



Doctors of Intelligence &Technology(DOIT)

W800 开发套件鸿蒙系统教程

(文档版本: V1.0 软件版本: V1.0 硬件版本: V1.0)

2022-04-21



目录

1.1.开发板硬件资料分配	<u> </u>	、 w800开发套件及鸿蒙系统介绍	. 3
1.2 供电 3 1.3 USB 接口 3 1.4 按键 3 1.5 RGB 彩灯 3 1.6 DS18B20 温度传感器 3 1.7 显示屏接口 3 1.8 光敏传感器 4 1.9 w800 鸿蒙系统介绍 4 2. 开发环境搭建 7 2.1 串口驱动的安装 7 2.1 串口驱动的安装 7 2.2 w800 Linux开发环境搭建 11 2.3 w800 鸿蒙系统环境搭建 12 三、外设基础实验 15 3.1 鸿蒙系统第一个驱动程序helloworld 15 3.1 鸿蒙系统第一个驱动程序helloworld 15 3.5w800 鸿蒙系统定时器实验 25 四、wifi 通信实验 30 4.1 w800 鸿蒙系统 wifi AP模式 30 4.1 w800 鸿蒙系统 wifi STA模式 34 4.3 w800 鸿蒙系统 wifi STA模式 34 4.3 w800 鸿蒙系统 wifi STA模式 34 4.3 w800 鸿蒙系统 wifi J描热点 40 5. w800 鸿蒙家统 连榜 dohome 平台云 43		1.1. 开发板 硬件资料分配	3
1.3 USB 接口 3 1.4. 按键 3 1.5. RGB 彩灯 3 1.6. DS18B20 温度传感器 3 1.7. 显示屏接口 3 1.8. 光敏传感器 4 1.9. w800 鸿蒙系统介绍 4 二. 开发环境搭建 7 2.1. 串口驱动的安装 7 2.1. 串口驱动的安装 7 2.2 w800 Linux开发环境搭建 11 2.3 w800 鸿蒙系统环境搭建 12 三. 、外设基础实验 15 3.1 鸿蒙系统第一个驱动程序helloworld 15 3.1 鸿蒙系统第一个驱动程序helloworld 15 3.1 鸿蒙系统第一个驱动程序helloworld 15 3.5w800 鸿蒙系统定时器实验 23 3.5w800 鸿蒙系统定时器实验 25 四、wifi 通信实验 30 4.1 w800 鴻蒙系统 wifi AP模式 30 4.2 w800 鴻蒙系统 wifi STA模式 30 4.2 w800 鴻蒙系统 wifi STA模式 30 4.3 w800 鸿蒙系统 wifi STA模式 34 4.3 w800 鸿蒙系统 wifi JA模式 34 4.3 w800 鸿蒙系统 wifi JH截点 40 万、 w800 鸿蒙系统 详持 dohome 平台云 43		1.2 供电	3
1.4. 按键 3 1.5. RGB 彩灯 3 1.6. DS18B20 温度传感器 3 1.7. 显示屏接口 3 1.8. 光敏传感器 4 1.9. w800 鸿蒙系统介绍 4 二. 开发环境搭建. 7 2.1. 串口驱动的安装. 7 2.2. w800 Linux开发环境搭建. 11 2.3 w800 鸿蒙系统环境搭建. 12 三. 、外设基础实验. 15 3.1 鸿蒙系统第一个驱动程序helloworld 15 3.1 鸿蒙系统第一个驱动程序helloworld 15 3.1 鸿蒙系统第一个驱动程序helloworld 15 3.4 w800 鸿蒙系统定时器实验 25 四、wifi 通信实验 30 4.1 w800 鸿蒙系统 wifi AP模式 30 4.2 w800 鸿蒙系统 wifi STA模式 30 4.3 w800 鸿蒙系统 wifi STA模式 30 4.3 w800 鸿蒙系统 wifi STA模式 30 4.4 w800 鸿蒙系统 wifi扫描热点 40 T、 w800 鸿蒙系统 wifi扫描热点 40		1.3 USB 接口	3
1.5. RGB 彩灯 3 1.6. DS18B20 温度传感器 3 1.7. 显示屏接口 3 1.8. 光敏传感器 4 1.9. w800 鸿蒙系统介绍 4 二. 开发环境搭建 7 2.1. 串口驱动的安装 7 2.2 w800 Linux开发环境搭建 11 2.3 w800 鸿蒙系统环境搭建 12 三、外设基础实验 15 3.1 鸿蒙系统第一个驱动程序helloworld 15 3.1 鸿蒙系统第一个驱动程序helloworld 15 3.4 w800 鸿蒙系统电口收发控制实验 23 3.5 w800 鸿蒙系统定时器实验 25 四、wifi 通信实验 30 4.1 w800 鸿蒙系统 wifi AP模式 30 4.2 w800 鸿蒙系统 wifi STA模式 30 4.3 w800 鸿蒙系统 wifi JAP模式 37 编译程序并下载到开发板,通信成功发 hello 到服务器: 40 4.4 w800 鸿蒙系统 wifi扫描热点 40 5. w800 鸿蒙素统连接 dohome 平台云 43		1.4. 按键	3
1.6. DS18B20温度传感器 3 1.7. 显示屏接口 3 1.8. 光敏传感器 4 1.9. w800 鸿蒙系统介绍 4 二. 开发环境搭建 7 2.1. 串口驱动的安装 7 2.2 w800 Linux开发环境搭建 11 2.3 w800 鸿蒙系统环境搭建 12 三. 、外设基础实验 15 3.1 鸿蒙系统第一个驱动程序helloworld 15 3.1 鸿蒙系统第一个驱动程序helloworld 15 3.4w800 鸿蒙系统串口收发控制实验 23 3.5w800 鸿蒙系统定时器实验 25 四、wifi 通信实验 30 4.1 w800 鸿蒙系统 wifi AP模式 30 4.2 w800 鸿蒙系统 wifi STA模式 34 4.3 w800 鸿蒙系统 wifi STA模式 34 4.3 w800 鸿蒙系统 wifi JT描热点 40 五、w800 鸿蒙系统 wifi JT描热点 40 五、w800 鸿蒙系统 wifi JT描热点 40 五、w800 鸿蒙系统 wifi JT描热点 40		1.5. RGB 彩灯	3
1.7. 显示屏接口		1.6. DS18B20温度传感器	3
1.8. 光敏传感器 4 1.9. w800 鸿蒙系统介绍 4 二. 开发环境搭建 7 2.1. 串口驱动的安装 7 2.2 w800 Linux开发环境搭建 11 2.3 w800 鸿蒙系统环境搭建 12 三. 外设基础实验 15 3.1 鸿蒙系统第一个驱动程序helloworld 15 3.4 w800 鸿蒙系统定时器实验 23 3.5 w800 鸿蒙系统定时器实验 25 四、wifi 通信实验 30 4.1 w800 鸿蒙系统 wifi AP模式 30 4.1 w800 鸿蒙系统 wifi STA模式 30 4.2 w800 鸿蒙系统 wifi STA模式 34 4.3 w800 鸿蒙系统 rCP客户端通信 37 编译程序并下载到开发板,通信成功发 hello 到服务器: 40 五、w800 鸿蒙系统 wifi扫描热点 40 五、w800 鸿蒙系统 按 dohome 平台云 43		1.7. 显示屏接口	3
1.9. w800 鸿蒙系统介绍 4 二. 开发环境搭建. 7 2.1. 串口驱动的安装. 7 2.2 w800 Linux开发环境搭建. 11 2.3 w800 鸿蒙系统环境搭建. 12 三. 、外设基础实验. 15 3.1 鸿蒙系统第一个驱动程序helloworld. 15 3.1 鸿蒙系统第一个驱动程序helloworld. 15 3.4w800 鸿蒙系统串口收发控制实验. 23 3.5w800 鸿蒙系统串口收发控制实验. 25 四、wifi 通信实验. 30 4.1 w800 鸿蒙系统 wifi STA模式. 30 4.2 w800 鸿蒙系统 wifi STA模式. 34 4.3 w800 鸿蒙系统 TCP客户端通信. 37 编译程序并下载到开发板,通信成功发 hello 到服务器: 40 五、w800 鸿蒙系统 wifi扫描热点. 40 五、w800 鸿蒙系统连接 dohome 平台云. 43		1.8. 光敏传感器	4
 二. 开发环境搭建		1.9. w800 鸿蒙系统介绍	4
2.1. 串口驱动的安装 7 2.2 w800 Linux开发环境搭建 11 2.3 w800 鸿蒙系统环境搭建 12 三、外设基础实验 15 3.1 鸿蒙系统第一个驱动程序helloworld 15 3.1 鸿蒙系统第一个驱动程序helloworld 15 3.4w800 鸿蒙系统串口收发控制实验 23 3.5w800 鸿蒙系统定时器实验 25 四、wifi 通信实验 30 4.1 w800 鸿蒙系统 wifi AP模式 30 4.2 w800 鸿蒙系统 wifi STA模式 34 4.3 w800 鸿蒙系统 TCP客户端通信 37 编译程序并下载到开发板,通信成功发 hello 到服务器: 40 五、w800 鸿蒙系统 wifi扫描热点 40 五、w800 鸿蒙系统连接 dohome 平台云 43		开发环境搭建	. 7
2.2 w800 Linux开发环境搭建		2.1. 串口驱动的安装	7
2.3 w800 鸿蒙系统环境搭建 12 三.、外设基础实验 15 3.1 鸿蒙系统第一个驱动程序helloworld 15 3.4w800 鸿蒙系统串口收发控制实验 23 3.5w800 鸿蒙系统定时器实验 23 3.5w800 鸿蒙系统定时器实验 25 四、wifi 通信实验 30 4.1 w800 鸿蒙系统 wifi AP模式 30 4.2 w800 鸿蒙系统 wifi STA模式 34 4.3 w800 鸿蒙系统 wifi STA模式 34 4.3 w800 鸿蒙系统 wifi JTA模式 34 4.4 w800 鸿蒙系统 wifi JTA模式 34 4.5 w800 鸿蒙系统 wifi JTA模式 34 4.6 w800 鸿蒙系统 wifi JTA模式 34 4.7 w800 鸿蒙系统 wifi JTA 集 40 五、 w800 鸿蒙系统 jt Hello JTA Hello JTA Hello JTA Hello JTA Hello JTA Hello JTA JTA Hello JTA		2.2 w800 Linux开发环境搭建	. 11
 三、外设基础实验. 3.1 鸿蒙系统第一个驱动程序helloworld. 3.4w800鸿蒙系统串口收发控制实验. 3.5w800鸿蒙系统定时器实验. 23 3.5w800鸿蒙系统定时器实验. 25 四、wifi 通信实验. 30 4.1 w800鸿蒙系统 wifi AP模式. 30 4.2 w800鸿蒙系统 wifi STA模式. 34 4.3 w800鸿蒙系统 TCP客户端通信. 37 编译程序并下载到开发板,通信成功发 hello 到服务器: 40 4.4 w800 鸿蒙系统 wifi扫描热点. 40 五、w800 鸿蒙系统连接 dohome 平台云. 		2.3 w800 鸿蒙系统环境搭建	. 12
3.1 鸿蒙系统第一个驱动程序helloworld 15 3.4w800 鸿蒙系统串口收发控制实验 23 3.5w800 鸿蒙系统定时器实验 25 四、wifi 通信实验 30 4.1 w800 鸿蒙系统 wifi AP模式 30 4.2 w800 鸿蒙系统 wifi STA模式 34 4.3 w800 鸿蒙系统 TCP客户端通信 37 编译程序并下载到开发板,通信成功发 hello 到服务器: 40 4.4 w800 鸿蒙系统 wifi扫描热点 40 五、w800 鸿蒙系统 wifi扫描热点 43	Ξ	、外设基础实验	15
3.4w800 鸿蒙系统串口收发控制实验 23 3.5w800 鸿蒙系统定时器实验 25 四、wifi 通信实验 30 4.1 w800 鸿蒙系统 wifi AP模式 30 4.2 w800 鸿蒙系统 wifi STA模式 34 4.3 w800 鸿蒙系统 TCP客户端通信 37 编译程序并下载到开发板,通信成功发 hello 到服务器: 40 4.4 w800 鸿蒙系统 wifi扫描热点 40 五、w800 鸿蒙系统连接 dohome 平台云 43		3.1 鸿蒙系统第一个驱动程序helloworld	. 15
3.5w800 鸿蒙系统定时器实验 25 四、wifi 通信实验 30 4.1 w800 鸿蒙系统 wifi AP模式 30 4.2 w800 鸿蒙系统 wifi STA模式 34 4.3 w800 鸿蒙系统 TCP客户端通信 37 编译程序并下载到开发板,通信成功发 hello 到服务器: 40 4.4 w800 鸿蒙系统 wifi扫描热点 40 五、w800 鸿蒙系统连接 dohome 平台云 43		3.4w800鸿蒙系统串口收发控制实验	. 23
 四、wifi 通信实验		3.5w800鸿蒙系统定时器实验	. 25
4.1 w800 鸿蒙系统 wifi AP模式 30 4.2 w800 鸿蒙系统 wifi STA模式 34 4.3 w800 鸿蒙系统 TCP客户端通信 37 编译程序并下载到开发板,通信成功发 hello 到服务器: 40 4.4 w800 鸿蒙系统 wifi扫描热点 40 五、w800 鸿蒙系统连接 dohome 平台云. 43	匹	、wifi 通信实验	30
 4.2 w800 鸿蒙系统 wifi STA模式		4.1 w800 鸿蒙系统 wifi AP模式	. 30
 4.3 w800 鸿蒙系统 TCP客户端通信		4.2 w800 鸿蒙系统 wifi STA模式	. 34
编译程序并下载到开发板,通信成功发 hello 到服务器:		4.3 w800 鸿蒙系统 TCP客户端通信	. 37
4.4 w800 鸿蒙系统 wifi扫描热点		编译程序并下载到开发板,通信成功发 hello 到服务器:	. 40
五、w800 鸿蒙系统连接 dohome 平台云 43		4.4 w800 鸿蒙系统 wifi扫描热点	. 40
	Ŧ	、w800 鸿蒙系统连接 dohome 平台云	43



