE,   /* TXAGC_LUT */
        WTR2965_FBRX_ATTN_DEFAULT,   /* FBRX_ATTN_STATE */
        0,  /* Array Filler */
      },
    },
    {
      RFDEVICE_ASM,
      GEN_ASM /*sky77916_ASM*/, /* NAME */
      0,  /* DEVICE_MODULE_TYPE_INSTANCE */
      0 /*Warning: Not specified*/,  /* PHY_PATH_NUM */
      {
        0  /* Orig setting:  */,  /* INTF_REV */
        (0x01A5 << 22)/*mfg_id*/ | (0x96 << 14)/*prd_id*/ | (6)/*port_num*/,  /*
PORT_NUM */
        (0x02E9 << 22)/*mfg_id*/ | (0x29 << 14)/*prd_id*/ | (6)/*port_num*/,  /*
PORT_NUM */
        0,  /* Array Filler */
        0,  /* Array Filler */
        0,  /* Array Filler */
      },
    },
  },
};
```

rfc_device_info_type定义如下：

```
typedef struct
{
```

```
  /*32 bit element capturing the RFC revision:
    upper 8 bits:  Branch/PL revision
    next 8 bits:   Major revision: This gets updated when there is
                   a change to GPIO/GRFC mapping information, that
                   could impact all RF Cards. A major revision
                   update triggers release for all RF cards.
    lower 16 bits: Minor revision: Any changes specific to certain
                   RF cards only, such as signal logic or device
                   configurations. Minor revision update only
                   mandates release of affected RF cards. */
  uint32 rfc_revision;

  /* Modem Chain is specified in ag files per
     logical path (RFM device) and band.
     For Rx configuration, this represents the ADC/WB chain to be used.
     For Tx configuration, this represents the DAC/TXC/TXR chain to be used.
     This information is required to be band specific as some cards
     split bands across transceivers: All low bands on one TRx, which
     is hardwired to a certain ADC/DAC chain and all high bands on
     the other TRx, which is hardwired to the other ADC/DAC chain */
  uint32 modem_chain;

  /* This captures which NV container to derive calibrated data from.
     Multiple logical paths (RFM devices) which share the same RF path
     will share the same NV container. */
  uint32 nv_container;

  /* Antenna number */
  uint32 ant_num;

  /* Number of physical devices, such as PAs, Antenna Switch modules
     and transceivers */
  uint32 num_devices;

  /* Configuration information for each device, such as port info */
  rfc_asic_info_type rf_asic_info[];
} rfc_device_info_type;
```

在rf_card_wtr2965_non_ca2_4320_sim_0_rx0_wcdma_b1_device_info中选择，WTR2965_WCDMA_PRXLGY1_BAND1_PMB3的依据：硬件设计。

1. NUM_DEVICES_TO_CONFIGURE

需要设置的设备数量，根据实际用到的设备的数量来配置。该值与下面的设备数目保持一致。例如在

rf_card_wtr2965_non_ca2_4320_sim_0_rx0_wcdma_b1_device_info中，该值为2，是因为下面有2个设备：

- RFDEVICE_TRANSCEIVER
- RFDEVICE_ASM

2. port

在rf_card_wtr2965_non_ca2_4320_sim_0_rx0_wcdma_b1_device_info中选择，WTR2965_WCDMA_PRXLGY1_BAND1_PMB3由硬件设计决定

2、rf_card_wtr2965_non_ca2_4320_sim_0_tx0_wcdma_b1_device_info

rf_card_wtr2965_non_ca2_4320_sim_0_tx0_wcdma_b1_device_info配置如下：

