

USR-EPC500 软件设计手册

文件版本：V1.0.3



目录

USR-EPC500 软件设计手册.....	1
1. 产品概述.....	6
1.1. 产品简介.....	6
1.2. 产品特点.....	6
2. 产品功能.....	7
2.1. EPC500 状态信息查看.....	7
2.1.1. 状态栏.....	7
2.1.2. 网络信息.....	8
2.2. 串口.....	9
2.3. USB 白名单功能.....	10
2.4. 网络连接.....	12
2.4.1. 指定 4G 无线网卡上网.....	12
2.4.2. 指定以太网卡一上网.....	14
2.4.3. 指定以太网卡二上网.....	16
2.4.4. 全自动联网.....	16
2.5. 路由器功能.....	17
2.5.1. 有线路由器.....	17
2.5.2. 4G 路由器.....	18
2.6. 流量控制.....	19
2.6.1. 查看网卡一使用的总流量.....	19
2.6.2. 配置流量报警.....	20
2.7. 视频.....	21
2.7.1. 调节输出分辨率.....	21
2.7.2. 视频输出选择.....	21
2.8. 音频.....	22
2.8.1. 音频输出选择.....	22
2.8.2. 音量调节.....	22
2.9. 软件升级.....	23
2.10. 修改管理员密码.....	23
2.11. 重启设备.....	24
2.12. 恢复出厂设置.....	25
2.13. 指示灯.....	25
2.14. 按键.....	26
3. 有人 SDK.....	27
3.1. 准备工作.....	27
3.1.1. 添加库文件.....	27
3.1.2. 添加 java 类.....	27
3.2. 返回值说明.....	27
3.3. 指令说明.....	28
3.3.1. 获取 Sim 卡网络制式.....	28
3.3.2. 获取运营商类型.....	28
3.3.3. 获取 4G 网卡信号强度.....	28

3.3.4.	获取当前使用的网卡.....	29
3.3.5.	获取当前 LAN 功能设备.....	29
3.3.6.	获取 IMEI.....	29
3.3.7.	获取 IMSI.....	30
3.3.8.	获取工控机联网状态.....	30
3.3.9.	获取 IP 信息.....	30
3.3.10.	获取当前子网掩码.....	31
3.3.11.	获取当前网关.....	31
3.3.12.	获取当前 DNS.....	31
3.3.13.	获取 WAN 口 MAC.....	32
3.3.14.	获取 LAN 口 MAC.....	32
3.3.15.	获取当前分辨率.....	32
3.3.16.	获取当前显示设备.....	33
3.3.17.	获取当前声卡输出.....	33
3.3.18.	设置显示设备.....	33
3.3.19.	设置分辨率.....	34
3.3.20.	设置当前声卡.....	34
3.3.21.	获取移动数据开关状态.....	34
3.3.22.	获取 APN 模式.....	35
3.3.23.	获取 4G 网卡寻网方式.....	35
3.3.24.	获取 APN 名.....	35
3.3.25.	获取 APN 卡用户名.....	35
3.3.26.	获取 APN 卡密码.....	36
3.3.27.	设置移动数据开关状态.....	36
3.3.28.	设置 APN 模式.....	37
3.3.29.	设置 4G 网卡寻网方式.....	37
3.3.30.	设置 APN 名.....	38
3.3.31.	设置 APN 卡密码.....	38
3.3.32.	设置 APN 账号.....	38
3.3.33.	获取以太网卡一开关状态.....	39
3.3.34.	获取以太网卡一模式.....	39
3.3.35.	获取以太网卡一 DHCP 开关状态.....	39
3.3.36.	获取以太网卡一静态 IP.....	40
3.3.37.	获取以太网卡一静态 MASK.....	40
3.3.38.	获取以太网卡一静态网关.....	40
3.3.39.	获取以太网卡一静态 DNS.....	40
3.3.40.	获取以太网卡一作为 LAN 口时网段.....	41
3.3.41.	获取以太网卡一 LAN 口限速开关状态.....	41
3.3.42.	获取以太网卡一 LAN 口下载速度.....	41
3.3.43.	获取以太网卡一 LAN 口上传速度.....	42
3.3.44.	获取以太网卡一优先级.....	42
3.3.45.	获取以太网卡二开关状态.....	42
3.3.46.	获取以太网卡二模式.....	43
3.3.47.	获取以太网卡二 DHCP 开关状态.....	43

3.3.48.	获取以太网卡二静态 IP	43
3.3.49.	获取以太网卡二静态 MASK	44
3.3.50.	获取以太网卡二静态网关	44
3.3.51.	获取以太网卡二静态 DNS	44
3.3.52.	获取以太网卡二作为 LAN 口时网段	45
3.3.53.	获取以太网卡二 LAN 口限速开关状态	45
3.3.54.	获取以太网卡二 LAN 口下载速度	45
3.3.55.	获取以太网卡二 LAN 口上传速度	46
3.3.56.	获取以太网卡二优先级	46
3.3.57.	获取 4G 上网卡优先级	46
3.3.58.	设置以太网卡一开关状态	47
3.3.59.	设置以太网卡一模式	47
3.3.60.	设置以太网卡一 DHCP 开关状态	48
3.3.61.	设置以太网卡一静态 IP	48
3.3.62.	设置以太网卡一静态 MASK	48
3.3.63.	设置以太网卡一静态网关	49
3.3.64.	设置以太网卡一静态 DNS	49
3.3.65.	设置以太网卡一作为 LAN 口时网段	49
3.3.66.	设置以太网卡一 LAN 口限速开关状态	50
3.3.67.	设置以太网卡一 LAN 口上传速度	50
3.3.68.	设置以太网卡一 LAN 口下载速度	50
3.3.69.	设置以太网卡一优先级	51
3.3.70.	设置以太网卡二开关状态	51
3.3.71.	设置以太网卡二模式	52
3.3.72.	设置以太网卡二 DHCP 开关状态	52
3.3.73.	设置以太网卡二静态 IP	52
3.3.74.	设置以太网卡二静态 MASK	53
3.3.75.	设置以太网卡二静态网关	53
3.3.76.	设置以太网卡二静态 DNS	53
3.3.77.	设置以太网卡二作为 LAN 口时网段	54
3.3.78.	设置以太网卡二 LAN 口限速开关状态	54
3.3.79.	设置以太网卡二 LAN 口上传速度	54
3.3.80.	设置以太网卡二 LAN 口下载速度	55
3.3.81.	设置以太网卡二优先级	55
3.3.82.	设置 4G 上网卡优先级	56
3.3.83.	获取网卡一使用流量总数	56
3.3.84.	获取流量报警阈值	56
3.3.85.	获取报警电话	57
3.3.86.	获取网卡二使用流量总数	57
3.3.87.	获取 4G 上网卡使用流量总数	57
3.3.88.	设置流量报警阈值	58
3.3.89.	设置获取报警电话	58
3.3.90.	清空设置以太网卡一流量	58
3.3.91.	清空以太网卡二流量	59

3.3.92.	清空 4G 无线网卡流量.....	59
3.3.93.	获取白名单状态开关.....	59
3.3.94.	获取白名单列表.....	60
3.3.95.	设置白名单开关状态.....	60
3.3.96.	添加一条 usb 白名单.....	60
3.3.97.	删除一条 usb 白名单.....	61
3.3.98.	清空白名单.....	61
3.3.99.	用户 LED 控制.....	61
3.3.100.	获取第一启动 APP.....	62
3.3.101.	设置第一启动 APP.....	62
3.3.102.	开启串口.....	63
3.3.103.	获取串口输入流.....	64
3.3.104.	获取串口输出流.....	64
3.3.105.	串口使用示例.....	64
4.	联系方式.....	67
5.	免责声明.....	67
6.	更新历史.....	67

1. 产品概述

1.1. 产品简介

USR-EPC500 是有人物联网 2017 年推出的新品。将 ARM 架构的处理器与 Android 操作系统进行有机结合，为强调视频和图像处理效果的设备制造商带来全新的解决方案。USR-EPC500 系列嵌入式计算平台使用 NXP 四核处理器，并运行 Android 操作系统，为以往依赖 X86 架构计算平台的设备制造商（尤其是媒体广告播放设备制造商）带来功耗更低发热更小的解决方案。USR-EPC500 支持 OpenGLES2.0 和 OpenVG™1.1 硬件加速器，全高清 1080P 视频编解码硬件引擎，为媒体播放设备提供强劲流畅的视频体验。