-、 w800开发套件及鸿蒙系统介绍

源码下载:链接:https://pan.baidu.com/s/1q2asbei_QsZs867eSIVxfw 提取码: z53u

1.1. 开发板硬件资料分配

w800 的 I0 口和外接设备的连接关系,其中外接设备包括:显示屏、按键、RGB 彩灯、温度传感器、光敏传感器、USB 串口。通过板上的拨码开关可以控制板上不可插拔的设备,板上不可插拔的设备是指:彩灯、温度传感器、光敏传感器和 USB 口。

1.2 供电

开发板可以通过 USB 接口。板上集成有AMS1117-3.3V 稳压芯片,把输入的5V 电压 稳压到3.3V, 用于给板上的设备和模块都是供电。

1.3 USB 接口

USB 接口有三种功能:

- ●供电:只要插上USB 线到USB 接口即可。
- 下载: USB 下载会有单独的一节说明。
- 串口调试: 串口调试在实验中有说明。

注: 当使用 USB 接口进行调试或者程序下载的时候, 拨码 开关 5 和 6 必须打开。

1.4. 按键

开发板一个复位键用于复位 W800 系统。点击下载软件复位后会进入下载模式。

1.5. RGB 彩灯

W800的底板上集成了一个共阳极的彩灯,彩灯的阴极通过拨码开关后连接到w800的IO 口(PB2、 PB7 和 PB11),控制IO 口为低电平时,对应的灯会亮起来。要使用彩灯,必须打开拨码开关的234。

1.6. DS18B20 温度传感器

W800的底板上集成有一个DS18B20温度传感器,通过拨码开关后连接到W800的PB3口,W800通过PB3可以读出当前的温度。要使用DS18B20传感器,须打开拨码开关的1。

1.7. 显示屏接口

开发板支持两种显示屏: 1.44 寸 TFT 彩屏、0.96 寸 OLED 屏。 这 2 种显示屏都是 4 线 SPI 接口。注: 底板上显示屏接口边上都有丝印,插显示屏的时候,一定要注意显示屏方向 要正确,电源标志要 对应上,否则会烧坏显示屏。



1.8. 光敏传感器

W800通过光敏传感器可以采集环境光线的强弱,W800的PA1通过拨码开关后连接到光 敏传感器 上,W800使用ADC功能可以读取当前光线的强弱。要使用光敏传感器功能,必 须打开拨码开关的7。

1.9. w800 鸿蒙系统介绍

HarmonyOS 是一款"面向未来"、面向全场景(移动办公、运动健康、社交通信、媒体娱乐等)的分布式操作系统。在传统的单设备系统能力的基础上,HarmonyOS 提出了基于同一套系统能力、适配多种终端形态的分布式理念,能够支持多种终端设备。

对消费者而言,HarmonyOS 能够将生活场景中的各类终端进行能力整合,可以实现不同的终端设备之间的快速连接、能力互助、资源共享,匹配合适的设备、提供流畅的全场景体验。

对应用开发者而言,HarmonyOS 采用了多种分布式技术,使得应用程序的开发实现与 不同终端设备的形态差异无关。这能够让开发者聚焦上层业务逻辑,更加便捷、高效地开 发应用。

对设备开发者而言,HarmonyOS采用了组件化的设计方案,可以根据设备的资源能力和业务特征进行灵活裁剪,满足不同形态的终端设备对于操作系统的要求。

HarmonyOS 提供了支持多种开发语言的 API,供开发者进行应用开发。支持的开发语言 包括 Java、XML(Extensible Markup Language)、C/C++、JS(JavaScript)、CSS

(Cascading Style Sheets) 和 HML (HarmonyOS Markup Language)。

HarmonyOS 整体遵从分层设计,从下向上依次为:内核层、系统服务层、框架层和应 用层。系统功能按照"系统 > 子系统 > 功能/模块"逐级展开,在多设备部署场景下,支 持根据实际需求裁剪某些非必要的子系统或功能/模块。HarmonyOS 技术架构如下所示。



http://www.doit.am https://szdoit.taobao.com/ Tel: 186 7666 2425, QQ: 114209716



HarmonyOS 官网: <u>https://device.harmonyos.com/cn/home</u> 技术特性: 硬件互助,资源共享一次开发,多端部署统一 0S,弹性部署

HarmonyOS 提供了用户程序框架、Ability 框架以及 UI 框架,支持应用开发过程中多 终端的业务逻辑和界面逻辑进行复用,能够实现应用的一次开发、多端部署,提升了跨设 备应用的开发效率。

HarmonyOS 通过组件化和小型化等设计方法,支持多种终端设备按需弹性部署,能够 适配不同类别的硬件资源和功能需求。支撑通过编译链关系去自动生成组件化的依赖关 系,形成组件树依赖图,支撑产品系统的便捷开发,降低硬件设备的开发门槛。

在搭载 HarmonyOS 的分布式终端上,可以保证"正确的人,通过正确的设备,正确地使用数据"。

通过"分布式多端协同身份认证"来保证"正确的人"。通过"在分布式终端 上构筑可信运行环境"来保证"正确的设备"。通过"分布式数据在跨终端流动的 过程中,对数据进行分类分级管理"来保证"正确

地使用数据"。在官网获取源码后,源码目录说明:



目录名	描述
applications	应用程序样例
base	基础软件服务子系统集&硬件服务子系统集
build	组件化编译、构建和配置脚本
docs	说明文档
domains	增强软件服务子系统集
drivers	驱动子系统
foundation	系统基础能力子系统集
kernel	内核子系统
prebuilts	编译器及工具链子系统
test	测试子系统
third_party	开源第三方组件
utils	常用的工具集
vendor	厂商提供的软件
build.py	编译脚本文件



二. 开发环境搭建

这一章的内容是驱动的安装和软件开发环境的搭建。

2.1. 串口驱动的安装

我们的开发板使用了 CH340G 的 USB 转串口芯片。通过 USB 线把开发板接到电脑上:接着查看电脑的设备管理器,如下图表示驱动已经正确安装,可以跳过这一节:

		×
文件(F) 操作(A) 查看(V) 帮助(H)		
⇐ ➡ ☶ 📴 📓 ☶ 🖳 💺 🗙 💿		
 ▲ 書建义 > IDE ATA/ATAPI 控制器 > 型 处理器 > 回 传感器 我这里生成的是com3,大家生成的可能是其他的 > 磁盘驱动器 串口 > 梁 存储控制器 > □ 打印队列 > □ USB-SERIAL CH340 (COM3) □ 可信病口 (COM11) > □ 计算机 > □ 监视器 	5	^
 > ■ 鍵盘 > 风 人体学输入设备 > ■ 软件设备 > ▲ 声音、视频和游戏控制器 > ● 鼠标和其他指针设备]	~

