

ID INFO 9000 API documentation

Date: 12.08.2025



Please read and follow the installation instructions before operating the device and keep the instructions for future reference or for use after troubleshooting

Table of Contents

1. Interface Document Maintenance Record	3
2. Function Description	3
2.1 Camera	3
2.1.1 RGB camera call.....	3
2.1.2 NIR camera call	3
2.2 Fill Light	3
2.2.1 Open	3
2.2.2 Closure	4
2.3 Relay.....	4
2.3.1 Initialization (restoring the initial state)	4
2.3.2 Open	4
2.3.3 Close	5
2.4 OTG HOST mode switch.....	5
2.4.1 Enable OTG	5
2.4.2 Enable HOST	5
2.4 OTG HOST mode switch.....	6
2.4.1 Enable OTG	6
2.4.2 Enable HOST	6
2.5 Temperature.....	6
2.5.1 Get temperature.....	6
2.6 Scan code and 232, 485 serial ports	7
2.6.1 Read and write 232 and 486.....	7
2.7 Card Reading	7

1. Interface Document Maintenance Record

Record:

Modify and update records		
Version	Changes	Time
V1.0.0	First edition	20twenty three/04/20

2. Function Description

2.1 Camera

2.1.1 RGB Camera

Function prototype: Camera.open(0);

Parameter: int

Note: cameraId 0 is an RGB camera

2.1.2 NIR camera call

Function prototype: Camera.open(1);

Parameter: int

Note: cameraId 1 is a NIR camera

2.2 Fill Light

2.2.1 Open

Function prototype:

```
FileOutputStream outPutStream = newFileOutputStream("/sys/class/leds/pwm-flash/brightness");
byte[] LIGHT_OPEN = {'5','5'};
outPutStream.write(LIGHT_OPEN);
outPutStream.close();
```

Illustrate:

```
Fill light file address: "/sys/class/leds/pwm-flash/brightness";
Brightness value 0 - 255:byte[] LIGHT_OPEN= {'5', '5'};This example value is 55;
```

2.2.2 Closure

Function prototype:

```
FileOutputStream outPutStream = new FileOutputStream("/sys/class/leds/pwm-flash/brightness");
byte[] LIGHT_CLOSE = {'0'};
outPutStream.write(LIGHT_CLOSE);
outPutStream.close();
```

Illustrate:

```
Fill light file address: "/sys/class/leds/pwm-flash/brightness";
Brightness value 0 - 255: byte[] LIGHT_CLOSE= {'0'}; This example value is 0, 0 is off;
```

2.3 Relay

2.3.1 Initialization (restoring the initial state)

Function prototype:

```
FileOutputStream outPutStream = new FileOutputStream("/sys/class/gpio/gpio114/direction");
byte[] OPEN_OUT = {'o','u','t'};
outPutStream.write(OPEN_OUT);
outPutStream.close();
```

Illustrate:

```
gpioaddress: "/sys/class/gpio/gpio114/direction";
Set the mode to output: byte[] openOut = {'o','u','t'}
Note: If you use the relay function, you must call this method once in the APP to set the mode to output;
just call it once
```

2.3.2 Open

Function prototype:

```
FileOutputStream outPutStream = new FileOutputStream("/sys/class/gpio/gpio114/value");
byte[] OPEN_DOOR = {'1'};
outPutStream.write(OPEN_DOOR);
outPutStream.close();
```

Illustrate:

```
gpioaddress: "/sys/class/gpio/gpio114/value";
A value of 1: byte[] OPEN_DOOR = {'1'}
```

2.3.3 Close

Function prototype:

```
FileOutputStream outPutStream = new FileOutputStream("/sys/class/gpio/gpio114/value");
byte[] CLOSE_DOOR= {'0'};
outPutStream.write(CLOSE_DOOR);
outPutStream.close();
```

Illustrate:

```
gpioaddress:"/sys/class/gpio/gpio114/value";
Value is 0:byte[]CLOSE_DOOR= {'0'}
```

