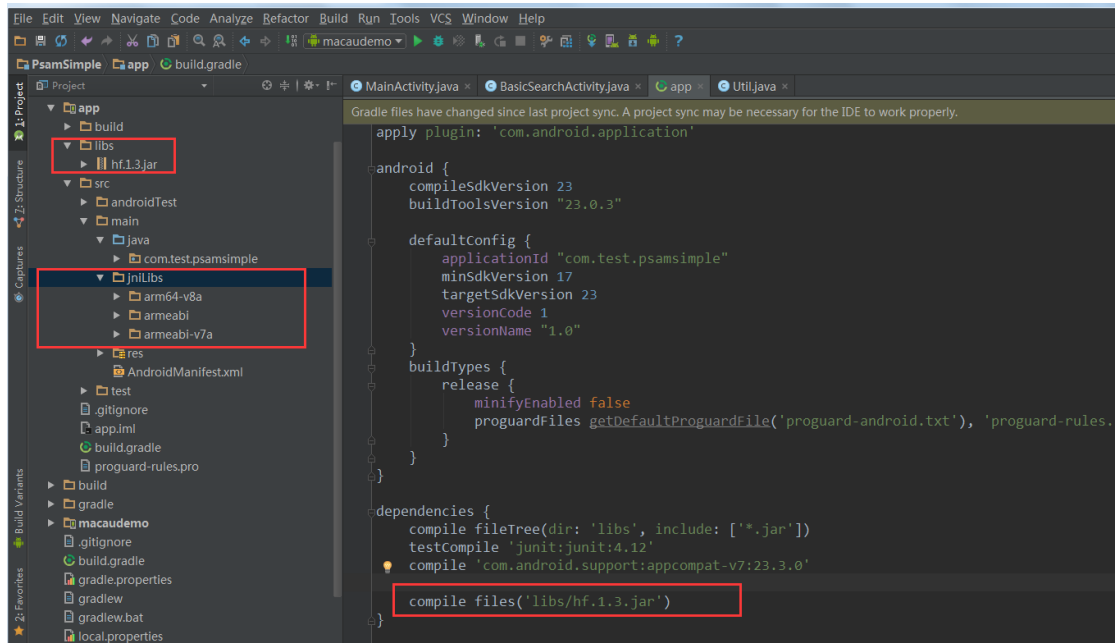


# PSAM RFID API Instruction

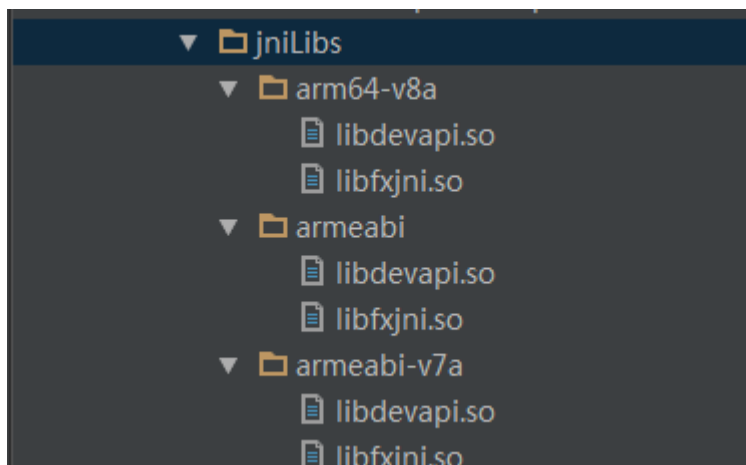
## 1. Import so library and jar

hf.1.3.jar as the java interface based on libdevapi.so and libfxjni.so, arm64-v8a, armeabi, armeabi-v7a can be selected based on the project requirements(one or all can be selected based on the requirements)



## 2 Function Interface

Function interfaces are packaged in RfidNative.class of jar, and functional callings based on the module are in libdevapi.so and libfxjni.so.



## 2.01 RfidNative ()

|             |                     |
|-------------|---------------------|
| Function    | RfidNative ()       |
| Description | Constructor         |
| Parameter   | None                |
| Return      | RfidNative Instance |

## 2.02 open(int serialPort, int baudrate)

|             |   |
|-------------|---|
| Function    | <b>public int open(int serialPort, int baudrate)</b>  |
| Description | Open PSAM RFID module and init  |
| Parameter   | <b>int</b> serialPort, , serial port, 14 by default<br><b>int</b> baudrate, baudrate, 115200 by default |
| Return      | Int value large than 0 for success, -1 for failure  |

## 2.03 close(int serialPort)

|             |  |
|-------------|--|
| Function    | public int close(int serialPort)                   |
| Description | Close the hardware module                          |
| Parameter   | <b>int</b> serialPort, serial port, 14 by default  |
| Return      | Int value large than 0 for success, -1 for failure |