如下图, 表示要安装驱动:





安装驱动步骤一:CH340串口驱动位置: .\开发软件\USB-SERIAL CH340 Driver.rar, 解压后

如

下:



安装驱动步骤二:把开发板通过USB线接到电脑上(要打开开发板电源),提示安装驱动如

下:



安装驱动步骤三:打开电脑的设备管理器,查看串口的驱动是否已自动安装,如下图是未 安装的。





安装驱动步骤四:右键更新驱动,如下图:





安装驱动步骤五:选择第一步解压好的目录:



安装驱动步骤六:选择确定后,有可能会出现以下的提示,选择"始终安装此驱动程 序软件"。



http://www.doit.am https://szdoit.taobao.com/ Tel: 186 7666 2425, QQ: 114209716



安装驱动步骤七:安装完成后,设备管理器,如下图:

9. 设备管理器	X
文件(F) 操作(A) 查看(V) 帮助(H)	
🚧 🤿 📰 🚺 📰 👧	
a 🛃 2013-0507-2359	
▷ 🏭 DVD/CD-ROM 驱动器	
▷·· 🕞 IDE ATA/ATAPI 控制器	
▷ 🛄 处理器	
▶ → → 磁盘驱动器	
> 🤪 电池	
▲ ¹ 詳曰 (COM 租 LPT)	
USB-SERIAL CH340 (COM4)	
···· · · · · · · · · · · · · · · · · ·	
▷ 1 壘 计算机	
▶ 🌉 监视器	
, ▷·□□ 键盘	
▶□▲ 声音、视频和游戏控制器	
▷ 🕑 鼠标和其他指针设备	
▶ 員 通田串行总线控制器	

串口驱动程序安装好之后,关于串口程序下载和调试,后面会陆续讲

解。

2.2 w800 Linux 开发环境搭建

推荐使用 Ubuntu 18.04。从官网 https://occ.t-head.cn 平头哥芯片 开



放社区->技术部落->资源下载->工具,根据自己本地系统环境选择下载适用的 "800 Series Toolchain"。

工具链名称	适用系统
-i386-minilibc-.tar	32 位 linux 系统
-x86_64-minilibc-.tar	64 位 linux 系统
-mingw-minilibc-.tar	windows 系统

下载后将编译工具链解压到的某个路径下(如 opt 目录),设置编译工具链路径至环境变量,举例如下:

export PATH=\$PATH:/opt/ csky-elfabiv2-tools/bin 上述设置 完成,编译工具链即可用,可以进行接下来的编译工作。 用户也可将 工具链的路径配置写至.profile 等配置文件中达到自动配

置的目的,用户还可以在 sdk 中直接指定工具链路径。

2.3 w800 鸿蒙系统环境搭建

安装 gn、

ninja wget 🎽

https://repo.huaweicloud.com/harmonyos/compiler/gn/1523/linux/g

<u>n.1523.tar</u>

wget

https://repo.huaweicloud.com/harmonyos/compiler/ninja/1.9.0/linux

/ninja.1.9.0.tar

```
tar -xvf gn.1523.tar -C ^{\sim}/
```

```
tar -xvf ninja.1.9.0.tar -C
```

~/ sudo vim ~/.bashrc

export PATH=~/gn:\$PATH



export PATH=~/ninja:\$PATH

source ~/. bashrc

进入代码 hhi_d801_v1.0 根目录,编译命

令: python build.py w800

🗩 🗊 root@book-virtual-machine: /opt/hhi_d801_v1.0

make[3]: Entering directory '/opt/hhi_d801_v1.0/vendor/winnermicro/w800/src/app/ nDNS/mDNSCore make[3]: Leaving directory '/opt/hhi_d801_v1.0/vendor/winnermicro/w800/src/app/m DNS/mDNSCore' make[3]: Entering directory '/opt/hhi_d801_v1.0/vendor/winnermicro/w800/src/app/ mDNS/mDNSPosix' make[3]: Leaving directory '/opt/hhi_d801_v1.0/vendor/winnermicro/w800/src/app/m DNS/mDNSPosix' make[2]: Leaving directory '/opt/hhi_d801_v1.0/vendor/winnermicro/w800/src/app/m DNS make[1]: Leaving directory '/opt/hhi_d801_v1.0/vendor/winnermicro/w800/src/app' libs has been updated. INK w800.elf)BJCOPY w800.bin generate normal image completed. compress binary completed. generate compressed image completed. ouild finished! Build success! [152/152] STAMP obj/vendor/winnermicro/w800/build w800 sdk.stamp ohos w800 build success!

完成编译链接后,w800 固件会生成,固件位于 out/w800/ 目录下,生成文件有:

w800.fls: 串口烧录 w800_ota.img: OTA 升级

w800.map: map 文件

接下来介绍如何把固件烧录到 w800 开发板,以Windows 环境的烧录为例:打开软件目录 ThingsTurn_Serial_Tool_V1.8.0.0,双击

ThingsTurn_Serial_Tool.exe.



查看				
电脑 > 本地磁盘 (E:) > W800 > Thir	ngsTurn_Serial_Tool_V1.8.0.0 >	ٽ ~	搜索"ThingsTurn_Serial_Tool	
名称 个	修改日期	类型	大小	
📒 download	2020/11/25 19:03	文件夹		
📙 gain	2020/6/6 7:00	文件夹		
📜 history	2020/11/25 19:03	文件夹		
📜 img	2020/6/6 7:00	文件夹		
📜 script	2020/6/6 7:11	文件夹		
ImageViewer.dll	2019/3/30 21:10	应用程序扩展	53 KB	
Newtonsoft.Json.dll	2016/5/16 11:35	应用程序扩展	399 KB	
README.txt	2020/6/6 7:27	文本文档	7 KB	
serial_tool.cfg	2020/11/25 19:03	CFG 文件	2 KB	
SimpleUpdater.dll	2017/8/10 10:43	应用程序扩展	378 KB	
🖹 SimpleUpdater.xml	2017/8/10 10:43	XML 文档	491 KB	
🗟 template.dll	2019/12/27 17:36	应用程序扩展	2,648 KB	
ThingsTurn_Serial_Tool.exe	2020/6/5 15:57	应用程序	944 KB	

打开串口,波特率115200,选择固件文件w800.fls后点击下载,然后 按一下开发板的复位键开始下载。下载完成后复位板子运行程序。

Start Download w80 start connect devi	0(V) 00.fls Lee deviceCCC	引脚分布图	引脚分布图
sync success, chip Wi-Fi MAC: 286DCD40	: w800 CEBF5: BT MAC:FFFFFFFFFFF	嵌入式Wi-Fi/BT/BLE芯片	<u>TB-08 W800 开发板</u>
MAC EEOR! MAC EEOR! MAC EEOR! End Sync, Spend 5. old gain: FF FF FF FF FF FF FF FF	9987163 Seconds FF FF FF FF FF FF FF FF FF FF FF FF FF	W600	W601
change baud to 200 Start Load	PF FF FF FF FF FF FF	引脚分布图	引脚分布图
	25% ¥	嵌入式 Wi-Fi 芯片 W600	嵌入式 Wi-Fi 芯片 W
端口号 COM7 ~	打开串口 清空接收 🗆 接收时间	字体设置 □ HEX显示	隐藏面板
波特率 115200 👻	复位设备 保存接收 □ 自动换行	背景设置 ☑ 切换串口	
数据位 8 ~ 校验位 None ~	型号: Unknown ~ 速率: 2Mbps ~] 连 固件: E:\\\800\程序源码\wm_sdk_w800\bin\\%8	续下载 □ 擦除Flash 300\w800.fls 取消	i 打包
停止位 One 🗸 🗸	□ 定时发送 1000 ms/次 □ 发送新行	□ HEX发送 □ 发送文件	
流控制 None ~	发送 t-ble-demo-on		
COM7 Closed	Received: 120	Sent: 0	2020-11-25 21

鸿蒙系统启动:



http://www.doit.am https://szdoit.taobao.com/ Tel: 186 7666 2425, QQ: 114209716

and DownLoad				^	罗大华汉区	双広切丈 P	私/リズ リロー・ハニハ 115355
00 00:00:00 00 00:00:00 (count:3). 00 00:00 faskPool:0x2 00 00:00:00 faskPool:0x2 00 00:00:00 faskPool:0x2 00 00:00:00 faskPool:0x2	0 92 D 0 92 D 0 92 I 0 92 I 00123c 0 92 I 0012a3 0 92 I 0012bf 0 160	0/HIVIEW: hi: 0/HIVIEW: lo; 1/SAMGR: Boot 1/SAMGR: Init 8 1/SAMGR: Init 8 1/SAMGR: Init 1 1/SAMGR: Init	log init su g limit init tstrap core t service:0: t service:0: t service:0:	ccess. t success. services x8193b8f x8193bb1 x8193cb9 0x8193cb9	W 引脚	800 分布图	TB-08 引脚分布图
time: Oms> 10 00:00:00 time: Oms> 10 00:00:00 00 00:00:00 time: Oms> 10 00:00:00 tystem servi 10 00:00:00 pplication 10 00:00:00 md applicat	succes 0 76 I succes 0 244 0 244 succes 0 244 ces! 0 76 I servic 0 76 I ion se 0 76 I	s! 1/SAMGR: Init s! D O/HIVIEW: h: I 1/SAMGR: Init 1/SAMGR: Init 1/SAMGR: Boot es(count:0). 1/SAMGR: Init rvices! 1/SAMGR: Boot	t service 0: iview init : it service d itialized a tstrap syste tialized al tstrap dyma	x8193b8f success. 0x8193cb9 11 core em and 1 system mic ~	<u>►-iwf> 茶ă</u> (明尼 -iw 5.6ă	600 分布图 fi 芯片 W600	TB-08 W800 井发板 W601 引脚分布图 ^{嵌入式 Wi-Fi 芯片 W601}
端口号 COM(7 皮特率 115200	~	关闭串口	清空接收保存接收	 □ 接收时间 □ 自动换行 	字体设置	□ HEX显示 ☑ 切换串口	隐藏面板
数据位 8	Ŷ	Det Conknown	a ∨ 速率:	2Mbps ∨□连	卖下载 □ 擦除	Flash	
交验位 None	\sim	固件: E:\w800	鸿蒙系统\w800).fls] 卜载	115
亭止位 One	×	□ 定时发送	1000 ms///	欠 □ 发送新行	□ HEX发送 □	2 发送文件	
紀控制 None	~	发送	t-ble-demo-or	n -			
M7 Opend		Rece	ived: 2052		Sent: 0		2020-12-12 15:54:

三.、外设基础实验

说明:在这一章实验里,我们以打包好例程开始学习讲解。

3.1 鸿蒙系统第一个驱动程序helloworld

编写第一个程序 helloworld, 首先我们在 app 目录新建一个

myhelloworld.c文件。



接下来编写 helloworld 任务程序,调用 SYS_RUN 来执行:





```
static void *helloworldTask(const char *arg)
{
    (void)arg;
   while (1) {
        printf("hello world!\n");
        osDelay(100);
static void myhelloworld(void)
    osThreadAttr_t attr;
    attr.name = "myhelloworldTask";
    attr.attr_bits = 0U;
    attr.cb_mem = NULL;
    attr.cb_size = 0U;
    attr.stack_mem = NULL;
    attr.stack_size = TASK_STACK_SIZE;
    attr.priority = TASK_PRIO;
    if (osThreadNew((osThreadFunc_t)helloworldTask, NULL, &attr) == NUL
L) {
        printf("Falied to create Task!\n");
SYS_RUN(myhelloworld); //if test add it
```

在同目录新建一个 BUILD.gn,加入以下内容:



http://www.doit.am https://szdoit.taobao.com/ Tel: 186 7666 2425, QQ: 114209716

)	资源管理器 …	≣ BUILD.gn ×
	> 打开的编辑器	applications > sample > wifi-iot > app > iothardware > 🗧 BUILD.gn
5	∨ HHI_D801_V1.0	1 # Copyright (c) 2020 Huawei Device Co., Ltd.
r.	\checkmark applications \ sample \ wifi-iot	2 # Licensed under the Apache License, Version 2.0 (the "License");
		3 # you may not use this file except in compliance with the License.
¢	> aht20	4 # You may obtain a copy of the License at
	V istherduara	5 #
		6 # <pre>http://www.apache.org/licenses/LICENSE-2.0</pre>
>	≡ BUILD.gn	7 #
	C led_example.c	8 # Unless required by applicable law or agreed to in writing, software
כ	C lowpower_example.c	9 # distributed under the License is distributed on an "AS IS" BASIS,
	C myhelloworld.c	# Wilhout WARKANILES OK CONDITIONS OF ANY KIND, either express or implied.
	C uart_example.c	11 # See the License for the specific language governing permissions and
	> network	12 # limitations under the License.
	> samgr	13 14 static library("mybelloworld") {
	> startun	
	> startup	16 myhelloworld c"
	≡ BUILD.gn	18
	1 LICENSE	19 include dirs = [
	> base	20 "//utils/native/lite/include",
	> build	<pre>21 "//kernel/liteos_m/components/cmsis/2.0",</pre>
	> domains	22 "//base/iot_hardware/interfaces/kits/wifiiot_lite",
	> foundation	23]
	> kernel	24 }
	> out	

在 app 同目录下的 BUILD. gn 加入我们需要编译运行的程序 helloworld。

HHI_D801_V1.0	1 # Copyright (c) 2020 Huawei Device Co., Ltd.
✓ applications \ sample \ wifi-iot	2 # Licensed under the Apache License, Version 2.0 (the "License");
	3 # you may not use this file except in compliance with the License.
> abt20	4 # You may obtain a copy of the License at
	5 #
	6 # <pre>http://www.apache.org/licenses/LICENSE-2.0</pre>
≣ BUILD.gn	7 #
C led_example.c	8 # Unless required by applicable law or agreed to in writing, software
C lowpower_example.c	9 # distributed under the License is distributed on an "AS IS" BASIS,
C myhelloworld.c	10 # WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
C uart example c	11 # See the License for the specific language governing permissions and
	12 # limitations under the License.
> Tietwork	13
> samgr	<pre>14 import("//build/lite/config/component/lite_component.gni")</pre>
> startup	15
> wifitest	16 lite_component("app") {
🗉 BUILD.gn 🛛 🛑 📥	17 features = [
	18 "iothardware:myhelloworld",
> base	19
> build	20 }
	21
> foundation	
> kernel	

编译运行结果:



🛑 星通智联串口调试助	手 V1.8.1.0 更多信息可访问:https://www	.w600.fun	- 🗆 X
接收 TaskPool: 0x20012c5 00 00:00:00 0 92 I TaskPool: 0x20012e1 00 00:00:00 0 244 : <time: oms=""> succes 00 00:00:00 0 160 <time: oms=""> succes 00 00:00:00 0 72 I <time: oms=""> succes 00 00:00:00 0 72 I <time: oms=""> succes</time:></time:></time:></time:>	4 1/SAMGR: Init service: 0x8193e29 4 I 1/SAMGR: Init service 0x8193d21 s! 0/HIVIEW: hiview init success. 1/SAMGR: Init service 0x8193e29 s! 1/SAMGR: Initialized all core		
00 00:00:00 0 160 : application servic 300 00:00:00 0 160 : and application se 00 00:00:00 0 160 : registered service: hello world! hello world! hello world! hello world!	I I/SAMAR: Bootstrap system and es(count:0). I I/SAMGR: Initialized all system rvices! I 1/SAMGR: Bootstrap dynamic s(count:0).	W600 引脚分布图	W601 引脚分布图
端口号 COM7 ~ 波特率 115200 ~	关闭串口 清空接收 接收时 夏位设备 保存接收 自动换	 间 字体设置 □ HEX显示 行 背景设置 ☑ 切换串口 	隐藏面板
数据位 8 · · ·	型号: Unknown > 速率: 210bps > □	连续下载□擦除Flash 下载	: 打包
校验ID Mone V	□□F: [1. \woodysgar, 系统 (wood). fis	」 └・・・」 └─── 行 □ HEX发送 □ 发送文件	
流控制 None 🗸	发送 t-ble-demo-on		
OM7 Opend	Received: 125731	Sent: 0	2020-12-12 20:19:06

3.2 w800 鸿蒙系统 GPIO 控制 LED 灯实验

要使用彩灯,必须打开拨码开关的234。本节内容我们实现 RGB 每隔 1 秒进行闪烁。W800 的底板上集成了一个共阳极的彩灯,彩灯的阴极通过拨码 开关后连接到 w800 的 IO 口(PB2、 PB7 和 PB11),控制 IO 口为低电平时,对应的灯会亮起来。这里我们以 PB2 红灯为例子实现控制。

```
首先创建一个 LED 任务。
```





http://www.doit.am https://szdoit.taobao.com/ Tel: 186 7666 2425, QQ: 114209716



接下来编写任务程序,输出高电平熄灭,输出低电平点亮。

```
enum LedState g_ledState = LED_SPARK;
static void *GpioTask(const char *arg)
{
    (void)arg;
   while (1) {
         switch (g_ledState)
           { case LED_ON:
                printf(" LED_ON! \n");
                GpioSetOutputVal(WIFI_IOT_GPIO_PB_02, WIFI_IOT_GPIO_ATT
R_PULLLOW);
                osDelay(100);
                break;
            case LED_OFF:
                printf(" LED_OFF! \n");
                GpioSetOutputVal(WIFI IOT GPIO PB 02, WIFI IOT GPIO ATT
R PULLHIGH);
                osDelay(100);
                break;
            case LED_SPARK:
            printf(" LED SPARK! \n");
                GpioSetOutputVal(WIFI_IOT_GPIO_PB_02, WIFI_IOT_GPIO_ATT
R PULLLOW);
                osDelay(100);
                GpioSetOutputVal(WIFI_IOT_GPIO_PB_02, WIFI_IOT_GPIO_ATT
R PULLHIGH);
                osDelay(100);
                break:
            default:
                osDelay(100);
                break;
```



http://www.doit.am https://szdoit.taobao.com/ Tel: 186 7666 2425, QQ: 114209716



编译选项添加如下内容:

5		
	> OPEN EDITORS	applications > sample > wifi-iot > app > iothardware > 🖺 BUILD.gn
С	∨ HHI_D801_V1.0	9 # distributed under the License is distributed on an "AS IS" BASIS, usleep
	✓ applications/sample/wifi-iot	10 # WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implie
_	∼ app	11 # See the License for the specific language governing permissions and osuelay
2	> aht20	12 # limitations under the License.
	✓ iothardware	13 14 static library("mybelloworld") {
>	🗉 BUILD.gn	15 sources = [
	C led_example.c	16 "myhelloworld.c"
n	C lowpower_example.c	
Ľ	C myhelloworld.c	
	C uart example.c	19 include_dirs = [
	> network	20 "//utils/native/lite/include",
	> samgr	21 "//kernel/liteos_m/components/(msi5/2.0",
	✓ startup	
	≣ BUILD.an	24 }
	> wifitest	25
	≣ BUILD.an	26 static_library("led_example") {
		27 sources = [
	> base	28 "led_example.c"
	> build	29]
	✓ domains/int/link	30 include dirs = [
	> demolink	32 "//utils/native/lite/include",
	✓ libbuild	33 "//kernel/liteos_m/components/cmsis/2.0",
		34 "//base/iot_hardware/interfaces/kits/wifiiot_lite",
	C demosdk c	
	C demosdk h	36 }
2		applications > sample > wifi-int > app > E RUILD op
		1 # Copyright (c) 2020 Hugwei Device Co., Ltd.
Ś.	applications / sample / wifi-iot	<pre>2 # Licensed under the Apache License, Version 2.0 (the "License");</pre>
		3 # you may not use this file except in compliance with the License. osDelay
8	> abt20	4 # You may obtain a copy of the License at
	viothardware	
>	E BUILD on	b # <u>nllp://www.apacne.org/llcenses/LlCENSE-2.0</u>
		8 # Unless required by applicable law or agreed to in writing, software
		9 # distributed under the License is distributed on an "AS IS" BASIS,
	C mybelloworld c	10 # WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
	C ust example c	11 # See the License for the specific language governing permissions and
	> network	12 # limitations under the License.
	> samor	13
		14 Import(//build/lite/config/component/lite_component.gni")
		15 lite component("app") {
	- BUILD.gi	17 features = []
	F RIM D as	18 "iothardware:myhelloworld",
	BUILD.gli	19 "iothardware:led_example",
	A LICENSE	20
	> base	21 }
	✓ domains/iot/link	
	> demolink	