```
rfc_device_info_type rf_card_wtr2965_non_ca2_4320_sim_0_tx0_wcdma_b1_device_info
=
{
  RFC_ENCODED_REVISION,
  RFC_TX_MODEM_CHAIN_0,   /* Modem Chain */
  0,   /* NV Container */
  0,   /* Antenna */
  5,  /* NUM_DEVICES_TO_CONFIGURE */
  {
    {
      RFDEVICE_TRANSCEIVER,
      WTR2965, /* NAME */
      0,   /* DEVICE_MODULE_TYPE_INSTANCE */
      0,   /* PHY_PATH_NUM */
      {
        0 /* Warning: Not specified */,   /* INTF_REV */
        (int)WTR2965_WCDMA_TX_BAND1_THMLB4,   /* PORT */
        ( RFDEVICE_PA_LUT_MAPPING_VALID | WTR2965_LP_LUT_TYPE <<
RFDEVICE_PA_STATE_0_BSHFT | WTR2965_HP_LUT_TYPE << RFDEVICE_PA_STATE_1_BSHFT |
WTR2965_HP_LUT_TYPE << RFDEVICE_PA_STATE_2_BSHFT | WTR2965_HP_LUT_TYPE <<
RFDEVICE_PA_STATE_3_BSHFT  ),   /* RF_ASIC_BAND_AGC_LUT_MAPPING */
        FALSE,   /* TXAGC_LUT */
        WTR2965_FBRX_LOW_ATTN_MODE,   /* FBRX_ATTN_STATE */
        0,  /* Array Filler */
      },
    },
    {
      RFDEVICE_ASM,
      GEN_ASM /*sky77916_ASM_with_gsm_pa*/, /* NAME */
      0,  /* DEVICE_MODULE_TYPE_INSTANCE */
      0 /*Warning: Not specified*/,  /* PHY_PATH_NUM */
      {
        0  /* Orig setting:  */,  /* INTF_REV */
        (0x1A5 << 22)/*mfg_id*/ | (0x96 << 14)/*prd_id*/ | (15)/*port_num(11)*/,
 /* PORT_NUM */
        (0x02E9 << 22)/*mfg_id*/ | (0x29 << 14)/*prd_id*/ | (15)/*port_num*/,
 /* PORT_NUM */
        0,  /* Array Filler */
        0,  /* Array Filler */
        0,  /* Array Filler */
      },
    },
    {
      RFDEVICE_PA,
      GEN_PA /*sky77638_PA*/, /* NAME */
      0,  /* DEVICE_MODULE_TYPE_INSTANCE */
      0 /*Warning: Not specified*/,  /* PHY_PATH_NUM */
      {
        0  /* Orig setting:  */,  /* INTF_REV */
        (0x1A5 << 22)/*mfg_id*/ | (0x1C << 14)/*prd_id*/ | (33)/*port_num 0*/,
 /* PORT_NUM */
```

```
        (0x02E9 << 22)/*mfg_id*/ | (0x89 << 14)/*prd_id*/ | (0)/*port_num*/,  /*
PORT_NUM */
        (0x2E9 << 22)/*mfg_id*/ | (0x8A << 14)/*prd_id*/ | (0),  /* PORT_NUM */
        0,  /* Array Filler */
        0,  /* Array Filler */
      },
    },
    {
      RFDEVICE_ASM,
      GEN_ASM /*SKY77638_ASM*/, /* NAME */
      1,  /* DEVICE_MODULE_TYPE_INSTANCE */
      0 /*Warning: Not specified*/,  /* PHY_PATH_NUM */
      {
        0  /* Orig setting:  */,  /* INTF_REV */
        (0x1A5 << 22)/*mfg_id*/ | (0x1C << 14)/*prd_id*/ | (4),  /* PORT_NUM */
        (0x02E9 << 22)/*mfg_id*/ | (0x0089 << 14)/*prd_id*/ | (4)/*port_num*/,
 /* PORT_NUM */
        (0x02E9 << 22)/*mfg_id*/ | (0x008A << 14)/*prd_id*/ | (4)/*port_num*/,
 /* PORT_NUM */
        0,  /* Array Filler */
        0,  /* Array Filler */
      },
    },
    {
      RFDEVICE_HDET,
      TRX_HDET,  /* NAME */
      0,  /* DEVICE_MODULE_TYPE_INSTANCE */
      0 /*Warning: Not specified*/,  /* PHY_PATH_NUM */
      {
        0  /* Orig setting:  */,  /* INTF_REV */
        0,  /* Array Filler */
        0,  /* Array Filler */
        0,  /* Array Filler */
        0,  /* Array Filler */
        0,  /* Array Filler */
      },
    },
  },
};
```