USR-EPC500 拥有双以太网口，其中 WAN 口支持 10M/100M/1000M 自适应切换，LAN 口则为 10M/100M 自适应切换；还拥有一个 4G 上网卡，其支持联通移动全网通以及电信 4G，无论更新媒体内容，还是实现远程联网，都能实现设备的“永远在线”。除此之外，通过 USR-EPC500Settings 对以太网卡和 4G 网卡的配置，工控机可实现 4G 路由器的功能；USR-EPC500 具备丰富的外围接口，6 个串口，其中两个具备 RS232, RS485, RS422 功能，剩下 4 个全为标准 RS232，4 个高速 USB2.0 接口；视频显示接口：HDMI。使设备可灵活连接扫描枪、打印机、二代身份证读卡器、交通一卡通读卡器、POS 机，红外触摸屏、摄像头、鼠标，液晶屏等多种外设。

1.2. 产品特点

- 基于飞思卡尔 QUAD-CORE Cortex-A9 架构的处理器 i.MX6Q，带来强劲的计算能力
- 优化的硬件底层驱动带来更稳定快速的网络连接，更流畅的操作体验，更强劲的外设性能
- 运行 Android 操作系统，享受开放的 Android 开发资源和丰富的 Android 应用软件
- 支持 OpenGLES2.0 和 OpenVG™1.1 硬件加速器，支持 2D,3D 图形加速
- 全高清 1080P 视频编解码器，带来酣畅淋漓的视频体验
- 支持视频多路输出以及分辨率设置
- 1000M/100M 双以太网和 4G/3G /2G 无线网络带来丰富的网络能力
- 支持 WAN 口和 LAN 自由转换，摇身变为工业 4G 路由器
- 支持流量统计，流量清空，流量报警
- 专属定制的系统级的设置软件，让设置变得轻松、可靠
- 提供 EPC500 专属 api，让二次开发变得更加简单、合理
- 丰富的外设接口，支持设备厂商外接多种外设
- 工业级别的硬件看门狗，设备异常自启自恢复
- 一键升级，告别繁琐

2. 产品功能

2.1. EPC500 状态信息查看


2.1.1. 状态栏



图 1

USR-EPC500Settings 状态信息如下

表 1

参数	描述
LTE	SIM 卡的网络制式
中国电信	运营商类型
	信号状态，信号强度由 1 格至 4 格，信号强度依次变大
EPC500 Ver	软件版本
Kernel Ver	内核版本
更新时间	最后一次更新时间

2.1.2. 网络信息



图 2

USR-EPC500Settings 网络信息参数如下

表 2

参数	描述
当前网卡	代表当前用于连接网络的网卡
系统运行时间	系统开机时长
实时网速	当前工控机实时网速
无线网卡	高亮的则代表当前被用于连接网络，且以太网卡一和以太网卡二支持 WAN/LAN 相互切换，并显示当前状态网卡模式 (WAN/LAN)
以太网卡一	
以太网卡二	

网卡信息如下：

移动数据信息：	
移动设备身份码(IMEI)：	358511040081402
移动用户识别码(IMS)：	460110319458266
网卡数据信息：	
ip地址(IP)：	192.168.4.116
网关地址(GW)：	192.168.4.1
域名解析(DNS)：	202.102.128.68
子网掩码(MASK)：	255.255.255.0
以太网卡一地址(MAC)：	1EED:19:27:1A:B3
以太网卡二地址(MAC)：	00:0E:C6:8F:0A:14

图 3

联网信息传输如下

表 3

参数	描述
ip 地址	当前 EPC500 的 ip
网关地址(GW)	当前 EPC500 的网关
域名解析(DNS)	当前 EPC500 的 dns
子网掩码(MASK)	当前 EPC500 的子网掩码
以太网卡一、以太网卡二的 MAC	分别代表 EPC500 两块网卡的 mac 地址

2.2. 串口

USR-EPC500 支持六路独立串口通信，并且 COM2、COM4 支持 RS232/RS485/RS422 相互切换。串口拨码开关功能映射如下

表 4

串口名	拨码开关	RS232 功能	RS485 功能	RS422 功能
COM2	KEYB	11	00	10
COM4	KEYA	11	00	10

<说明>:

- 拨码开关指向“on”为 1，反之则为 0;
 - USR-EPC500 提供的串口工具与其他串口工具并无差别，故对使用方法不再赘述；
- 用户可以通过 USR-EPC500Settings 中的“串口设置”使用 COM1~6 进行通信。使用界面如下，

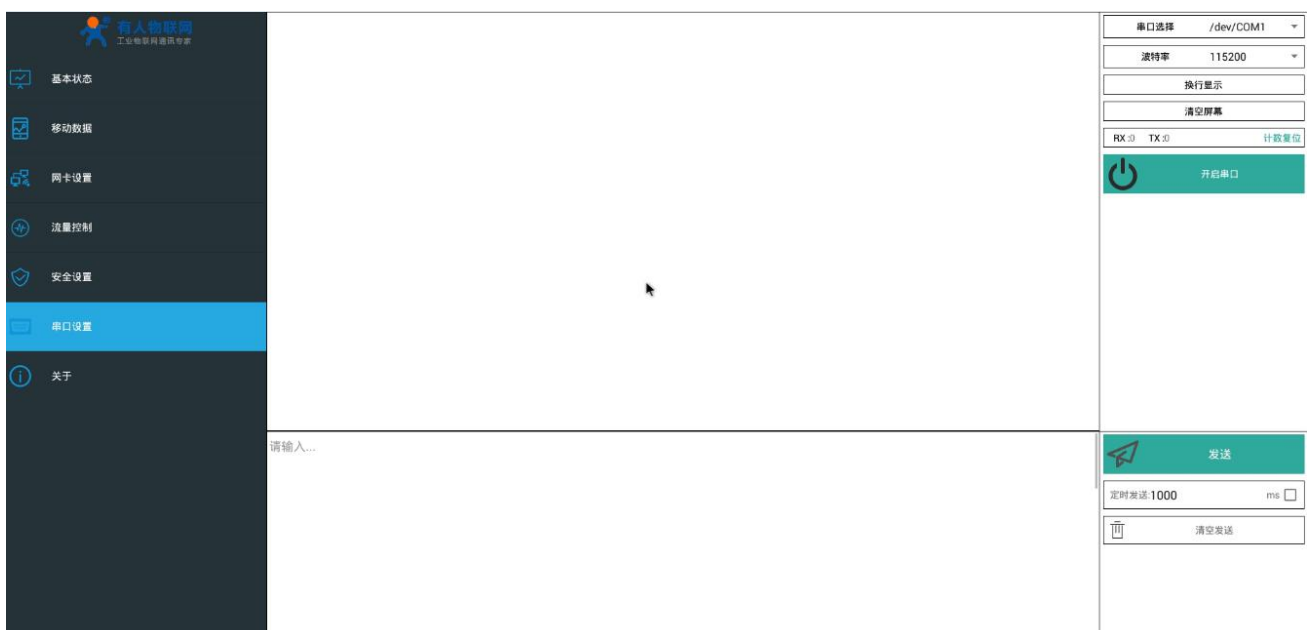


图 4

2.3. USB 白名单功能

配合 USR-EPC500Settings 用户可以实现对存储设备的挂载进行管理。用户开启白名单功能后,USR-EPC500 将禁止白名单之外的所有 USB 存储设备。

配置步骤:

1.插入 U 盘,安全设置并点击“目前插入存储设备”的刷新按钮,可以看到目前插入的存储设备 ID,



图 5

2.点击“添加”即可将该 U 盘 ID 添加进白名单列表,同时在“设备列表”处显示当前已被添加至白名单的设备 ID 及数量,如下图(如要将指 U 盘的 ID 从白名单中删除设备时,直接点击 ID 后的“删除”按钮即可),



图 6

3.点击“开启白名单功能”按钮;

4.拔出 U 盘;

此时插入未添加至白名单的 U 盘,则无法挂载至 EPC500,再插入之前添加过白名单的 U 盘,则可以在 `udisk` 目录下挂载。(可以使用自带 ES 文件管理器查看 `udisk` 文件夹);

<说明>

- 同时插入多个 U 盘时，USR-EPC500 只挂载第一个接入的 U 盘；
- 白名单设置成功后，对下次插拔生效；
- 插拔 U 盘前先进入设置对 U 盘进行卸载，否则会造成无法挂载的现象；



图 7

2.4. 网络连接

USR-EPC500 具有三个网卡分别为以太网卡一、以太网卡二、4G 无线网卡。其中以太网卡一、以太网卡二支持 LAN/WAN 功能切换。

2.4.1. 指定 4G 无线网卡上网

2.4.1.1. 普通卡连接上网：

移动数据：

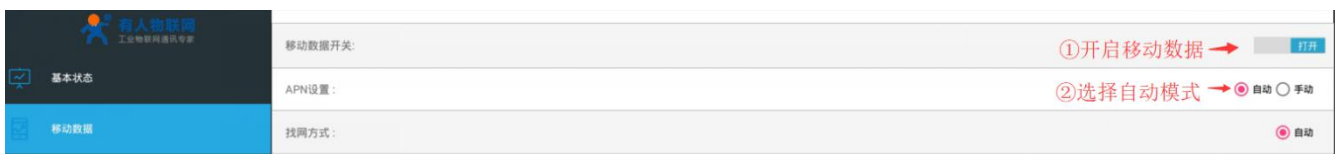


图 8

网卡设置：



图 9

<说明>

- 使用 4G 无线上网卡连接上网，需要在开机前即插入 sim 卡（支持移动、联通全网通，电信则只支持 4G）；
- 4G 上网卡连接上网稍慢，耐心等待 NET 灯亮起即可上网；

2.4.1.2. APN 卡连接上网

1. 配置 APN 卡信息，配置如下，



图 10

2. 关闭其他网卡，网卡配置如下，



图 11

<说明>:

➢ 运营商 APN 名映射表

表 5

参数	描述
中国联通	3gnet
中国移动	cmnet
中国电信	ctnet

- 使用 4G 无线上网卡连接上网，需要在开机前即插入 sim 卡（支持移动、联通全网通，电信则只支持 4G）；
- 4G 上网卡连接上网稍慢，耐心等待 NET 灯亮起即可上网；

2.4.2. 指定以太网卡一上网

2.4.2.1. 动态获取联网信息

1. 将网线接入 USR-EPC500 的 WAN 口
2. 关闭 4G 上网卡，配置如下，



图 12

3. 配置以太网卡,配置如下，



图 13

<说明>

- 接入的网络需要支持动态 ip 获取；

2.4.2.2. 设置 epc500 为静态 ip 上网:

- 1.将网线接入 EPC500 的 WAN 口
2. 关闭 4G 上网卡，配置如下，



图 14

- 3.配置网卡上网信息，配置如下，



图 15

<说明>

- 虽然用户设置了 dns，但是 epc500 会自动去匹配当前网络下最优的 DNS，所以出现用户设置 dns 与最终 dns 会有差异。

2.4.3. 指定以太网卡二上网

配置方法与以太网卡一完全一致，故不再赘述。

2.4.4. 全自动联网

1. 配置移动数据，配置如下，



图 16

2. 配置以太网卡，以及优先级，配置如下，



图 17

<说明>:

- 此时 USR-EPC500 网卡优先级从高至低依次为以太网卡一、4G 无线网卡、以太网卡二，故此处配置下在以太网卡一无法连接网络时，EPC500 则自动使用 4G 无线网卡连接上网,一旦以太网卡可正确连接网络，EPC500 将自动切换至以太网卡一；

2.5. 路由器功能

2.5.1. 有线路由器

USR-EPC500 出厂默认以太网口一为 WAN 口，以太网口二为 LAN 口，用户可以根据需求自行更改设置。
配置以太网卡一和以太网卡二分别为 LAN 口、WAN 口：

1.配置网卡设置，配置如下，



图 18

2.使用网线将 PC 与 EPC500 的 LAN 功能口直连。

配置 LAN 口网段

3. 如果以太网卡的模式被设置为 LAN，那么关闭 DHCP 后可以对路由器网段进行设置，配置图如下



图 19

LAN 口限速



图 20

2.5.2. 4G 路由器

以太网口一、以太网口二都可以被配置为 LAN 口，以下使用以太网口一作为 LAN 演示。

1.配置移动数据，配置如下，



图 21

2. 配置网卡，配置如下，



图 22

2. 使用网线将 PC 与 EPC500 的 LAN 功能口直连。

配置 LAN 口网段

3. 如果以太网卡的模式被设置为 LAN，那么关闭 DHCP 后可以对路由器网段进行设置，配置图如下



图 23

LAN 口限速



图 24

<说明>:

➢ USR-EPC500 同一时刻只能存在一个 LAN 口，即以太网卡一与以太网卡二不能同时被设置为 LAN 口模式；

2.6. 流量控制

用户可在此界面查看各个网卡的流量使用情况，并且可以对 4G 上网卡的流量设置流量阈值，一旦 4G 无线网卡剩余流量少于用户设置的剩余报警流量，USR-EPC500 将自动以短信的形式发送报警信息）。

2.6.1. 查看网卡一使用的总流量

1. 选择以太网卡一



图 25

2. 流量显示如下，



图 26

3. 清空选中网卡的流量



图 27

2.6.2. 配置流量报警

配置流量报警，配置如下，



图 28

取消流量报警，配置如下，



图 29

<说明>:

- 由于 EPC500 只支持电信 4G 故不支持电信卡发送短信；
- 流量控制参数说明如下

表 6

参数	描述
当前网卡	显示当前用于联网的设备
选择网卡	通过链表选择，用户可以自由查看以太网卡一、以太网卡二以及无线卡的流量使用情况
流量统计	用于显示被选择网卡的流量使用情况，并且单位会以更具流量使用的量级自动的在 KB、MB、GB、TB 之间切换
流量总数	用于设置当前流量卡可使用的总流量
流量报警	用于设置流量卡的报警阈值，即剩余流量
报警电话	管理员电话号码
按钮“取消流量报警”	设置后，将取消流量报警功能
按钮“清空流量统计”	清空选中网卡的流量统计

2.7. 视频

2.7.1. 调节输出分辨率

配置显示分辨率，配置如下，

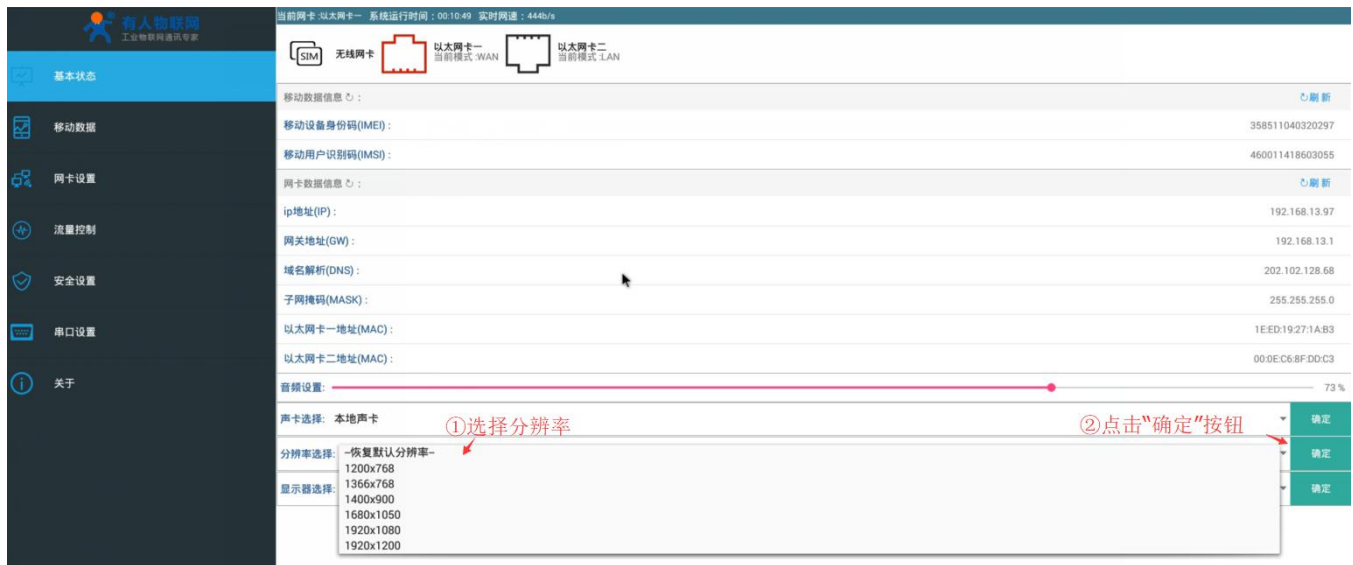


图 30

2.7.2. 视频输出选择

配置 USR-EPC500 HDMI 接口输出，配置如下，

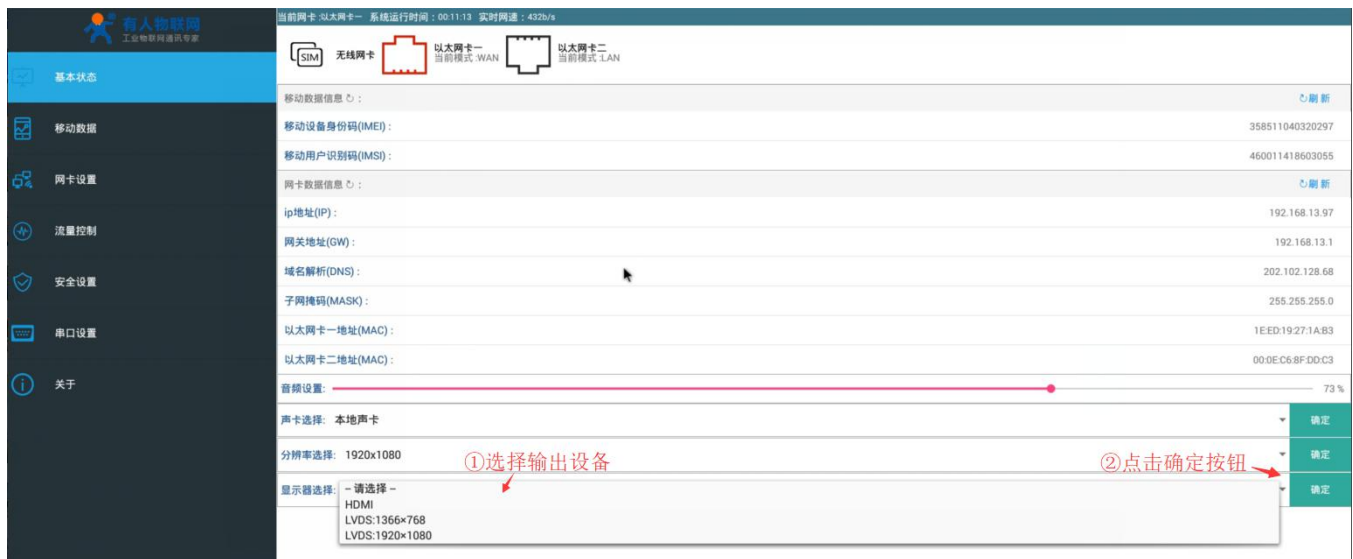


图 31

<说明>:

- 视频设置成功后在下次重启生效;
- 选择 HDMI,默认为 1080p 输出;

2.8. 音频

2.8.1. 音频输出选择

USR-EPC500 支持音频声道输出选择，如果用户仅仅使用 HDMI 接口作为视频输出，那么用户可以设置音频从本地声卡输出。

配置本地声卡作为音频输出，配置如下，



图 32

<说明>:

- 设置完声卡通道后需要重启 USR-EPC500;

2.8.2. 音量调节



图 33

2.9. 软件升级

开启 USR-EPC500Settings 时，会检查是否有可用更新，用户可以点击“直接下载”，也可以点击“以后再说”后续再更新。用户也可以在 USR-EPC500Settings 的“关于”页点击检查更新进行手动更新。

2.10. 修改管理员密码

1. 点击修改密码，

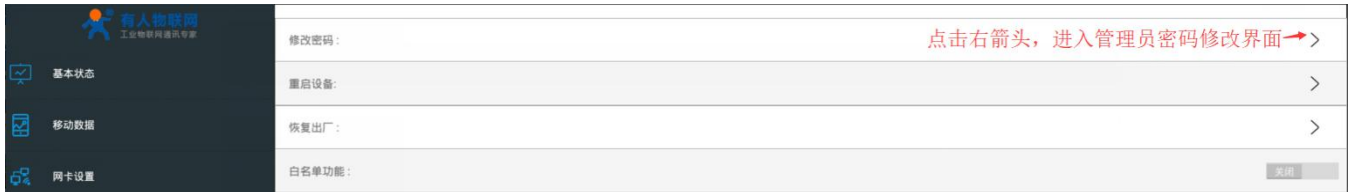


图 34

2. 创建新密码



图 35

2.11. 重启设备

1. 点击“重启设备”按钮，

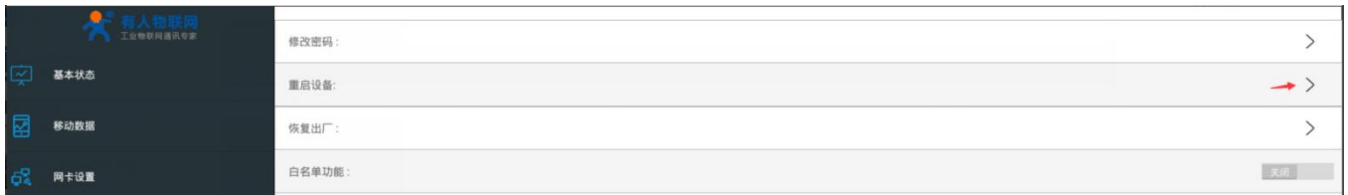


图 36

2. 在对话框中选中“确定”，



图 37

<说明>:

- 点击确定后将重启设备，注意资料的保存；

2.12. 恢复出厂设置

1. 点击“恢复出厂”按钮



图 38

2. 点击对话框中的“确定”按钮



图 39

<说明>

1. 恢复出厂值将恢复通过 USR-Settings 设置的配置项。

2.13. 指示灯

表 7

参数	功能描述	详细介绍
信号强度	4G 网卡信号指示灯	信号强度越强，灯亮的数量则越多
NET	网络连接指示灯	亮：工控机正确联网 灭：工控机无法正常联网
WORK	EPC500 工作状态灯	常亮或者常灭：工控机系统未正确启动 每秒闪烁：工控机系统已正确启动
PWR	EPC500 电源指示灯	亮：工控机电源正常 灭：工控机非正常连接电源
REV	保留，供用户二次开发使用	用户自定义使用

<说明>:

➤ 保留指示灯使用接口详见《USR-EPC500 软件设计手册》；

2.14. 按键

USR-EPC500 具有音量加、音量减、BACK 以及 HOME 四个实体按键，并且对接入的鼠标按键也做了重定位。

表 8

鼠标按键名	功能描述
右键	返回
滚轮	向上则为模拟触摸“向右滑动”
	向下则为模拟触摸“向左滑动”

3. 有人 SDK

“有人 SDK”能够让用户在对 android 系统完全不了解的情况下将工控的网络、LED、屏幕显示路径、分辨率、音频输出路径、usb 存储设备的挂载、串口等功能集成到用户 APP 中。在阅读 SDK 接口前，建议先熟悉产品介绍中 **USR-EPC500Settings** 的相关操作，其中的核心功能都是基于“有人 SDK”实现的。

3.1. 准备工作

3.1.1. 添加库文件

1. 需要有该目录结构 YouProject\app\src\main\jniLibs\armeabi
2. 直接添加 libserial_port.so、libusr-client.so 至 armeabi 目录

3.1.2. 添加 java 类

1. 建立如下包结构 com.usr.jni.main、com.wits.serialport
2. 将 UsrJniClass.java 类放置至 com.usr.jni.main 目录下
3. 将 SerialPort.java 类放置至 com.wits.serialport 目录下

3.2. 返回值说明

表 9

接口	返回值类型	返回值	含义
Public String infoCmdSend(String str)	String	0 ERROR	执行成功
		1 ERROR	命令执行失败
		2 ERROR	文件不存在
		3 ERROR	文件建立失败
		4 ERROR	写文件失败
		5 ERROR	读文件失败
		6 ERROR	改变文件属性操作失败
		7 ERROR	无效的参数/命令
Public Int noInfoCmdSend(String str)	Int	0	执行成功
		1	命令执行失败
		2	文件不存在
		3	文件建立失败
		4	写文件失败
		5	读文件失败
		6	改变文件属性操作失败
		7	无效的参数/命令

<说明>:

- 对于“0 ERROR”，0 和 ERROR 之间存在一个空格，便于返回字符串类接口提取错误类型；

3.3. 指令说明

3.3.1. 获取 Sim 卡网络制式

使用接口:

```
public String infoCmdSend(String str);
```

参数:

```
usr-get-get_network_info_property 4G_NETWORK_TYPE
```

返回值:

[4G_NETWORK_TYPE] GSM

[4G_NETWORK_TYPE] LTE

[4G_NETWORK_TYPE] TDSCDMA

[4G_NETWORK_TYPE] WCDMA

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.2. 获取运营商类型

使用接口:

```
public String infoCmdSend(String str);
```

参数:

```
usr-get-get_network_info_property 4G_OPERATOR_TYPE
```

返回值:

[4G_OPERATOR_TYPE] 1: “中国联通”

[4G_OPERATOR_TYPE] 2: “中国移动”

[4G_OPERATOR_TYPE] 3: “中国电信”

[4G_OPERATOR_TYPE] 0: “无运营商”

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.3. 获取 4G 网卡信号强度

使用接口:

```
public String infoCmdSend(String str);
```

参数:

```
usr-get-get_network_info_property 4G_SIGNAL_STRENGTH
```

返回值:

[4G_SIGNAL_STRENGTH] 1:信号一格

[4G_SIGNAL_STRENGTH] 2:信号两格

[4G_SIGNAL_STRENGTH] 3:信号三格

[4G_SIGNAL_STRENGTH] 4:信号四格

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.4. 获取当前使用的网卡

使用接口:

```
public String infoCmdSend(String str);
```

参数:

```
usr-get-get_network_info_property DEVICE
```

返回值:

```
[DEVICE] eth0: 网卡一
```

```
[DEVICE] ppp0: 4G 上网卡
```

```
[DEVICE] eth2: 以太网卡二
```

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.5. 获取当前 LAN 功能设备

使用接口:

```
public String infoCmdSend(String str);
```

参数:

```
usr-get-get_network_info_property LAN_DEVICE
```

返回值:

```
[LAN_DEVICE] eth0 : 以太网口一作为 LAN 口
```

```
[LAN_DEVICE] eth2 : 以太网口二作为 LAN 口
```

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.6. 获取 IMEI

使用接口:

```
public String infoCmdSend(String str);
```

参数:

```
usr-get-get_ethx_property IMEI ppp0
```

返回值:

```
[IMEI] <IMEI 码>
```

```
[IMEI] null: 代表工控机当前无 IMEI
```

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.7. 获取 IMSI

使用接口:

```
public String infoCmdSend(String str);
```

参数:

```
usr-get-get_ethx_property IMSI ppp0
```

返回值:

[IMSI] <IMSI 码>

[IMSI] null: 代表工控机当前无 IMSI

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.8. 获取工控机联网状态

使用接口:

```
public String infoCmdSend(String str);
```

参数:

```
usr-get-get_network_info_property STATUS
```

返回值:

[STATUS] yes: 工控机已连接网络

[STATUS] no: 工控机未连接网络

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.9. 获取 IP 信息

使用接口:

```
public String infoCmdSend(String str);
```

参数:

```
usr-get-get_network_info_property IP
```

返回值:

[IMEI] <IP 地址>

[IMEI] null: 代表工控机未连接网络

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.10. 获取当前子网掩码

使用接口:

```
public String infoCmdSend(String str);
```

参数:

```
usr-get-get_network_info_property MASK
```

返回值:

[MASK] <MASK 地址>

[MASK] null: 代表工控机未连接网络

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.11. 获取当前网关

使用接口:

```
public String infoCmdSend(String str);
```

参数:

```
usr-get-get_network_info_property GW
```

返回值:

[GW] <GW 地址>

[GW] null: 代表工控机未连接网络

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.12. 获取当前 DNS

使用接口:

```
public String infoCmdSend(String str);
```

参数:

```
usr-get-get_network_info_property DNS
```

返回值:

[DNS] <DNS 地址>

[DNS] null: 代表工控机未连接网络

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.13. 获取 WAN 口 MAC

使用接口:

```
public String infoCmdSend(String str);
```

参数:

```
usr-get-get_ethx_property MAC eth0
```

返回值:

[MAC] <MAC 地址>

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.14. 获取 LAN 口 MAC

使用接口:

```
public String infoCmdSend(String str);
```

参数:

```
usr-get-get_network_info_property IP
```

返回值:

[IMEI] <IP 地址>

[IMEI] null: 代表工控机未连接网络

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.15. 获取当前分辨率

使用接口:

```
public String infoCmdSend(String str);
```

参数:

```
usr-get-get_epc_base_info_property RESOLUTION
```

返回值:

[RESOLUTION] <长分辨率×宽分辨率>: 如 1920*1080

要求:

```
import com.usr.jni.main.UsrJniClass
```


3.3.16. 获取当前显示设备

使用接口:

```
public String infoCmdSend(String str);
```

参数:

```
usr-get-get_network_info_property IP
```

返回值:

[IMEI] <IP 地址>

[IMEI] null: 代表工控机未连接网络

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.17. 获取当前声卡输出

使用接口:

```
public String infoCmdSend(String str);
```

参数:

```
usr-get-audio_selected_dev
```

返回值:

audio: 本地声卡输出

hdmi: HDMI 声卡输出

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.18. 设置显示设备

使用接口:

```
public Int noinfoCmdSend(String str);
```

参数:

```
usr-get-get_network_info_property IP
```

返回值:

0: 成功

其他值: 失败

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.19. 设置分辨率

使用接口:

```
public Int noinfoCmdSend(String str);
```

参数:

```
usr-set-set_screen_resolution <resolution_length> <resolution_height>
```

resolution_length: 长分辨率

resolution_height: 高分辨率

返回值:

0: 设置成功

其他值: 失败

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.20. 设置当前声卡

使用接口:

```
public Int noinfoCmdSend(String str);
```

参数:

```
usr-set-audio_route_select <audio_device>
```

audio_device:

audio: 使用本地声卡

hdmi: hdmi 声卡输出

返回值:

0: 设置成功

其他值: 失败

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.21. 获取移动数据开关状态

使用接口:

```
public String infoCmdSend(String str);
```

参数:

```
usr-get-get_ethx_property STATUS ppp0
```

返回值:

[STATUS] on: 关闭

[STATUS] off: 关闭

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.22. 获取 APN 模式

使用接口:

```
public String infoCmdSend(String str);
```

参数:

```
usr-get-get_ethx_property APN_MODE ppp0
```

返回值:

[APN_MODE] auto: 自动匹配 APN 账号

[APN_MODE] manual: 手动匹配 APN 账号模式

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.23. 获取 4G 网卡寻网方式

使用接口:

```
public String infoCmdSend(String str);
```

参数:

```
usr-get-get_ethx_property FIND_NETWORK_TYPE ppp0
```

返回值:

[FIND_NETWORK_TYPE] auto: 自动寻找网络

要求:

```
import com.usr.jni.main.UsrJniClass
```

说明:

固件版本 V1.0.1 只不支持手动设置 4G/3G/2G 寻网, 只支持自动寻网。

3.3.24. 获取 APN 名

使用接口:

```
public String infoCmdSend(String str);
```

参数:

```
usr-get-get_ethx_property APN ppp0
```

返回值:

[APN] <APN 类型>

要求:

```
import com.usr.jni.main.UsrJniClass
```

说明:

<APN 类型>: 3gnet、cmnet、ctnet

3.3.25. 获取 APN 卡用户名

使用接口:

```
public String infoCmdSend(String str);
```

参数:

```
usr-get-get_ethx_property APN_USR ppp0
```

返回值:

[APN_USR] <APN 账号>

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.26. 获取 APN 卡密码

使用接口:

```
public String infoCmdSend(String str);
```

参数:

```
usr-get-get_ethx_property APN_PW ppp0
```

返回值:

[APN_PW] <APN 密码>

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.27. 设置移动数据开关状态

使用接口:

```
public String noInfoCmdSend(String str);
```

参数:

```
usr-set-set_ethx_property STATUS ppp0 <status>
```

status:

on: 开启移动数据

off: 关闭移动数据

返回值:

0: 设置成功

其他值: 设置失败

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.28. 设置 APN 模式

使用接口:

```
public String noInfoCmdSend(String str);
```

参数:

```
usr-set-set_ethx_property APN_MODE ppp0 <values>
```

values:

auto:设置自动匹配 APN 账号

manual:设置为用户输出 APN 账号

返回值:

0: 设置成功

其他值: 设置失败

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.29. 设置 4G 网卡寻网方式

使用接口:

```
public String noInfoCmdSend(String str);
```

参数:

```
usr-get-get_ethx_property FIND_NETWORK_TYPE ppp0 <mode>
```

values:

auto:设置 4G 网卡自动匹配网络类型

返回值:

0: 设置成功

其他值: 设置失败

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.30. 设置 APN 名

使用接口:

```
public String noInfoCmdSend(String str);
```

参数:

```
usr-get-get_ethx_property APN ppp0 <values>
```

values:

3gnet: 中国移动卡

cmnet: 中国联通卡

ctnet: 中国电信卡

返回值:

0: 设置成功

其他值: 设置失败

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.31. 设置 APN 卡密码

使用接口:

```
public String noInfoCmdSend(String str);
```

参数:

```
usr-set-set_ethx_property APN_PW ppp0 <APN_PW>
```

APN_PW:

用户输入 APN 卡密

返回值:

0: 设置成功

其他值: 设置失败

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.32. 设置 APN 账号

使用接口:

```
public String noInfoCmdSend(String str);
```

参数:

```
usr-set-set_ethx_property APN_USR ppp0 <APN_NAME>
```

APN_NAME:

用户输入的 APN 账号

返回值:

0: 设置成功

其他值: 设置失败

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.33. 获取以太网卡一开关状态

使用接口:

```
public String infoCmdSend(String str);
```

参数:

```
usr-get-get_ethx_property STATUS eth0
```

返回值:

[STATUS] on:开启

[STATUS] off:关闭

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.34. 获取以太网卡一模式

使用接口:

```
public String infoCmdSend(String str);
```

参数:

```
usr-get-get_ethx_property MODE eth0
```

返回值:

[MODE] wan:wan 模式

[MODE] lan:lan 模式

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.35. 获取以太网卡一 DHCP 开关状态

使用接口:

```
public String infoCmdSend(String str);
```

参数:

```
usr-get-get_ethx_property DHCP eth0
```

返回值:

[DHCP] on:开启

[DHCP] off:关闭

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.36. 获取以太网卡一静态 IP

使用接口:

```
public String infoCmdSend(String str);
```

参数:

```
usr-get-get_ethx_property IP eth0
```

返回值:

```
[IP] <IP 地址>
```

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.37. 获取以太网卡一静态 MASK

使用接口:

```
public String infoCmdSend(String str);
```

参数:

```
usr-get-get_ethx_property MASK eth0
```

返回值:

```
[MASK] <MASK 地址>
```

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.38. 获取以太网卡一静态网关

使用接口:

```
public String infoCmdSend(String str);
```

参数:

```
usr-get-get_ethx_property GW eth0
```

返回值:

```
[GW] <GW 地址>
```

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.39. 获取以太网卡一静态 DNS

使用接口:

```
public String infoCmdSend(String str);
```

参数:

```
usr-get-get_ethx_property DNS eth0
```

返回值:

```
[DNS] <DNS 地址>
```

要求:

```
import com.usr.jni.main.UsrJniClass
```


3.3.40. 获取以太网卡一作为 LAN 口时网段

使用接口:

```
public String infoCmdSend(String str);
```

参数:

```
usr-get-get_ethx_property LAN_IP eth0
```

返回值:

```
[LAN_IP] <LAN_IP 地址>
```

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.41. 获取以太网卡一 LAN 口限速开关状态

使用接口:

```
public String infoCmdSend(String str);
```

参数:

```
usr-get-get_ethx_property LIMIT_RATE_STATUS eth0
```

返回值:

```
[LIMIT_RATE_STATUS] on:开启
```

```
[LIMIT_RATE_STATUS] off:关闭
```

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.42. 获取以太网卡一 LAN 口下载速度

使用接口:

```
public String infoCmdSend(String str);
```

参数:

```
usr-get-get_ethx_property LAN_DOWNLOAD_RATE eth0
```

返回值:

```
[LAN_DOWNLOAD_RATE] <限速值>
```

要求:

```
import com.usr.jni.main.UsrJniClass
```

说明:

单位: KB/S

3.3.43. 获取以太网卡一 LAN 口上传速度

使用接口:

```
public String infoCmdSend(String str);
```

参数:

```
usr-get-get_ethx_property LAN_UPLOAD_RATE eth0
```

返回值:

```
[LAN_UPLOAD_RATE] <限速值>
```

要求:

```
import com.usr.jni.main.UsrJniClass
```

说明:

单位: KB/S

3.3.44. 获取以太网卡一优先级

使用接口:

```
public String infoCmdSend(String str);
```

参数:

```
usr-get-get_priority_property eth0
```

返回值:

HIGH: 最高优先级

MIDDLE: 中间优先级

LOW: 最低优先级

要求:

```
import com.usr.jni.main.UsrJniClass
```

说明:

单位: KB/S

3.3.45. 获取以太网卡二开关状态

使用接口:

```
public String infoCmdSend(String str);
```

参数:

```
usr-get-get_ethx_property STATUS eth2
```

返回值:

```
[STATUS] on:开启
```

```
[STATUS] off:关闭
```

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.46. 获取以太网卡二模式

使用接口:

```
public String infoCmdSend(String str);
```

参数:

```
usr-get-get_ethx_property MODE eth2
```

返回值:

[MODE] wan:wan 模式

[MODE] lan:lan 模式

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.47. 获取以太网卡二 DHCP 开关状态

使用接口:

```
public String infoCmdSend(String str);
```

参数:

```
usr-get-get_ethx_property DHCP eth2
```

返回值:

[DHCP] on:开启

[DHCP] off:关闭

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.48. 获取以太网卡二静态 IP

使用接口:

```
public String infoCmdSend(String str);
```

参数:

```
usr-get-get_ethx_property IP eth2
```

返回值:

[IP] <IP 地址>

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.49. 获取以太网卡二静态 MASK

使用接口:

```
public String infoCmdSend(String str);
```

参数:

```
usr-get-get_ethx_property MASK eth2
```

返回值:

```
[MASK] <MASK 地址>
```

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.50. 获取以太网卡二静态网关

使用接口:

```
public String infoCmdSend(String str);
```

参数:

```
usr-get-get_ethx_property GW eth2
```

返回值:

```
[GW] <GW 地址>
```

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.51. 获取以太网卡二静态 DNS

使用接口:

```
public String infoCmdSend(String str);
```

参数:

```
usr-get-get_ethx_property DNS eth2
```

返回值:

```
[DNS] <DNS 地址>
```

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.52. 获取以太网卡二作为 LAN 口时网段

使用接口:

```
public String infoCmdSend(String str);
```

参数:

```
usr-get-get_ethx_property LAN_IP eth2
```

返回值:

```
[LAN_IP] <LAN_IP 地址>
```

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.53. 获取以太网卡二 LAN 口限速开关状态

使用接口:

```
public String infoCmdSend(String str);
```

参数:

```
usr-get-get_ethx_property LIMIT_RATE_STATUS eth2
```

返回值:

```
[LIMIT_RATE_STATUS] on:开启
```

```
[LIMIT_RATE_STATUS] off:关闭
```

要求:

```
import com.usr.jni.main.UsrJniClass
```

KB/S

3.3.54. 获取以太网卡二 LAN 口下载速度

使用接口:

```
public String infoCmdSend(String str);
```

参数:

```
usr-get-get_ethx_property LAN_DOWNLOAD_RATE eth2
```

返回值:

```
[LAN_DOWNLOAD_RATE] <限速值>
```

要求:

```
import com.usr.jni.main.UsrJniClass
```

说明:

单位: KB/S

3.3.55. 获取以太网卡二 LAN 口上传速度

使用接口:

```
public String infoCmdSend(String str);
```

参数:

```
usr-get-get_ethx_property LAN_UPLOAD_RATE eth2
```

返回值:

```
[LAN_UPLOAD_RATE] <限速值>
```

要求:

```
import com.usr.jni.main.UsrJniClass
```

说明:

单位: KB/S

3.3.56. 获取以太网卡二优先级

使用接口:

```
public String infoCmdSend(String str);
```

参数:

```
usr-get-get_priority_property eth2
```

返回值:

HIGH: 最高优先级

MIDDLE: 中间优先级

LOW: 最低优先级

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.57. 获取 4G 上网卡优先级

使用接口:

```
public String infoCmdSend(String str);
```

参数:

```
usr-get-get_priority_property ppp0
```

返回值:

HIGH: 最高优先级

MIDDLE: 中间优先级

LOW: 最低优先级

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.58. 设置以太网卡一开关状态

使用接口:

```
public String noInfoCmdSend(String str);
```

参数:

```
usr-set-set_ethx_property STATUS eth0 <STATUS>
```

STATUS:

on:开启

off:关闭

返回值:

0: 设置成功

其他返回值: 设置失败

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.59. 设置以太网卡一模式

使用接口:

```
public Int noInfoCmdSend(String str);
```

参数:

```
usr-set-set_ethx_property MODE eth0 <MODE>
```

MODE:

wan:wan 模式

lan:lan 模式

返回值:

0: 设置成功

其他返回值: 设置失败

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.60. 设置以太网卡一 DHCP 开关状态

使用接口:

```
public Int noInfoCmdSend(String str);
```

参数:

```
usr-set-set_ethx_property DHCP eth0 <STATUS>  
STATUS
```

on:开启

off:关闭

返回值:

0: 设置成功

其他返回值: 设置失败

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.61. 设置以太网卡一静态 IP

使用接口:

```
public Int noInfoCmdSend(String str);
```

参数:

```
usr-set-set_ethx_property IP eth0 <IP>
```

IP: ip 地址

返回值:

0: 设置成功

其他返回值: 设置失败

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.62. 设置以太网卡一静态 MASK

使用接口:

```
public Int noInfoCmdSend(String str);
```

参数:

```
usr-set-set_ethx_property MASK eth0 <MASK>
```

MASK:子网掩码地址

返回值:

0: 设置成功

其他返回值: 设置失败

要求:

```
import com.usr.jni.main.UsrJniClass
```


3.3.63. 设置以太网卡一静态网关

使用接口:

```
public Int noInfoCmdSend(String str);
```

参数:

```
usr-set-set_ethx_property GW eth0 <GW>
```

GW:网关地址

返回值:

0: 设置成功

其他返回值: 设置失败

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.64. 设置以太网卡一静态 DNS

使用接口:

```
public Int noInfoCmdSend(String str);
```

参数:

```
usr-set-set_ethx_property DNS eth0 <DNS>
```

DNS:DNS 地址

返回值:

0: 设置成功

其他返回值: 设置失败

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.65. 设置以太网卡一作为 LAN 口时网段

使用接口:

```
public Int noInfoCmdSend(String str);
```

参数:

```
usr-set-set_ethx_property LAN_IP eth0 <LAN_IP>
```

返回值:

0: 设置成功

其他返回值: 设置失败

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.66. 设置以太网卡一 LAN 口限速开关状态

使用接口:

```
public Int noInfoCmdSend(String str);
```

参数:

```
usr-set-set_ethx_property LIMIT_RATE_STATUS eth0 <STATUS>
```

STATUS:

on: 开启

off: 关闭

返回值:

0: 设置成功

其他返回值: 设置失败

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.67. 设置以太网卡一 LAN 口上传速度

使用接口:

```
public Int noInfoCmdSend(String str);
```

参数:

```
usr-set-set_ethx_property LAN_UPLOAD_RATE eth0 <限速值>
```

返回值:

0: 设置成功

其他返回值: 设置失败

要求:

```
import com.usr.jni.main.UsrJniClass
```

说明:

单位: KB/S

3.3.68. 设置以太网卡一 LAN 口下载速度

使用接口:

```
public Int noInfoCmdSend(String str);
```

参数:

```
usr-set-set_ethx_property LAN_DOWNLOAD_RATE eth0 <限速值>
```

返回值:

0: 设置成功

其他返回值: 设置失败

要求:

```
import com.usr.jni.main.UsrJniClass
```

说明:

单位: KB/S

3.3.69. 设置以太网卡一优先级

使用接口:

```
public Int noInfoCmdSend(String str);
```

参数:

```
usr-set-set_priority_property eth0 <PRIORITY>
```

PRIORITY:

HIGH: 最高优先级

MIDDLE: 中间优先级

LOW: 最低优先级

返回值:

0: 设置成功

其他返回值: 设置失败

要求:

```
import com.usr.jni.main.UsrJniClass
```

说明:

单位: KB/S

3.3.70. 设置以太网卡二开关状态

使用接口:

```
public Int noInfoCmdSend(String str);
```

参数:

```
usr-set-set_ethx_property STATUS eth2 <STATUS>
```

STATUS:

on:开启

off:关闭

返回值:

0: 设置成功

其他返回值: 设置失败

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.71. 设置以太网卡二模式

使用接口:

```
public Int noInfoCmdSend(String str);
```

参数:

```
usr-set-set_ethx_property MODE eth2 <MODE>
```

MODE:

wan:wan 模式

lan:lan 模式

返回值:

0: 设置成功

其他返回值: 设置失败

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.72. 设置以太网卡二 DHCP 开关状态

使用接口:

```
public Int noInfoCmdSend(String str);
```

参数:

```
usr-set-set_ethx_property DHCP eth2 <STATUS>
```

STATUS

on:开启

off:关闭

返回值:

0: 设置成功

其他返回值: 设置失败

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.73. 设置以太网卡二静态 IP

使用接口:

```
public Int noInfoCmdSend(String str);
```

参数:

```
usr-set-set_ethx_property IP eth2 <IP>
```

IP: ip 地址

返回值:

0: 设置成功

其他返回值: 设置失败

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.74. 设置以太网卡二静态 MASK

使用接口:

```
public Int noInfoCmdSend(String str);
```

参数:

```
usr-set-set_ethx_property MASK eth2 <MASK>
```

MASK:子网掩码地址

返回值:

0: 设置成功

其他返回值: 设置失败

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.75. 设置以太网卡二静态网关

使用接口:

```
public Int noInfoCmdSend(String str);
```

参数:

```
usr-set-set_ethx_property GW eth2 <GW>
```

GW:网关地址

返回值:

0: 设置成功

其他返回值: 设置失败

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.76. 设置以太网卡二静态 DNS

使用接口:

```
public Int noInfoCmdSend(String str);
```

参数:

```
usr-set-set_ethx_property DNS eth2 <DNS>
```

DNS:DNS 地址

返回值:

0: 设置成功

其他返回值: 设置失败

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.77. 设置以太网卡二作为 LAN 口时网段

使用接口:

```
public Int noInfoCmdSend(String str);
```

参数:

```
usr-set-set_ethx_property LAN_IP eth2 <LAN_IP>
```

返回值:

0: 设置成功

其他返回值: 设置失败

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.78. 设置以太网卡二 LAN 口限速开关状态

使用接口:

```
public Int noInfoCmdSend(String str);
```

参数:

```
usr-set-set_ethx_property LIMIT_RATE_STATUS eth2 <STATUS>
```

STATUS:

on: 开启

off: 关闭

返回值:

0: 设置成功

其他返回值: 设置失败

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.79. 设置以太网卡二 LAN 口上传速度

使用接口:

```
public Int noInfoCmdSend(String str);
```

参数:

```
usr-set-set_ethx_property LAN_UPLOAD_RATE eth2 <限速值>
```

返回值:

0: 设置成功

其他返回值: 设置失败

要求:

```
import com.usr.jni.main.UsrJniClass
```

说明:

单位: KB/S

3.3.80. 设置以太网卡二 LAN 口下载速度

使用接口:

```
public Int noInfoCmdSend(String str);
```

参数:

```
usr-set-set_ethx_property LAN_DOWNLOAD_RATE eth2 <限速值>
```

返回值:

0: 设置成功

其他返回值: 设置失败

要求:

```
import com.usr.jni.main.UsrJniClass
```

说明:

单位: KB/S

3.3.81. 设置以太网卡二优先级

使用接口:

```
public Int noInfoCmdSend(String str);
```

参数:

```
usr-set-set_priority_property eth2 <PRIORITY>
```

PRIORITY:

HIGH: 最高优先级

MIDDLE: 中间优先级

LOW: 最低优先级

返回值:

0: 设置成功

其他返回值: 设置失败

要求:

```
import com.usr.jni.main.UsrJniClass
```

说明:

单位: KB/S

3.3.82. 设置 4G 上网卡优先级

使用接口:

```
public Int noInfoCmdSend(String str);
```

参数:

```
usr-set-set_priority_property ppp0 < PRIOTITY >
```

PRIOTITY:

HIGH: 最高优先级

MIDDLE: 中间优先级

LOW: 最低优先级

返回值:

0: 设置成功

其他返回值: 设置失败

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.83. 获取网卡一使用流量总数

使用接口:

```
public String InfoCmdSend(String str);
```

参数:

```
usr-get-get_ethx_property WAN_USED_FLOW eth0
```

返回值:

[WAN_USED_FLOW] <流量值>: 单位 byte

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.84. 获取流量报警阈值

使用接口:

```
public String InfoCmdSend(String str);
```

参数:

```
usr-get-get_ethx_property SOS_FLOW ppp0
```

返回值:

[SOS_FLOW] <阈值流量值>: 单位 MB

要求:

```
import com.usr.jni.main.UsrJniClass
```


3.3.85. 获取报警电话

使用接口:

```
public String InfoCmdSend(String str);
```

参数:

```
usr-get-get_epc_base_info_property SOS_NUMBER
```

返回值:

[SOS_NUMBER] <报警电话>

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.86. 获取网卡二使用流量总数

使用接口:

```
public String InfoCmdSend(String str);
```

参数:

```
usr-get-get_ethx_property WAN_USED_FLOW eth2
```

返回值:

[WAN_USED_FLOW] <流量值>: 单位 byte

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.87. 获取 4G 上网卡使用流量总数

使用接口:

```
public String InfoCmdSend(String str);
```

参数:

```
usr-get-get_ethx_property WAN_USED_FLOW eth2
```

返回值:

[WAN_USED_FLOW] <流量值>: 单位 byte

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.88. 设置流量报警阀值

使用接口:

```
public Int noInfoCmdSend(String str);
```

参数:

```
usr-set-set_ethx_property SOS_FLOW ppp0 <流量值>
```

流量值:

单位 MB

返回值:

0: 设置成功

其他值: 设置失败

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.89. 设置获取报警电话

使用接口:

```
public Int noInfoCmdSend(String str);
```

参数:

```
usr-set-set_epc_base_info_property SOS_NUMBER <tele_number>
```

tele_number:

报警对象电话号码

返回值:

0: 设置成功

其他值: 设置失败

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.90. 清空设置以太网卡一流量

使用接口:

```
public Int noInfoCmdSend(String str);
```

参数:

```
usr-set-set_ethx_property WAN_USED_FLOW eth0 0
```

返回值:

0: 设置成功

其他值: 设置失败

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.91. 清空以太网卡二流量

使用接口:

```
public Int noInfoCmdSend(String str);
```

参数:

```
usr-set-set_ethx_property WAN_USED_FLOW eth2 0
```

返回值:

0: 设置成功

其他值: 设置失败

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.92. 清空 4G 无线网卡流量

使用接口:

```
public Int noInfoCmdSend(String str);
```

参数:

```
usr-set-set_ethx_property WAN_USED_FLOW ppp0 0
```

返回值:

0: 设置成功

其他值: 设置失败

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.93. 获取白名单状态开关

使用接口:

```
public Int InfoCmdSend(String str);
```

参数:

```
usr-get-get_usb_white_state
```

返回值:

open: 开启

close: 关闭

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.94. 获取白名单列表

使用接口:

```
public Int InfoCmdSend(String str);
```

参数:

```
usr-get-get_white_dev_list
```

返回值:

格式:<number>|<ID1>|<ID2>

number 白名单总数

其中 ID1 格式为< idP1 idV1>

idP1 : 产品 id

idV1: 产家 id

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.95. 设置白名单开关状态

使用接口:

```
public Int noInfoCmdSend(String str);
```

参数:

```
usr-set-set_usb_white_state <STATUS>
```

返回值:

0:设置成功

其他值: 设置失败

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.96. 添加一条 usb 白名单

使用接口:

```
public Int noInfoCmdSend(String str);
```

参数:

```
usr-set-add_usb_white_item < idP > < idV >
```

idP1: 产品 id

idV1: 产家 id

返回值:

0:设置成功

其他值: 设置失败

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.97. 删除一条 usb 白名单

使用接口:

```
public Int noInfoCmdSend(String str);
```

参数:

```
usr-set-delete_usb_white_item < idP > < idV >
```

idP1: 产品 id

idV1: 产家 id

返回值:

0: 设置成功

其他值: 设置失败

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.98. 清空白名单

使用接口:

```
public Int noInfoCmdSend(String str);
```

参数:

```
usr-set-clear_usb_white_tables
```

返回值:

0: 设置成功

其他值: 设置失败

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.99. 用户 LED 控制

使用接口:

```
public Int noInfoCmdSend(String str);
```

参数:

```
usr-set-led_switch usr-custom-led <status>
```

status:

open: 点亮 LED

close: 熄灭 LED

返回值:

0: 设置成功

其他值: 设置失败

要求:

```
import com.usr.jni.main.UsrJniClass
```

3.3.100. 获取第一启动 APP

3.3.100.1. 获取包名

使用接口:

```
public String InfoCmdSend(String str);
```

参数:

```
usr-get-get_epc_base_info_property FIRST_PACKAGE
```

返回值:

```
[FIRST_PACKAGE] <包名>
```

3.3.100.2. 获取 Activity

使用接口:

```
public String InfoCmdSend(String str);
```

参数:

```
usr-get-get_epc_base_info_property FIRST_ACTIVITY
```

返回值:

```
FIRST_ACTIVITY <activity 路径全名>
```

要求:

```
import com.usr.jni.main.UsrJniClass
```

说明:

以下是 USR-EPC500Settings 做为第一 app 时, 包名以及 Activity

```
[FIRST_PACKAGE] cn.usr.www.epc_500setting
```

```
[FIRST_ACTIVITY] cn.usr.www.epc_500setting.Activity.WelcomeActivity
```

3.3.101. 设置第一启动 APP

3.3.101.1. 设置包名

使用接口:

```
public Int noInfoCmdSend(String str);
```

参数:

```
usr-set-set_epc_base_info_property FIRST_PACKAGE <包名>
```

<包名>:

用户 APP 启动第一页面所在包名

返回值:

0:设置成功

其他值: 设置失败

3.3.101.2. 设置 Activity

使用接口:

```
public Int noInfoCmdSend(String str);
```

参数:

```
usr-set-set_epc_base_info_property FIRST_ACTIVITY <activity 全名>  
< activity 全名>:
```

用户 APP 启动第一页面名字

返回值:

0:设置成功

其他值: 设置失败

要求:

```
import com.usr.jni.main.UsrJniClass
```

说明:

以下是 USR-EPC500Settings 作为第一 app 时, 报名以及 Activity

```
[FIRST_PACKAGE] cn.usr.www.epc_500setting
```

```
[FIRST_ACTIVITY] cn.usr.www.epc_500setting.Activity.WelcomeActivity
```

3.3.102. 开启串口

使用接口:

```
public SerialPort(File device, int baudrate, int flags);
```

参数:

device:

```
/dev/COM1
```

```
/dev/COM2
```

```
/dev/COM3
```

```
/dev/COM4
```

```
/dev/COM5
```

```
/dev/COM6
```

baudrate:

波特率

flags:

任意值

返回值:

```
SerialPort
```

要求:

```
import com.wits.serialport.SerialPort
```

3.3.103. 获取串口输入流

使用接口:

```
public InputStream getInputStream();
```

参数:

无

返回值:

InputStream

要求:

```
import com.wits.serialport.SerialPort
```

3.3.104. 获取串口输出流

使用接口:

```
public OutputStream getOutputStream()
```

参数:

无

返回值:

OutputStream

要求:

```
import com.wits.serialport.SerialPort
```

3.3.105. 串口使用示例

3.3.105.1. 打开串口

```
private void OpenPort() throws IOException {  
    mSerialPort = new SerialPort(new File(port), baud, 0);  
    //获取输入流  
    mInputStream = mSerialPort.getInputStream();  
    //开启线程接收数据  
    new Thread(runnable).start();  
}
```


3.3.105.2. 关闭串口

```
public void ClosePort() {
    try {
        if (mSerialPort != null) {
            mSerialPort.close();
            mSerialPort = null;
        }
    } catch (Exception e) {
        System.out.println("关闭失败");
    }
}
```

3.3.105.3. 读取数据线程

```
Runnable runnable = new Runnable() {
    public void run() {
        while (true) {
            try {
                int size = mInputStream.read(buffer);
                if (size > 0) {
                    Message message = handler.obtainMessage();
                    Bundle bundle = new Bundle();
                    bundle.putString("readdata", readdata);
                    bundle.putString("readdatal", String.valueOf(rxs));
                    message.setData(bundle);
                    handler.sendMessage(message);
                }
                Thread.sleep(500);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
    }
};
```

3.3.105.4. 发送数据线程

```
Runnable sendData = new Runnable() {
    public void run() {
        inputData = getInputBoxData();
        if (!inputBoxData.equals("")) {
            handler.post(new Runnable() {
                public void run() {
                    String string = tv_receivecount_tx.getText().toString();
                }
            });
            sendData(inputBoxData);
        } else {
            //用户没有输入值
            handler.post(new Runnable() {
                public void run() {
                    showSnackbar("请先输入要发送的内容");
                }
            });
        }
    }
};
```

4. 联系方式

公 司：济南有人物联网技术有限公司

地 址：山东省济南市高新区新泺大街 1166 号奥盛大厦 1 号楼 11 层

网 址：<http://www.usr.cn>

客户支持中心：<http://h.usr.cn>

邮 箱：sales@usr.cn

电 话：4000-255-652 或者 0531-88826739

有人愿景：拥有自己的有人大厦

公司文化：有人在认真做事!

产品理念：简单 可靠 价格合理

有人信条：天道酬勤 厚德载物 共同成长

5. 免责声明

本文档提供有关 USR-EPC500 产品的信息，本文档未授予任何知识产权的许可，并未以明示或暗示，或以禁止发言或其它方式授予任何知识产权许可。除在其产品的销售条款和条件声明的责任之外，我公司概不承担任何其它责任。并且，我公司对本产品的销售和/或使用不作任何明示或暗示的担保，包括对产品的特定用途适用性，适销性或对任何专利权，版权或其它知识产权的侵权责任等均不作担保。本公司可能随时对产品规格及产品描述做出修改，恕不另行通知。

6. 更新历史

2017-06-25 版本 V1.0.0 创立。

2017-07-25 版本 V1.0.1 创立。

2017-08-15 版本 V1.0.2 创立。

2017-08-24 版本 V1.0.3 创立。