程序下载到开发板,可以看到红灯周期性的闪烁。



💼 星诵智联串口调试助:	手 V1.8.1.0 更多信息可访问: https://www.w6	00.fun	- П X
接收 TaskPool: 0x20013030		多文本发送 发送历史 下	载历史 引脚定义 在线资源
135k7001.00200 0000:00000 72 I <time:< td=""> 0ms> success 00000:00000 244 I <time:< td=""> 0ms> success 00000:00:000 156 I 0000:00:000 156 I 0000:00:000 156 I 0000:00:000 156 I system services! 0000:00:000 244 I application services! 0000:00:000 244 I and application services! hello world! hello world!</time:<></time:<>	1/SAMGR: Init service 0x8194011 1/SAMGR: Init service 0x8193fef 0/O/HIVIEW: hiview init success. 1/SAMGR: Init service 0x8194119 1/SAMGR: Initialized all core 1/SAMGR: Bootstrap system and es(count: 0). 1/SAMGR: Initialized all system rvices! 1/SAMGR: Bootstrap dynamic s(count: 0).	W800 引脚分布图 武法式WI-FI/DILE式正式 W600 引脚分布图	へ TB-08 引脚分布图 ^{TB-08 W800 开发板} W601 引脚分布图
端口号 COM7 ~	关闭串口 清空接收 □ 接收时间	字体设置 □ ਮॻऱ⊥显示	隐藏面板
波特率 115200 👻	夏位设备 保存接收 □ 自动换行	背景设置 ☑ 切换串口	
数据位 8 🛛 🗸	型号: Unknown 🗸 速率: 21Mbps 🗸 🗆 连	续下载□擦除Flash _{下封}	打句
校验位 None 🗸 🗸	固件: E:\w800鸿蒙系统\w800.fls		116
停止位 One V	□ 定时发送 1000 ms/次 □ 发送新行	□ HEX发送 □ 发送文件	
COM7 Opend	Received: 446359	Sent: 0	2020-12-12 21:43:44
		Instantion in portugi	· · · · · · · · · · · · · · · · · · ·

3.3 w800 鸿蒙系统低功耗休眠模式

在某些时候我们设计的产品可能不具备持久供电的环境,那通常会采用 锂电池、干电池一类的轻便型的非持久性电源。当遇到这种情况时,产品的 续航能力可能就会成用户评估产品的一个重要指标,加大电池容量当然是最 为直接的方案,但是这也意味着提高产品的生产成本。那增加产品续航能力 的另一个方案就是原自产品自身——降低不必要的能源消耗。

首先建立低功耗的任务:





```
attr.stack_mem = NULL;
   attr.stack size = LP TASK STACK SIZE;
   attr.priority = LP_TASK_PRIO;
   if (osThreadNew((osThreadFunc_t)LowpowerTask, NULL, &attr) == NULL)
        printf("[LowpowerExample] Falied to create LowpowerTask!\n");
static void *LowpowerTask(const char *arg)
    (void)arg;
   static int i = 0;
   printf("enter LowpowerTask\n");
   while (1) {
       if (i == 0)
            { i++;
            LpcSetType(LIGHT_SLEEP);
       else
            LpcSetType(DEEP_SLEEP);
       osDelay(500);
   }
```

配置编译选后编译下载程序,程序进入低功耗后停止串口打印,停



3.4w800 鸿蒙系统串口收发控制实验

本例我们实现 w800 鸿蒙系统串口收发控制实

验。 建立一个串口收发的任务:

```
static void UartExampleEntry(void)
{
    osThreadAttr_t attr;
    attr.name = "UartTask";
    attr.attr_bits = 0U;
    attr.cb_mem = NULL;
    attr.cb_size = 0U;
    attr.stack_mem = NULL;
    attr.stack_size = UART_TASK_STACK_SIZE;
    attr.priority = UART_TASK_PRIO;
    printf("[UartExample] create UartTask!\n");
    if (osThreadNew((osThreadFunc_t)UartTask, NULL, &attr) == NULL)
        { printf("[UartExample] Falied to create UartTask!\n");
    }
}
```

我们配置为串口 0, 串口 0 接收到数据并原样返回数据:

```
static void *UartTask(const char *arg)
{
    (void)arg;
   unsigned int ret;
   int len;
   unsigned char uartWriteBuff[] = "hello uart!";
   WifiIotUartAttribute uartAttr;
   uartAttr.baudRate = INIT BAUD RATE,
   uartAttr.dataBits = WIFI_IOT_UART_DATA_BIT_8;
   uartAttr.stopBits = WIFI_IOT_UART_STOP_BIT_1;
   uartAttr.parity = WIFI_IOT_UART_PARITY_NONE;
   uartAttr.pad ='M';
   WifiIotUartExtraAttr extraAttr;
   (void)memset_s(&extraAttr, sizeof(WifiIotUartExtraAttr), 0,
sizeof( WifiIotUartExtraAttr));
   extraAttr.txFifoLine = WIFI_IOT_FIF0_LINE_ONE_EIGHT;
```



```
extraAttr.rxFifoLine = WIFI_IOT_FIF0_LINE_SEVEN_EIGHTS;
    extraAttr.flowFifoLine = WIFI IOT FIFO LINE ONE EIGHT;
    extraAttr.txBlock = WIFI_IOT_UART_BLOCK_STATE_BLOCK;
    extraAttr.rxBlock = WIFI_IOT_UART_BLOCK_STATE_BLOCK;
    extraAttr.txBufSize = TEST_BUF_SIZE;
    extraAttr.rxBufSize = TEST_BUF_SIZE;
    extraAttr.txUseDma = WIFI IOT UART USE DMA;
    extraAttr.rxUseDma = WIFI_IOT_UART_USE_DMA;
    ret = UartDeinit(WIFI_IOT_UART_IDX_0);
    ret = UartInit(WIFI_IOT_UART_IDX_0, &uartAttr, &extraAttr);
    len = UartWrite(WIFI_IOT_UART_IDX_0, uartWriteBuff, sizeof(uartWrit
eBuff));
    len = UartWrite(WIFI IOT UART IDX 0, uartWriteBuff, sizeof(uartWrit
eBuff));
   while (1) {
        len = UartRead(WIFI_IOT_UART_IDX_0, uartReadBuff, sizeof(uartRe
adBuff));
        if(len > 0)
            UartWrite(WIFI_IOT_UART_IDX_0, uartReadBuff, len);
    return NULL;
```

配 BUILD.gn:

> OPEN EDITORS	applications > sample > wifi-iot > app > iothardware > \Xi BUILD.gn
<pre>> HHI_D801_V1.0 > applications/sample/wifi-iot > app > aht20 > iothardware E BUILD.gn C led_example.c C lowpower_example.c C lowpower_example.c C uart_example.c > network > samgr > startup E BUILD.gn > wifitest E BUILD.gn \$ wifitest E BUILD.gn \$ uicENSE > base </pre>	<pre># Unless required by applicable law or agreed to in writing, software # distributed under the License is distributed on an "AS IS" BASIS, # WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied # See the License for the specific language governing permissions and # limitations under the License. static_library("uart_example") { sources = ["uart_example.c" l include_dirs = []</pre>

http://www.doit.am https://szdoit.taobao.com/ Tel: 186 7666 2425, QQ: 114209716

编译下载到开发板,串口随便输数据并发送,数据会原样返回:

🚭 星通智联串口调试助	手 V1.8.1.0 更多信息可访问: http	s://www.w600.fun	- 🗆 X
接收 service:0x8193eb 00 00:00:00 0 92 service:0x8193fb 00 00:00:00 0 24 0x8193eb1 <time: 00 00:00:00 0 16 0x8193e8f <time: 00 00:00:00 0 72 success. 00 00:00:00 0 72 0x8193fb9 <time: 00 00:00:00 0 72 0x8193fb9 <time: 00 00:00:00 0 16 and application 00 00:00:00 0 16 system and appli 00 00:00:00 16 dynamic regista</time: </time: </time: </time: 	1 TaskPool:0x20013454 I 1/SAMGR: Init 9 TaskPool:0x20013614 4 I 1/SAMGR: Init service Oms> success! D 0/HIVIEW: hiview init I 1/SAMGR: Init service Oms> success! I 0/HIVIEW: hiview init I 1/SAMGR: Initialized ices! O I 1/SAMGR: Bootstrap st services(count:0). O I 1/SAMGR: Initialized cation services! O I 1/SAMGR: Bootstrap ed services(count:0). 鳴蒙系统w800鸿蒙系统w800鸿	 タ文本发送 发送历史 シ文本发送 发送历史 W800 引脚分布图 磁入式Wi-Fi/BT/BLE芯片 W600 引脚分布图 (新入式 Wi-Fi 芯片 W600 	下載历史 引脚定义 在线资源 TB-08 引脚分布图 TB-08 W800 开发板 W601 引脚分布图 引脚分布图 小前脚分布图
端口号 COM7 ~	关闭串口	接收时间 字体设置 □ HEX显示	隐藏面板
波特率 115200 👻	夏位设备保存接收	〕自动换行 背景设置	
数据位 8	型号: Unknown v 速率: 2Mbp	ps ∨□ 连续下载□ 擦除Flash	ま 打句
校验位 None 🗸 🗸	固件: E:\w800鸿蒙系统\w800.fls	· · · · · · · · · · · · · · · · · · ·	
停止位 One 🗸 🗸	□ 定时发送 1000 ms/次 □	〕发送新行 □ HEX发送 □ 发送文件	
流控制 None 🗸 🗸	发送 #800鸿蒙系统		
Send OK !	Received: 1065	Sept 22	2020-12-13 11:47:32

3.5w800 鸿蒙系统定时器实验

这节我们使用鸿蒙系统定时1秒时间,首先建立一个定时器任务:

在任务中启动定时器,注册定时回调函数,并设置时间为1秒:

定时回调函数:

编译并运行程序,每隔1秒进行打印:

🛑 星通智联串口调试助	手 V1.8.1.0 更多信息可访问: https:	://www.w600.fun	- 🗆 X
▲ 建智铁串口场低的 →	4 I 1/SAMGR: Init service Oms) success! 0 I 1/SAMGR: Init service Oms) success! D 0/HIVIEW: hiview init I 1/SAMGR: Init service Oms) success! I 1/SAMGR: Initialized a ices! 0 I 1/SAMGR: Bootstrap sy services(count:0). 0 I 1/SAMGR: Initialized cation services! 0 I 1/SAMGR: Bootstrap ed services(count:0).	クマ本发送 发送历史 1 シ文本发送 发送历史 1 W800 引脚分布图 強入式Wi-Fi/BULK式片 W600 引脚分布图	TB-08 W800 开发板 W601 引脚分布图
端口号 COM7 ~ 波特率 115200 ~ 数据位 8 ~ 校验位 None ~ 停止位 One ~ 流控制 None ~	关闭串口 清空接收 夏位设备 保存接收 型号: Unknown 速率: 200 ps 固件: E: \w800 30 家系统 \w800. fls こおお发送 1000 ms/次 1000 发送 w800 30 家系统 1000 1000	 ☆ 嵌入式 Wi-Fi 芯片 W600 接收时间 字体设置 □ HEX显示 自动换行 背景设置 ☑ 切换串口 s ◇ □ 连续下载 □ 擦除Flash 広 下载 太送新行 □ HEX发送 □ 发送文件 	 嵌入式 Wi-Fi 芯片 W601 隐藏面板 其 打包
COM7 Opend	Received: 39273	Sent: 0	2020-12-14 11:50:38

3.6 w800 鸿蒙系统 PWM LED 以一定频率闪烁

W800 芯片支持 5 路 PWM 接口, 有 10 个引脚可被配置为 PWM 功能,

具体如下:

PWM	W800引脚组1	W800引脚组2
PWM1	WM_IO_PB_00	WM_IO_PB_19
PWM2	WM_IO_PB_01	WM_IO_PB_20
PWM3	WM_IO_PB_02	WM_IO_PA_00
PWM4	WM_IO_PB_03	WM_IO_PA_01
PWM5	WM_IO_PA_07	WM_IO_PA_04

代码里默认把 W800 芯片的 10 个可配置为 PWM 功能的引脚分成两组: W800 引脚组 1 和 W800 引脚组 2。

1. 鸿蒙系统 PWM API 头文件

base/iot_hardware/interfaces/kits/wifiiot_lite/wifiiot_pwm.h 2. 鸿 蒙系统 PWM API 使用说明

1) unsigned int PwmInit(WifiIotPwmPort port);初始化一个 PWM 设备。

说明:参数可设置为 WIFI_IOT_PWM_PORT_PWM0 ~

WIFI_IOT_PWM_PORT_PWM4 其中之一, 对应 W800 引脚组 1 的相关引脚。如 果想使用 W800 引脚组 2 的相关引脚,可以在代码里扩展枚举变量 WifiIotPwmPort。

unsigned int PwmDeinit(WifiIotPwmPort port);释放一个 PWM 设

备。

unsigned int PwmStart(WifiIotPwmPort port, unsigned short duty, unsigned short freq);按照入参配置启动对应的 PWM 功能。 建立一个 PWM 任务:

```
static void pwm(void)
{
    osThreadAttr_t attr;
    attr.name = "pwmTask";
    attr.attr_bits = 0U;
    attr.cb_mem = NULL;
    attr.cb_size = 0U;
    attr.stack_mem = NULL;
    attr.stack_size = TASK_STACK_SIZE;
    attr.priority = TASK_PRIO;
    if (osThreadNew((osThreadFunc_t)pwmTask, NULL, &attr) == NULL)
        { printf("Falied to create Task!\n");
    }
}
```

任务中初始化 PWM,设置频率 20hz 并启动:

http://www.doit.am https://szdoit.taobao.com/ Tel: 186 7666 2425, QQ: 114209716

<pre>PwmDeinit(WIFI_IOT_PWM_PORT_PWM2);</pre>
<pre>PwmInit(WIFI_IOT_PWM_PORT_PWM2);</pre>
<pre>PwmStart(WIFI_IOT_PWM_PORT_PWM2, TEST_PWM_PORT, TEST_PWM_DUTY);</pre>
while (1) {
<pre>printf("pwmTask!\n");</pre>
osDelay(1000);
}
return NULL;

接收		多文本发送 发送历史	下载历史 引脚定义 在线资源
service:0x819400 00 00:00:00 0 15 service:0x819468 00 00:00:00 0 52 0x8194551 <time: 00 00:00:00 0 22 0x819452f <time: 00 00:00:00 0 13 success. 00 00:00:00 0 13 0x8194685 <time: 00 00:00:00 0 13 core system serv 00 00:00:00 0 22 and application</time: </time: </time: 	<pre>bill TaskFool:0x20012c94 bill TaskFool:0x20012c94 bill TaskFool:0x20012e54 bill taskFool:0x</pre>	↑ W800 引脚分布图 ^武 ★→式wi-Fi/BILE式出 PM	TB-08 引脚分布图 TB-08 W800 开发板
system and appli system and appli 00 00:00:00 22 dynamic register pwmTask!	cation services! 24 I 1/SAMGR: Bootstrap red services(count:0).	引 脚 分 布 图 * ^{嵌入式 Wi-Fi 芯片 W600}	引脚分布图
SC 50:00:00 0 22 system and appli dynamic register pwmTask! 端口号 COM7 〜	24 I I/SAMGR: Initialized al cation services! 24 I I/SAMGR: Bootstrap red services(count:0). 关闭串口 清空接收 日接	リ 	引脚分布图 ^{嵌入式 Wi-Fi 芯片 W601}
by 50:00:00 00 22, system and appli 00 00:00:00 0 22 dynamic register owmTask! 端口号 COM7 〜 波特率 115200 +	A I I/SAMGR: Initialized al cation services! A4 I 1/SAMGR: Bootstrap red services(count:0).	・ ・ ・ ・ ・ </td <td>引脚分布图</td>	引脚分布图
wystem and appli system and appli dynamic register wmTask! 端口号 COM7 ~ 波特室 115200 ~	4 I I/SAMGR: Initialized al cation services! 24 I I/SAMGR: Bootstrap red services(count:0). 美闭串ロ 清空接收 自起 夏位设备 保存接收 自起 型号: Unknown 〜 速率: 2005 、		引脚分布图
kg 60:00:00 00 22 system and appli 00 00:00:00 0 22 dynamic register mwmTask! 端口号 COM7 ~ 波特室 115200 ~ 数据位 8 ~	4 I 1/SAMGR: Initialized al cation services! 24 I 1/SAMGR: Bootstrap red services(count:0). 美闭串ロ 清空接收 自起 型号: Waknown 〜 速率: 20tbps 、 固件: E:\w8003該系統\w800.fls	り ゆり はのです 「「「「「「」」」 「「」」	引脚分布图 嵌入式 Wi-Fi 芯片 W601 院廠面板 武
kystem and appli 00 00:00:00 0 22 dynamic register owmTask! 滅特案 115200 ↓ 数据位 8 ↓ 検验位 None ↓	A I I/SAMGR: Initialized al cation services! 24 I 1/SAMGR: Bootstrap red services (count:0). 美闭串ロ 清空接收 算 夏位设备 保存接收 自起 型号: Unknown 文速率: 2Mbps 、 固件: E:\w800独蒙系统\w800.fls こ定时发送 1000 ms/次 分詞		引脚分布图 嵌入式 Wi-Fi 芯片 W601 酸藏面板 試

四、wifi 通信实验

4.1 w800 鸿蒙系统 wifi AP模式

AP 是(Wireless) AccessPoint 的缩写,即(无线)访问接入点。简 单来讲就像是无线路由器一样,设备打开后进入 AP 模式,在手机的网络 列表里面,可以搜索到类似 AP 的名字(SSID)。我们的手机和笔记本电脑 均可连接到创建的 AP热点。

http://www.doit.am https://szdoit.taobao.com/ Tel: 186 7666 2425, QQ: 114209716

WiFi的工作模式

* AP 模式: 热点模式,提供无线接入服务,允许其它无线设备接入,提供数据访问,一般的无线路由/网桥工作在该模式。

* STA 模式:类似于无线终端,本身并不接受其他设备的接入,它可以连接到 AP,一般 无线网卡即工作在该模式。

* Harmony OS 的 WiFi 相关 API 头文件位于

`foundation\communication\interfaces\kits\wifi_lite\wifiservice`目录,该 目录下有 9 个文件;

* `wifi_device.h`中定义的是 STA 模式的主要接口,例如扫描其他热点、添加热点 配置(热点名称、密码等)、连接其他热点;

* `wifi_hotspot.h`中定义的是 AP 模式的主要接口,例如设置热点信息(热点名称、密码等)、查询连接的设备列表;

* `wifi_hotsport_config.h`中定义了设置和获取当前工作在 2.4G 或者 5G 频段的 接口`SetBand`和`GetBand`;

* 另外 6 个文件中定义了上述接口相关的类型,例如扫描结果、热点配置、热点连接状态等;

创建 AP 的任务:

```
static void WifiHotspotDemo(void)
{
    osThreadAttr_t attr;
    attr.name = "WifiHotspotTask";
    attr.attr_bits = 0U;
    attr.cb_mem = NULL;
    attr.cb_size = 0U;
    attr.stack_mem = NULL;
    attr.stack_size = 10240;
    attr.priority = osPriorityNormal;
    if (osThreadNew(WifiHotspotTask, NULL, &attr) == NULL)
        { LOGI("[WifiHotspotDemo] Falied to create
        WifiHotspotTask!\n");
    }
```

任务程序中注册联网事件,设置 AP 热点名称,密码:

```
static void WifiHotspotTask(void *arg)
{
    (void)arg;
    WifiErrorCode errCode;
    HotspotConfig config = {0};
```



```
// 准备 AP 的配置参数
    strcpy(config.ssid, "HiSpark-AP");
    strcpy(config.preSharedKey, "12345678");
    config.securityType = WIFI_SEC_TYPE_PSK;
    config.band = HOTSPOT_BAND_TYPE_2G;
    config.channelNum = 7;
    osDelay(10);
   while (1) {
        LOGI("starting AP ....");
        LOGI("IsHotspotActive(): %d", IsHotspotActive());
        errCode = StartHotspot(&config);
       LOGI("StartHotspot: %d", errCode);
        LOGI("IsHotspotActive(): %d", IsHotspotActive());
        HotspotConfig result = {0};
        errCode = GetHotspotConfig(&result);
        LOGI("GetHotspotConfig: %d, %d", errCode, memcmp(&config, &resu
lt, sizeof(result)));
        const int step = 5;
        for (int timeout = 60; timeout >= 0; timeout -= step)
            { LOGI("after %d seconds, I will dsiable AP.",
            timeout); osDelay(step * 100);
        LOGI("stop AP ....");
        StopHotspot();
        LOGI("stop AP done!");
       osDelay(100);
```