## 2.04 psamreset(int psamCard,byte[] result)

|             |   |
|-------------|---|
| Function    | public int psamreset(int psamCard,byte[] result)                            |
| Description | Psam reset, cold reset  |
| Parameter   | int psamCard psam slot number SAM1 = 1, SAM2 = 2<br>byte[] result c         |
| Return      | Int value large than 0 for success as the result data length, -1 as failure |

## 2.05 psamclose(int psamCard)

|             |  |
|-------------|--|
| Function    | public int psamclose(int psamCard)                 |
| Description | Close Psam   |
| Parameter   | int psamCard psam slot number SAM1 = 1, SAM2 = 2   |
| Return      | Int value large than 0 for success, -1 for failure |

## 2.06 psamapdu(int psamCard, byte[] apduCmd, byte[] result)

|             |   |
|-------------|---|
| Function    | public int psamapdu(int psamCard, byte[] apduCmd, byte[] result)  |
| Description | Execute APDU commands   |
| Parameter   | int psamCard psam slot number SAM1 = 1, SAM2 = 2<br>byte[] apduCmd APDU commands<br>byte[] result Returned data of APDU |
| Return      | Int value large than 0 for success as the result data length, -1 for failure  |

## 2.07 psamhotreset(int psamCard,byte[] result)

|             |   |
|-------------|---|
| Function    | <b>public int psamhotreset(int psamCard,byte[] result)</b>  |
| Description | Psam reset, hot reset   |
| Parameter   | int psamCard psam slot number SAM1 = 1, SAM2 = 2<br>byte[] result returned data after being reset |
| Return      | Int value larger than 0 for success as result data length, -1 for failure                         |

## 2.08 readreg(int regAddress, byte[] result)

|             |   |
|-------------|---|
| Function    | <b>public int readreg(int regAddress, byte[] result)</b>                      |
| Description | Read register value   |
| Parameter   | <b>int</b> regAddress, reg address<br><b>byte[]</b> result content of the reg |
| Return      | Int value larger than 0 for success as result data length, -1 for failure     |

## 2.09 writereg(int regAddress, byte[] value, byte[]result)

|             |  |
|-------------|--|
| Function    | public int writereg(int regAddress, byte[] value, byte[]result)  |
| Description | Constructor  |
| Parameter   | <b>int</b> regAddress, register address<br>byte[] value write value<br><b>byte[]</b> result returned content |
| Return      | Int value larger than 0 for success as result data length, -1 for failure                                    |

## 2.10 findM1(byte[] uid)

|             |                                  |
|-------------|----------------------------------|
| Function    | public int findM1(byte[] uid)    |
| Description | Find M1card(ISO 14443A protocol) |
| Parameter   | <b>byte[]</b> uid UID and Return |

|        |  |
|--------|--|
| Return | Int value larger than 0 for success as uid data length, -1 for failure |
|--------|--|

## 2.11 authM1Card(int keyType, int sector, byte[] keys)

|             |   |
|-------------|---|
| Function    | public int authM1Card(int keyType, int sector, byte[] keys)   |
| Description | M1 card authorize   |
| Parameter   | int keyType, keytype, 0 for KeyA ,1 for KeyB<br>int sector, sector, sector*4, like 03 sector = 3*4 = 12<br>byte[] keys 6 bytes key, key of white card is 0xffffffff |
| Return      | Int value large than 0 for success, others for failure  |

## 2.12 authM1SHCard(int keyType, int sector, byte[] keys)

|             |   |
|-------------|---|
| Function    | public int authM1SHCard(int keyType, int sector, byte[] keys)   |
| Description | FM serial card authorize, like FM1208   |
| Parameter   | int keyType, keytype, 0 for KeyA ,1 for KeyB<br>int sector, sector*4, like 03 sector = 3*4 = 12<br>byte[] keys 6 bytes key, key of white card is 0xffffffff |
| Return      | Int value large than 0 for success, others for failure  |

## 2.13 readM1(int block, byte[] readData)

|             |   |
|-------------|---|
| Function    | public int readM1(int block, byte[] readData)   |
| Description | Read M1 card block data   |
| Parameter   | int block, block number, block = sector*4 + (0 or 1 or 2 or 3)<br>byte[] readDataReturn |
| Return      | >0 for success as readData length, others for failure                                   |

## 2.14 writeM1(int block , byte[] writeData)

|             |  |
|-------------|--|
| Function    | public int writeM1(int block , byte[] writeData)                                   |
| Description | Write M1 card block data   |
| Parameter   | int block, block number, block = sector*4 + (0 or 1 or 2 or 3)<br>byte[] writeData |
| Return      | >0 for success, others for failure   |

## 2.15 initValM1(int block, byte[] value)

|          |   |
|----------|---|
| Function | public int initValM1(int block, byte[] value) |
|----------|---|

|             |   |
|-------------|---|
| Description | Init M1 value, as e-wallet  |
| Parameter   | int block, block number, like sector 02 block 0, 0x08<br>byte[] value , init value, 100(0x00000064) |
| Return      | >0 for success, others for failure  |