2.4 OTG HOST mode switch

2.4.1 Read the current mode

Function prototype:

```
public static String readInStream() { try { String path = "/sys/devices/platform/fe8a0000.usb2-phy/otg_mode"; File file = new File(path); FileInputStream inStream = null; inStream = new FileInputStream(file); byte[] buffer = new byte[1024]; int len = 0; ByteArrayOutputStream outStream = new ByteArrayOutputStream(); while ((len = inStream.read(buffer)) != -1) { outStream.write(buffer, 0, len); } byte[] data = outStream.toByteArray(); outStream.close(); inStream.close(); return new String(data); } catch (Exception e) { e.printStackTrace(); Log.e("readInStream", e.getMessage()); } return "";} 
```

Illustrate:

```
Read mode address:String path = "/sys/devices/platform/fe8a0000.usb2-phy/otg_mode";
If the String value is otg, it is in otg mode:return new String(data);
```

2.4.2 Enable OTG

Function prototype:

```
public void otg() { byte[] otg = {'o', 't', 'g'}; FileOutputStream fBlue = null; try { fBlue = new FileOutputStream("/sys/devices/platform/fe8a0000.usb2-phy /otg_mode"); fBlue.write(otg); fBlue.close(); } catch (Exception e) { e.printStackTrace(); } }
```

Illustrate:

```
Write mode address:String path = "/sys/devices/platform/fe8a0000.usb2-phy/otg_mode";
Change to otg working mode:byte[] otg = {'o', 't', 'g'};
```

2.4.3 Enable HOST

Function prototype:

```
public void host() { byte[] host = {'h', 'o', 's', 't'}; FileOutputStream fBlue = null; try { fBlue = new
FileOutputStream("/sys/devices/platform/fe8a0000.usb2-phy/otg_mode"); fBlue.write(host);
fBlue.close(); } catch (Exception e) { e.printStackTrace(); }}
```

Illustrate:

```
Write mode address:String path = "/sys/devices/platform/fe8a0000.usb2-phy/otg_mode";
Change to host working mode:byte[]host= {'h', 'o', 's','t'};
```

2.5 Temperature

2.5.1 Get Temperature

Function prototype:

1. Open the ttyS9 serial port for monitoring and writing operations

```
mSerialPortManager.openSerialPort(new File("/dev/ttyS9"), 115200);
mSerialPortManager.startReadThread(0);
mSerialPortManager.startSendThread();
```

2. Read and Write

```
boolean sendBytes = mSerialPortManager.sendBytes(Utils.hexStr2bytes("A55801FB"));
```

3. Monitor received data

```
mSerialPortManager.setOnSerialPortDataListener(new OnSerialPortDataListener() { @Override public
void onDataReceived(byte[] bytes) { setTemperature(bytes); } @Override public void onDataSent(byte[]
bytes) { }});

//Process the received data
private void setTemperature(byte[] bytes) { if (serial.length() > 100) { serial = ""; } try { String substring =
Utils.byte2hex(bytes).substring(0, 22); String substring1 = substring.substring(8, 10); String substring2 =
substring.substring(10, 12); int string = Utils.hexStringToAlgorism(substring1) ; int string1 =
Utils.hexStringToAlgorism(substring2); float temperature = (string + 256 * string1) / 100; serial +=
temperature + "\n"; runOnUiThread() -> tvSerial.setText(serial); } catch (Exception e) {
Log.e("Exception", e.getMessage()); }}
```

Process the received data. For specific code, see the demo example.

2.6 Scan code and 232, 485 serial ports

2.6.1 Read and write 232 and 485

Illustrate:

232 serial port is "/dev/ttyS5"
 The serial port of 485 is "/dev/ttyS6"
 The serial port for scanning the code is "/dev/ttyS8"

Open the corresponding serial port to read and write data. For specific examples, please see the demo.

Illustrate:

Fill light file address: "/sys/class/leds/pwm-flash/brightness";
 Brightness value 0 - 255:byte[] LIGHT_OPEN= {'5', '5'};This example value is 55;

2.7 Card Reading

Import NfcJni.java

Number	Function prototype	Parameter	Description
1	nfcJni.nfcInit();	-	Initialize NFC service
2	nfcJni.finishCard();	-	Release NFC service
3	nfcJni.getCardInfo();	-	Get the read card information
4	nfcJni.nfcSendApdu();	-	Send APDU command

6. Support

iDTRONIC

Ludwig-Reichling-Straße 4

67059 Ludwigshafen am Rhein

helpdesk@idtronic.de