配 BUILD.gn:

http://www.doit.am https://szdoit.taobao.com/ Tel: 186 7666 2425, QQ: 114209716

(k.	@snail 已连接,安全	
	<u>属性</u>	
		断开连接
(7.	CU_D2XP 安全	
(7.	HiSpark-AP 安全	
(î.	TP-LINK_57E8 安全	
(i.	TPGuest_57E8 安全	
(h	TP-LINK_AEB8 安全	

4.2 w800 鸿蒙系统 wifi STA 模式

这节我们使用 sta 模式连接路由器联网。建立联网任务:

任务中设置我们需要连接的路由器名称, 密码:

stat	tic void WifiConnectTask(void *arg)
{	
	(void)arg;
	WifiErrorCode errCode;
	WifiEvent eventListener = {
	.OnWifiConnectionChanged = OnWifiConnectionChanged,
	.OnWifiScanStateChanged = OnWifiScanStateChanged
	};
	WifiDeviceConfig apConfig = {};
	<pre>int netId = -1;</pre>
	osDelay(100);
	errCode = RegisterWifiEvent(&eventListener);
	LOGI("RegisterWifiEvent: %d", errCode);
	// setup your AP params
	<pre>strcpy(apConfig.ssid, "HIHOPE-AP");</pre>
	<pre>strcpy(apConfig.preSharedKey, "1234567890");</pre>
	<pre>apConfig.securityType = WIFI_SEC_TYPE_PSK;</pre>

http://www.doit.am https://szdoit.taobao.com/ Tel: 186 7666 2425, QQ: 114209716

```
while (1) {
        LOGI("sysTicks: %u, kernelTicks: %u", osKernelGetSysTimerCount(
), osKernelGetTickCount());
        LOGI("IsWifiActive(): %d", IsWifiActive());
       errCode = EnableWifi();
       LOGI("EnableWifi: %d", errCode);
       LOGI("IsWifiActive(): %d", IsWifiActive());
       osDelay(10);
       char buffer[32] = {0};
       unsigned char mac[6] = {0};
        errCode = GetDeviceMacAddress(mac);
       LOGI("GetDeviceMacAddress: %d, %s", errCode, FormatMacAddress(b
uffer, mac));
       g_scanDone = 0;
       errCode = Scan();
       LOGI("Scan: %d", errCode);
       LOGI("waiting for scan done...");
       while (!g_scanDone) {
           osDelay(5);
        }
        PrintScanResult();
       osDelay(100);
        errCode = AddDeviceConfig(&apConfig, &netId);
       LOGI("AddDeviceConfig: %d", errCode);
       g_connected = 0;
        errCode = ConnectTo(netId);
       LOGI("ConnectTo(%d): %d", netId, errCode);
       LOGI("waiting for connect success...");
       while (!g_connected) {
           osDelay(10);
       LOGI("g_connected: %d", g_connected);
       osDelay(100);
       // 联网业务开始
        // 这里是网络业务代码...
```


t)

深圳四博智联科技有限公司 Shenzhen Doctors of Intelligence & Technology (DOIT) http://www.doit.am https://szdoit.taobao.com/ Tel: 186 7666 2425, QQ: 114209716

	<pre>int i = 0; for (struct netif* p = netif_list; p; p = p->next, i++) { hexdump(p->name, sizeof(p->name)); LOGI("[%d]: ip = %s", i, inet_ntoa(p->ip_addr)); LOGI("[%d]: gw = %s", i, inet_ntoa(p->gw)); LOGI("[%d]: netmask = %s", i, inet_ntoa(p->netmask)); }</pre>
	<pre>// 模拟联网业务 const int step = 5; for (int timeout = 60; timeout >= 0; timeout -= step) { LOGI("after %d seconds, I will disconnect with AP.", timeou</pre>
و	osDelay(step * 100); } // 联网业务结束
	<pre>errCode = Disconnect(); // disconnect with your AP LOGI("Disconnect: %d", errCode);</pre>
	<pre>errCode = RemoveDevice(netId); // remove AP config LOGI("RemoveDevice: %d", errCode);</pre>
	<pre>errCode = DisableWifi(); LOGI("DisableWifi: %d", errCode); osDelay(200);</pre>
}	

编译下载运行程序:

👳 星通智联串口调试助	手 V1.8.1.0 更多信息可访问: https://www	v.w600.fun	🔅
接收 [336] INFO wifis [336] INFO wifis	ta Scan: 0 ta waiting for scan done	▲ 多文本发送 发送历史 下	载历史 引脚定义 在线资源
[336] INFO wifis 0, 300, TP-LINK_ [337] INFO wifis 0, 188, @snail	ta Result[0]: D0:, PSK, -94, 57E8 ta Result[1]: 68:, PSK, -80,	W800	TB-08
[437] INFO wifis InitWifiConfig, WifiEventCallbac [461] INFO wifis 105. state = 1.	ta AddDeviceConfig: 0 disconnect wifi k status = WIFI_JOIN_SUCCESS ta OnWifiConnectionChanged info =	引脚分布图	引脚分布图
[461] INFO wifis rssi: 77, connSt @snail Connect to @snai [467] INFO wifis	ta bssid: 68:DB:54:59:2F:74, ate: 1, reason: 0, ssid: 1 done, st atus 1 ta ConnectTo(0): 0	W600	W601
[467] INFO Wifis success [468] INFO wifis WifiEventCallbac	ta waiting for connect ta g_connected: 1 k status = NETIF_IP_NET_UP	引脚分布图 ★ ★ 入式 Wi-Fi 芯片 W600	引脚分布图
端口号 COM7 🗸	关闭串口	间 字体设置 🗆 нах显示	隐藏而板
波特率 115200 👻	复位设备 保存接收 🗆 自动换	行 【背景设置 ☑ 切换串口	156,984 pag DX
数据位 8 🛛 🗸 🗸	型号: Unknown 🗸 速率: 2Mbps 🗸	〕连续下载□ 擦除Flash	
校验位 None 🗸 🗸	固件: E:\w800驺蒙系统\w800.fls		11.67
停止位 One ~	□ 定时发送 1000 ms/次 □ 发送新	行 □ ਮॾХ发送 □ 发送文件	
流控制 None 🗸	发送 w800鸿蒙系统		
COM7 Opend	Received: 108301	Sent: 0	2020-12-13 17:17:10 .:

4.3 w800 鸿蒙系统 TCP客户端通信

本节我们使用开发板建立一个 tcp 客户端,再去连接指定地址和端口的 tcp 服务端并进行数据通信。首先配置我们需要连接的服务器 IP,测试时我们 写入电脑的 IP 地址,配置我们的路由器信息

C gpto_example.c 5 C led_example.c 5 C lowpower_example.c 5 C uart_example.c 3 ✓ network E BUILD.gn C net_common.h C net_demo_common.h C net_demo_main.c C net_demo_ohos.c C net_params.h C tcp_client_demo.c C tcp_client_test.c C tcp_client_test.c	<pre>24 * SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER 25 * CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY 26 * OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE US 27 * OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE. 28 */ 29 #ifndef NET_PARAMS_H 30 #define NET_PARAMS_H 31 32 #define PARAM_HOTSPOT_SSID "@snail" // your AP SSID 33 #define PARAM_HOTSPOT_PSK "ss559550" // your AP SSID 34 35 #define PARAM_HOTSPOT_TYPE WIFI_SEC_TYPE_PSK // defined in wifi_device_config.h 36 37 #define PARAM_SERVER_ADDR "192.168.2.104" // your PC IP address 38 #define PARAM_SERVER PORT 8080</pre>
C net_params.h C tcp_client_demo.c C tcp_client_test.c C tcp_client_test.h C wifi_connecter.c C wifi_connecter.h > samor	<pre>35 #define PARAM_HOTSPOT_TYPE WIFI_SEC_TYPE_PSK // defined in wifi_device_config.h 36 37 #define PARAM_SERVER_ADDR "192.168.2.104" // your PC IP address 38 #define PARAM_SERVER_PORT 8080 39 40 #endif // NET_PARAMS_H</pre>

创建一个 TCP 通信的 socket:

static char request[] = "Hello"; static char response[128] = "";


```
void TcpClientTest(const char* host, unsigned short port)
    ssize_t retval = 0;
    int sockfd = socket(AF_INET, SOCK_STREAM, 0); // TCP socket
    struct sockaddr in serverAddr = {0};
    serverAddr.sin_family = AF_INET;
    serverAddr.sin_port = htons(port);
    if (inet_pton(AF_INET, host, &serverAddr.sin_addr) <= 0)</pre>
        { printf("inet_pton failed!\r\n");
        goto do_cleanup;
    if (connect(sockfd, (struct sockaddr *)&serverAddr, sizeof(serverAd
dr)) < 0) {
        printf("connect failed!\r\n");
        goto do_cleanup;
    printf("connect to server %s success!\r\n", host);
    retval = send(sockfd, request, sizeof(request), 0);
    if (retval < 0) {
        printf("send request failed!\r\n");
        goto do_cleanup;
    printf("send request{%s} %ld to server done!\r\n", request, retval)
    retval = recv(sockfd, &response, sizeof(response), 0);
    if (retval <= 0) {</pre>
        printf("send response from server failed or done, %ld!\r\n", re
tval);
        goto do_cleanup;
    response[retval] = '\0';
    printf("recv response{%s} %ld from server done!\r\n", response, ret
val);
do cleanup:
    printf("do_cleanup...\r\n");
    close(sockfd);
```


http://www.doit.am https://szdoit.taobao.com/ Tel: 186 7666 2425, QQ: 114209716

配 BUILD.gn:

C lowpower_example.c	4 # You may obtain a copy of the License at
C myhelloworld.c	5 #
C uart_example.c	6 # <pre>http://www.apache.org/licenses/LICENSE-2.0</pre>
✓ network	7 #
≣ BUILD.gn	8 # Unless required by applicable law or agreed to in writing, software
C net_common.h	9 # UISCLIDULEU UNDER CHE LICENSE IS UISCLIDULEU UN AN AS IS DASIS, 10 # WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND wither express or implied
C net_demo_common.h	10 # Window Wakdamies of constricts of and kind, cities express of implied.
C net demo main.c	12 # limitations under the License.
C net_demo_ohos.c	13
C net params.h	<pre>14 import("//build/lite/config/component/lite_component.gni")</pre>
C tcp client demo.c	15
C tcp client test.c	16 Lite_component("app") {
C tcp_client_test.h	1/ Icalules – L 18 "network:net demo"
C wifi connecter.c	
C wifi connecter.h	20 }
> samgr	
✓ startup	
≣ BUILD.gn	
✓ wifitest	
≣ BUILD.gn	
① README.md	
C wifi_connect_demo.c	
C wifi_hotspot_demo.c	
C wifi_scan_demo.c	
E BUILD.gn	