## 2.16 incrementM1 (int block , byte[] value)

|             |   |
|-------------|---|
| Function    | public int incrementM1(int block , byte[] value)  |
| Description | Increase the value, as e-wallet(please call initValM1 to init if it is not initialized)             |
| Parameter   | int block, block number, like sector 02 block 0, 0x08<br>byte[] value increase, like 1 (0x00000001) |
| Return      | >0 for success, others for failure  |

## 2.17 decrementM1(int block, byte[] value)

|             |   |
|-------------|---|
| Function    | public int decrementM1(int block, byte[] value)   |
| Description | Decrease the value, as e-wallet, (please call initValM1 to init if it is not initialized)           |
| Parameter   | int block, block number, like sector 02 block 0, 0x08<br>byte[] value decrease, like 1 (0x00000001) |
| Return      | >0 for success, others for failure  |

## 2.18 restoreM1(int block)

|             |  |
|-------------|--|
| Function    | public int restoreM1(int block)  |
| Description | Restore, as e-wallet(please call initValM1 to init if it is not initialized) |
| Parameter   | int block, block number, like sector 02 block 0, 0x08                        |
| Return      | >0 for success, others for failure   |

## 2.19 transferM1(int block)

|             |  |
|-------------|--|
| Function    | public int transferM1(int block)   |
| Description | Transfer as e-wallet(please call initValM1 to init if it is not initialized) |
| Parameter   | int block, block number, like sector 02 block 0, 0x08                        |
| Return      | >0 for success, others for failure   |

## 2.20 haltM1 ()

|             |                     |
|-------------|---------------------|
| Function    | public int haltM1() |
| Description | Halt status         |

|           |                                    |
|-----------|------------------------------------|
| Parameter | None                               |
| Return    | >0 for success, others for failure |

## 2.21 readM1Val(int block, byte[] value)

|             |  |
|-------------|--|
| Function    | public int readM1Val(int block, byte[] value)                                |
| Description | Read value   |
| Parameter   | int block, block number, like sector 02 block 0, 0x08<br>byte[] value Return |
| Return      | >0 for success as value length, others for failure                           |

## 2.22 activeUcpu(byte[] result)

|             |   |
|-------------|---|
| Function    | public int activeUcpu(byte[] result)                |
| Description | Active contactless CPU                              |
| Parameter   | <b>byte[]</b> result Returned content               |
| Return      | >0 for success as result length, others for failure |

## 2.23 apduCpu(byte[] cos, byte[] result)

|             |  |
|-------------|--|
| Function    | public int apduCpu(byte[] cos, byte[] result)              |
| Description | Send COS command   |
| Parameter   | byte[] cos, APDU command<br>byte[] result returned content |
| Return      | >0 for success as result length, others for failure        |

## 2.24 closeUCPU()

|             |                                    |
|-------------|------------------------------------|
| Function    | public int closeUCPU()             |
| Description | Close CPU                          |
| Parameter   | None                               |
| Return      | >0 for success, others for failure |

## 2.25 getSAK(byte[] result)

|             |   |
|-------------|---|
| Function    | public int getSAK(byte[] result)  |
| Description | Get SAK (ISO 14443A Protocol)   |
| Parameter   | byte[] result Result, the ahead n-1 bytes as UID, the last byte as SAK, like Mifare S50 card 0x08 |

|        |   |
|--------|---|
| Return | >0 for success as result length, others for failure |
|--------|---|

## 2.26 find15693Card(byte[] uid)

|             |  |
|-------------|--|
| Function    | public int find15693Card(byte[] uid)         |
| Description | Find 15693 UID                               |
| Parameter   | <b>byte[]</b> uid ,                          |
| Return      | >0 for success as UID length, -1 for failure |

## 2.27 select15693Card(byte[] uid)

|             |  |
|-------------|--|
| Function    | public int select15693Card(byte[] uid) |
| Description | Select 15693 UID                       |
| Parameter   | <b>byte[]</b> uid                      |
| Return      | >0 for success, others for failure     |

## 2.28 read15693block(int startAddr, int block , byte[] result)

|             |  |
|-------------|--|
| Function    | public int read15693block(int startAddr, int block , byte[] result)                      |
| Description | Read 15693 block data  |
| Parameter   | int startAddr, starting address<br>int block , block numbers to be read<br>byte[] result |
| Return      | >1 for success, others for failure   |

## 2.29 write15693block(int startAddr, int block , byte[] data)

|             |  |
|-------------|--|
| Function    | public int write15693block(int startAddr, int block , byte[] data)   |
| Description | Write 15693 block data, and single block only each time  |
| Parameter   | int startAddr, starting address, starting block<br>int block, block number, 0 or 1<br>byte[]data, data to be written |
| Return      | >0 writing successfully, others indicate failure   |