PSAM Integration Instruction Documentation

Version	Description	Date	Written By
V1.0	Added PSAM Control Integration Instruction	2023/2/4	郑俊科
V1.1	Add Model:Falcon 1 、Swift 2 Pro、Lark 1	2023/4/8	郑俊科

1.Introduction

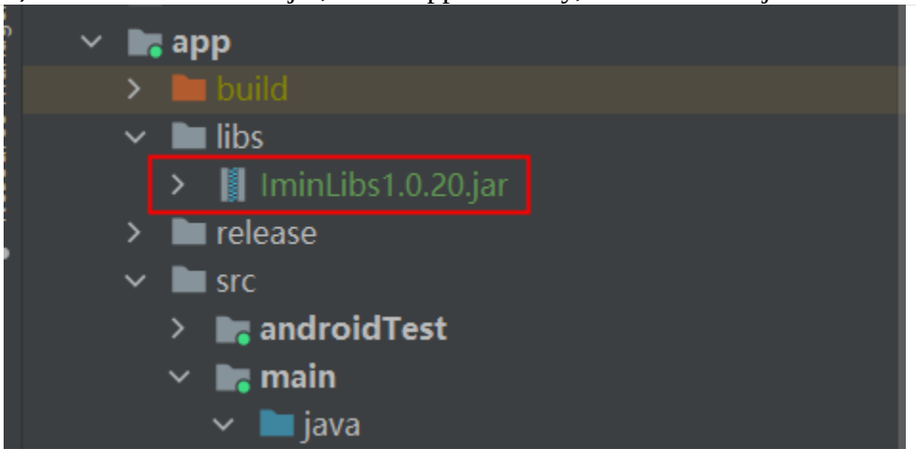
This document provides guidance for developers on how to integrate to retrieve data from PSAM.

Model

No	Model
1	Swift 2
2	Falcon 1 、Swift 2 Pro、Lark 1

2.Integration Procedure

1) Reference IminLib.jar, under app directory, add IminLibs.jar to the libs folder



2) Add IminLib.jar dependencies， under app directory, add dependencies to build.gradle.

```
dependencies {
    implementation files('libs/IminLibs1.0.20.jar')
    implementation 'androidx.appcompat:appcompat:1.6.1'
```

3) Request to retrieve the PSAM status (Prerequisite: PSAM card has been inserted).

```
public void psamTest(View view) {
        Toast.makeText(this, "正在读取...", Toast.LENGTH_SHORT).show();
          new Thread(){
            @Override
            public void run() {
                super.run();

                boolean isSuccess = false;
                String readResult = "";
                for (int i = 0; i < IminPsamService.SLOT_ARRAY.length; i++) {
                    for (int j = 0; j < IminPsamService.MODE_ARRAY.length; j++) {
                        Log.d(TAG, "slot: " + IminPsamService.SLOT_ARRAY[i] + ", mode: " +
IminPsamService.MODE_ARRAY[j]);
                        readResult = readPsam(IminPsamService.SLOT_ARRAY[i], IminPsamService.MODE_ARRAY[j], false);
                        Log.d(TAG, "readResult: " + readResult);
                        if (readResult.startsWith("ICC 检查成功")) {
                            isSuccess = true;
                            break;
                        }
                    }
                    if (isSuccess) {
                        break;
                    }
                }
                Log.d(TAG, "isSuccess: " + isSuccess);
                if (!isSuccess) {
                    try {
                        sleep(2000);
                    } catch (Exception e2) {
                        e2.printStackTrace();
                    }
                    Log.d(TAG, "read SuperHighCard: ");
                    readResult = readPsam(IminPsamService.NORMAL_SLOT, IminPsamService.VCC_5V_MODE, true);
                    if (!readResult.startsWith("ICC 检查成功")) {
                        try {
                            sleep(1000);
                        } catch (Exception e2) {
                            e2.printStackTrace();
                        }
                        readResult = readPsam(IminPsamService.NORMAL_SLOT, IminPsamService.VCC_5V_MODE, true);
                    }
                    }
                Message msg = new Message();
                msg.obj = readResult;
```

```
                msg.what = 2;
                mHandler.sendMessage(msg);
            }
        }.start();

    }

    private String readPsam(byte slot, byte mode, boolean isSuperHighCard) {
        byte[] atr = new byte[40];
        byte[] apduSend = new byte[600];
        byte[] apduRecv = new byte[600];
          try {
            int iRet;
            IminSDKManager.iccDevParaSet(MainActivity.this, slot, (byte)0, (byte)0, (byte)0);
            if (isSuperHighCard) {
                    int devParaSet = IminSDKManager.iccDevParaSet(MainActivity.this, (byte)0x01, (byte)1, (byte)1,
(byte) 12);
                Log.e(TAG, "devParaSet: " + devParaSet);
                iRet = IminSDKManager.openPsam(MainActivity.this, (byte)0x01, (byte)1, atr);
                if (iRet != 0) {
                    return "ICC 检查失败, result:" + iRet;
                }
            } else {
                iRet = IminSDKManager.openPsam(MainActivity.this, slot, mode, atr);
                if (iRet != 0) {
                    return "ICC 检查失败, result:" + iRet;
                }
            }
            //if(true) return ;
            apduSend[0] = (byte) 0x00;
            apduSend[1] = (byte) 0xa4;
            apduSend[2] = (byte) 0x04;
            apduSend[3] = (byte) 0x00;
            apduSend[4] = (byte) 0x00;
            apduSend[5] = (byte) 0x0e;

            System.arraycopy("1PAY.SYS.DDF01".getBytes(), 0, apduSend, 6, 14);
            apduSend[6 + 14] = (byte) 0x01;
            apduSend[7 + 14] = (byte) 0x00;


            iRet = IminSDKManager.commandPsam(MainActivity.this, IminPsamService.FAST_SLOT, apduSend, apduRecv);
            if (iRet == 0) {
                int lenout = apduRecv[0] * 256 + apduRecv[1];
                byte[] dataout = new byte[512];
                System.arraycopy(apduRecv, 2, dataout, 0, lenout);
                String dataOut = ByteUtil.bytearrayToHexString(dataout, lenout);
                return "ICC 检查成功:" + dataOut;
            } else {
//                Toast.makeText(this, "ICC 检查失败, result:" + iRet, Toast.LENGTH_SHORT).show();
                return "ICC 检查失败, result:" + iRet;
            }
```

```
        } finally {
            IminSDKManager.closePsam(MainActivity.this, IminPsamService.FAST_SLOT);
        }
    }
```

4) The source code above must be added in a sub-thread for reading. For the situation where iRet does not return 0, you can refer to the corresponding return values in the Android Psam Application Development Guide.