电脑建立一个 TCP 服务器:

http://www.doit.am https://szdoit.taobao.com/ Tel: 186 7666 2425, QQ: 114209716

		网络	调试助	f		→ - □ ×
网络设置 (1)协议类型 TCP Server ▼	数据日志				I	etAssist V4.3.13
(2)本地主机地址 192.168.2.104 💌						
(3)本地主机端口 8080						
关闭						
接收设置						
• ASCII C HEX						
▼ 按日志模式显示						
☑ 接收完自动换行	<					
□ 接收转向至文件						
□ 暂停接收区显示						
其他洗顶 清除接收						
发送设置						
• ASCII C HEX						
□ 自动解析转义符						
「 AT指令自动回车						~
□ 自动发送校验位	40.100222	سند ک	111 0	(1) (2)	4 MT TT	
□ 打开文件数据源	- - - - - - - - - -	各戶端:	ALL Conn	ections (U) 💌	━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━	★ 消除 ℃ 消除
□ 循环周期 50 ms	http://www.	cmsoft.cn				10.32
快捷定义历史发送						友达
(♂ 就绪!		2/2		RX:12	TX:40	复位计数

编译程序并下载到开发板,通信成功发 hello 到服务器:

▶ 星通智联串口调试助手 V1.8.1.0 更多信息可访问: h	ttps://www.w600.fun —		网络调试助手	- D	
Wk est!	▲ 多文本发送 发送历史 下载历史 引脚员	四络设置 (1)协议类型	數据日志	NetAssist V4.3.	
est! fter 6 seconds, I will start TopClien		TCP Server <u>▼</u> (2) 本地主机地址	[2020-12-13 15:55:58 129]# Client 192 168.2	121:51457 gets online.	
fter 5 seconds, I will start TopClien est!		(3)本地主机端口	2020-12-13 15:55:58 134j# REUV ASCII FRAM Hello	192, 168, 2, 121 - 514572	
est! fter 3 seconds, I will start TcpClien est!	ر جمع الحال العلي المعلي الحالي المعلي الحالي المعلي الحالي المعلي الحالي المعلي المعلي المعلي المعلي المعلي ال المعلي المعلي	() () () () () () () () () () () () () ([2020-12-13 15:57:17.787]# Client 192.168.2 [2020-12-13 15:57:17.798]# RECV ASCII FROM	.121:50416 gets online. 192.168.2.121 :50416>	
<pre>fter 2 seconds, I will start TcpClien est! fter 1 seconds, I will start TcpClien est! fter 0 seconds I will start TcpClien</pre>	tTest W600 W6	·接收设置 ⓒ ASCII ○ HEX ☞ 按日志模式显示	Rello		
est cpClientTest start onnect to server 192.168.2.104 succes end request[Hello] 6 to server done!	引脚分布图 引脚分	✓ 接收完自动换行 ✓ 接收完自动换行 ✓ 接收转向至文件 ✓ 暂停接收区显示 其他选项 置余接收	<		
第四号 COM7 ジ 关闭串ロ 青空接收	 ·	发送设置 ・ ASCII C HEX			
皮特率 115200 - 夏位设备 保存接收	□ 自动换行 背景设置 ✓ 切换串口	□ 自动解析转义符 □ AT指令自动回答			
数据位 8 型号: Unknown 速率: 2 交验位 None 固件: E: \w8003B蒙系统\w800.	Mbps ◇ 直续下载 擦涂Flash fls · · · · 下载 打包	「自动发送校验位」 「打开文件数据源	 教讃发送 客户端: All Connections (2) ▼ ◆ 断开 ↓ 青絲 1 清		
事止位 One □ 定时发送 1000 ms/次	□ 发送新行 □ HEX发送 □ 发送文件	「循环周期 50 ms 快捷完义 历史发送	http://www.omsoft.on	发送	
航控制 None 发送 w8003募系统		1 of ottin	4/2 RX:24	TX:40 复位计数	
MIT Onand Parahuad 4522	Sent 0 2020-1			38.12 11 84	

4.4 w800 鸿蒙系统 wifi 扫描热点

这节我们使用开发板扫描周围的热点并列出来,建立扫描任务

static void WifiScanDemo(void)
{
 osThreadAttr_t attr;
 attr.name = "WifiScanTask";
 attr.attr_bits = 0U;
 attr.cb_mem = NULL;
 attr.cb_size = 0U;
 attr.stack_mem = NULL;
 attr.stack_size = 10240;
 attr.priority = osPriorityNormal;
 if (osThreadNew(WifiScanTask, NULL, &attr) == NULL)
 { printf("[WifiScanDemo] Falied to create
 WifiScanTask!\n");
 }

任务程序中执行扫描操作:

```
static void WifiScanTask(void *arg)
   (void)arg;
   WifiErrorCode errCode;
   WifiEvent eventListener = {
        .OnWifiConnectionChanged = OnWifiConnectionChanged,
        .OnWifiScanStateChanged = OnWifiScanStateChanged
   };
   osDelay(100);
   errCode = RegisterWifiEvent(&eventListener);
   printf("RegisterWifiEvent: %d\r\n", errCode);
   while (1) {
       errCode = EnableWifi();
        printf("EnableWifi: %d\r\n", errCode);
       osDelay(200);
       g_scanDone = 0;
        errCode = Scan();
       printf("Scan: %d\r\n", errCode);
       while (!g_scanDone) {
```


编译下载运行程序:

http://www.doit.am https://szdoit.taobao.com/ Tel: 186 7666 2425, QQ: 114209716

五、w800 鸿蒙系统连接 dohome 平台云

Doit 智能云 (wechat.doit.am) 是由深圳四博智联科技有限公司开 发的可直接用于生产环境的物联网云平台。Doit 智能云可对单个设备或 是一组设备进行远程控制、接收上传数据并实时展示、实现定时任务 (精 确到秒)等,特有的事件统计功能可以对每台设备的开机时间和时长进行 统计和分析。

针对日益增长的微信控制智能设备需求, Doit 智能云实现了:

1、 针对每台设备生成唯一的二维码,该二维码可被微信和手机 APP 同时扫描绑定。若设备数量在 10 万以下,可直接免费使用 Doit 智能云 实现微信控制,省去微信 API 复杂开发流程。

2、 针对每一类产品,生成产品标示二维码,通过微信或者手机 APP 实现该类产品的批量推送和控制。

3、 在设备端提供最全面的配置上网方式案例,包括微信的 Airkiss、 ESP - Touch (针对 ESP8266)、Easylink (针对 EMW3165)、Soft AP、 网页配置等,确保只要有路由器,就能使设备配置上网成功。

4、 控制方式多种多样, 手机 app 控制、微信控制、直连 Soft AP 控制、局域网控制等。

5、 支持 TCP、Websocket 等多种接入方式。在协议设计上,采用纯 文本协议,支持推送、上传、管道等多种通讯功能,保证数据传输的便利 性、实时性和安全性。

基于强大的 Doit 智能云平台,开发一个智能插座或智能小车类功能, 包含手机端、微信端、设备端的程序仅需要半个工作日的时间!在开发过 程中,Doit 智能云提供设备虚拟功能,可实现并行开发,加速产品的开 发进程。基于 Doit 智能云,我们已经开发了智能插座、智能小车等产品, 后续会有新的产品模板添加,利用这些产品模板,厂商不需要写任何代码, 就能直接拿去生产相应的产品到市场销售。

首先我们建立一个任务,让板子连接上路由器:

```
static void NetDemoEntry(void)
{
    osThreadAttr_t attr;
    attr.name = "NetDemoTask";
    attr.attr_bits = 0U;
    attr.cb_mem = NULL;
    attr.cb_size = 0U;
    attr.stack_mem = NULL;
    attr.stack_size = 10240;
    attr.priority = osPriorityNormal;
    if (osThreadNew(NetDemoTask, NULL, &attr) == NULL) {
        printf("[NetDemoEntry] Falied to create NetDemoTask!\n");
     }
        RT買路由器信息并连接:
```

#define PARAM HOTSPOT SSID "@snail" // your AP SSID

```
深圳四博智联科技有限公司 Shenzhen Doctors of Intelligence & Technology
          (DOIT)
                          https://szdoit.taobao.com/ Tel: 186 7666 2425. OO: 114209716
          http://www.doit.am
#define PARAM HOTSPOT PSK "ss559550" // your AP PSK
    // 准备 AP 的配置参数
    strcpy(config.ssid, PARAM HOTSPOT SSID);
    strcpy(config.preSharedKey, PARAM_HOTSPOT_PSK);
    config.securityType = PARAM_HOTSPOT_TYPE;
    osDelay(10);
    int netId = ConnectToHotspot(&config);
    int timeout = 10;
    while (timeout--) {
        printf("After %d seconds, I will start %s test!\r\n", timeout, GetN
etDemoName());
        osDelay(100);
    }
```

等待 10 秒后开始连接 dohome 的服务器:

```
void TcpClientTest(const char* host, unsigned short port)
   ssize_t retval = 0;
   int sockfd = socket(AF INET, SOCK STREAM, 0); // TCP socket
   struct sockaddr_in serverAddr = {0};
   serverAddr.sin family = AF INET;
   serverAddr.sin_port = htons(port);
   if (inet_pton(AF_INET, host, &serverAddr.sin_addr) <= 0) {</pre>
       printf("inet_pton failed!\r\n");
       goto do_cleanup;
   if (connect(sockfd, (struct sockaddr *)&serverAddr, sizeof(serverAddr))
< 0) {
       printf("connect failed!\r\n");
       goto do cleanup;
   printf("connect to server %s success!\r\n", host);
   retval = send(sockfd, request, sizeof(request), 0);
   if (retval < 0) \{
       printf("send request failed!\r\n");
       goto do cleanup;
    }
   printf("send request{%s} %ld to server done!\r\n", request, retval);
   retval = recv(sockfd, &response, sizeof(response), 0);
   if (retval <= 0) {</pre>
```


http://www.doit.am https://szdoit.taobao.com/ Tel: 186 7666 2425, QQ: 114209716

printf("send response from server failed or done, %ld!\r\n", retval
);
 goto do_cleanup;
 }
 response[retval] = '\0';
 printf("recv response{%s} %ld from server done!\r\n", response, retval);
do_cleanup:
 printf("do_cleanup...\r\n");
 close(sockfd);

程序编译下载到开发板运行:

