阿里云 IoT物联网操作系统

快速开始

文档版本: 20191114

为了无法计算的价值 | [] 阿里云

法律声明

阿里云提醒您在阅读或使用本文档之前仔细阅读、充分理解本法律声明各条款的内容。如果您阅读 或使用本文档,您的阅读或使用行为将被视为对本声明全部内容的认可。

- 您应当通过阿里云网站或阿里云提供的其他授权通道下载、获取本文档,且仅能用于自身的合法 合规的业务活动。本文档的内容视为阿里云的保密信息,您应当严格遵守保密义务;未经阿里云 事先书面同意,您不得向任何第三方披露本手册内容或提供给任何第三方使用。
- 未经阿里云事先书面许可,任何单位、公司或个人不得擅自摘抄、翻译、复制本文档内容的部分 或全部,不得以任何方式或途径进行传播和宣传。
- 3. 由于产品版本升级、调整或其他原因,本文档内容有可能变更。阿里云保留在没有任何通知或者 提示下对本文档的内容进行修改的权利,并在阿里云授权通道中不时发布更新后的用户文档。您 应当实时关注用户文档的版本变更并通过阿里云授权渠道下载、获取最新版的用户文档。
- 4. 本文档仅作为用户使用阿里云产品及服务的参考性指引,阿里云以产品及服务的"现状"、"有缺陷"和"当前功能"的状态提供本文档。阿里云在现有技术的基础上尽最大努力提供相应的介绍及操作指引,但阿里云在此明确声明对本文档内容的准确性、完整性、适用性、可靠性等不作任何明示或暗示的保证。任何单位、公司或个人因为下载、使用或信赖本文档而发生任何差错或经济损失的,阿里云不承担任何法律责任。在任何情况下,阿里云均不对任何间接性、后果性、惩戒性、偶然性、特殊性或刑罚性的损害,包括用户使用或信赖本文档而遭受的利润损失,承担责任(即使阿里云已被告知该等损失的可能性)。
- 5. 阿里云文档中所有内容,包括但不限于图片、架构设计、页面布局、文字描述,均由阿里云和/或其关联公司依法拥有其知识产权,包括但不限于商标权、专利权、著作权、商业秘密等。 非经阿里云和/或其关联公司书面同意,任何人不得擅自使用、修改、复制、公开传播、改变、 散布、发行或公开发表阿里云网站、产品程序或内容。此外,未经阿里云事先书面同意,任何人 不得为了任何营销、广告、促销或其他目的使用、公布或复制阿里云的名称(包括但不限于单独 为或以组合形式包含"阿里云"、"Aliyun"、"万网"等阿里云和/或其关联公司品牌,上述 品牌的附属标志及图案或任何类似公司名称、商号、商标、产品或服务名称、域名、图案标示、 标志、标识或通过特定描述使第三方能够识别阿里云和/或其关联公司)。
- 6. 如若发现本文档存在任何错误,请与阿里云取得直接联系。

通用约定

格式	说明	样例
0	该类警示信息将导致系统重大变更甚 至故障,或者导致人身伤害等结果。	禁止: 重置操作将丢失用户配置数据。
	该类警示信息可能会导致系统重大变 更甚至故障,或者导致人身伤害等结 果。	▲ 警告: 重启操作将导致业务中断,恢复业务 时间约十分钟。
!	用于警示信息、补充说明等,是用户 必须了解的内容。	注意:权重设置为0,该服务器不会再接受 新请求。
Ê	用于补充说明、最佳实践、窍门 等,不是用户必须了解的内容。	道说明: 您也可以通过按Ctrl + A选中全部文件。
>	多级菜单递进。	单击设置 > 网络 > 设置网络类型。
粗体	表示按键、菜单、页面名称等UI元 素。	在结果确认页面,单击确定。
Courier字体	命令。	执行cd /d C:/window命令,进 入Windows系统文件夹。
##	表示参数、变量。	bae log listinstanceid
		Instance_ID
[]或者[a b]	表示可选项,至多选择一个。	ipconfig [-all -t]
{}或者{a b}	表示必选项,至多选择一个。	<pre>switch {active stand}</pre>

目录

法律声明	. I
通用约定	. I
1 获取源码	.1
2 使用命令行工具开发	. 2
3 使用 AliOS Studio (Visual Studio Code)	4
4 自动生成 Keil MDK 工程	8
5 自动生成 IAR 工程	11
6 生成IAR/KEIL工程常见问题	14
6.1 确认是否支持生成keil/IAR工程	.14
6.2 L6218E: Undefined symbol _Heap2Base (referred from soc_impl.o)	14
6.3 armcc: not found	. 15
6.4 C3903U: Argument '/hardfp' not permitted for option 'apcs'	. 15

1 获取源码

2 使用命令行工具开发

AliOS Things 在命令行中使用 aos-cube 工具完成系统编译构建。

配置环境

下载并安装 Python 2.7。

安装 aos-cube

首先,用 python 包管理器 pip 来安装 aos-cube。推荐在虚拟 Python 环境中进行,如 Conda , virtualenv。

\$ pip install aos-cube



请确认 pip 环境是基于 Python 2.7 的。如果遇安装问题,请添加 --user 参数。具体请参考 pip install 用法。

下载代码并编译烧录运行

此处使用 Developer Kit

・AliOS Things版本在2.1.0之前,运行指令如下。

```
git clone https://github.com/alibaba/AliOS-Things.git -b <release_br
anch_name>
cd AliOS-Things
aos make helloworld@developerkit # 编译构建
(... 编译过程)
aos upload helloworld@developerkit # 固件烧录
(... 烧录过程)
```

・AliOS Things版本在2.1.0之后,运行指令如下。

```
gitclone https://github.com/alibaba/AliOS-Things.git -b <release_br
anch_name>
cd AliOS-Thingsaosmakelinkkitpp@developerkit -c config #先配置好板
子和app
```

```
aos make
aos upload
```

#编译构建(...编译过程) #固件烧录(...烧录过程)

三〕 说明:

· -b <release_branch_name>请用具体的版本号代替,可以参考#unique_5/

unique_5_Connect_42_section_mu6_bsk_tt7

・如果没有-b <release_branch_name>, 默认下载最新版本。

串口log显示

连接串口并重启开发板,可以看见 app_delayed_action 在1秒时启动,每5秒触发一次。AliOS Things 3.0.0版本的log如下所示。

Welcome to AliOS Things nano entry here! hello world! count 0 hello world! count 1 hello world! count 2 hello world! count 3 hello world! count 4 hello world! count 5 hello world! count 6 hello world! count 7 hello world! count 8



因AliOS Things版本的不同,log日志也会有所差别。

更多示例代码请参考如下

- blink
- mqttapp
- linkkitapp

3 使用 AliOS Studio (Visual Studio Code)

本文介绍AliOS Studio软件的安装及使用方法。

AliOS Studio是一套基于vscode

安装

1. 下载/安装vscode。

访问 https://code.visualstudio.com/

2. 安装AliOS Studio插件。

打开vscode,按照下图所示安装AliOS Studio插件。



3. 安装 aos-cube。

AliOS Studio依赖aos-cube,如果想要手动安装aos-cube的话,请参考见aos-cube。



AliOS-Studio一键安装的aos-cube是安装在虚拟python环境里的(virualenv

使用

1. AliOS-Studio 工具栏。

AliOS-Studio的主要功能都集中在vscode下方工具栏中,小图标从左至右功能分别是编译、烧录、串口工具、创建工程、清除

2. 编译 - Build

左侧的helloworld@starterkit是编译目标,格式遵循应用名字@目标板名字的规则,点击它可以依次选择应用和目标板,参考如下所示进行编译。

3. 烧录 - Upload

- a. 通过 USB Micro 线缆连接好开发板和电脑。
- b. 单击下方工具栏闪电图标完成固件烧录。



- 4. 串口监控 Monitor
 - a. 通过 USB Micro 线缆连接好开发板和电脑。
 - b. 单击下方工具栏插头图标打开串口。第一次连接会提示填写串口设备名和波特率,再次点击可以看到串口输出,同时也可以在这里输入命令进行交互。
- 5. 调试

请参考视频: 使用 AliOS Studio 开始 AliOS Things 调试。

4 自动生成 Keil MDK 工程

本文主要介绍如何生成并编译Keil工程。

```
准备工作
```

- ・安装<u>Keil</u>。
- · 添加keil的toolchain的路径到windows的环境变量Path中。

📋 说明:

keil的toolchain路径默认为C:\Keil_v5\ARM\ARMCC\bin。关于如何添加环境变量请参见 How to add a folder to Path environment variable

生成keil工程

本示例主要介绍如何生成helloworld@developerkit的keil工程。

1. 执行如下命令,编译生成keil工程。

```
// v2.1.0之后
aos make IDE=keil
// v2.1.0之前
aos make helloworld@developerkit IDE=keil
```

2. 将生成的keil工程目录存在projects/autogen/helloworld@developerkit/

keil_project文件夹中,具体文件内容如下。

使用Keil IDE编译工程

 进入projects\autogen\helloworld@developerkit\keil_project目录,打开生成 的keil工程helloworld@developerkit。

2. 添加设备。

a. 选择Project > Options for Target 'helloworld@developerkit',打开工程设置。

		da\i	ot\	206)	developV			tc\aut	ogen\	helloworl	ld@developerkit\keil_project\belloworld@developerkit.uvprojy_u\/i	sion
00	100	ue (i	010	ausi		4103-11	ings (projec	Lis (aut	ogen	nenowon		SION
File	Edit	Vie	w [Proj	ect Flash	Debug	Peripherals	Tools	SVCS	Window	Help	
	3 ,	1	1		New µVisio	n Project						
ا 🍪	× 1	š 🧳	•		New Multi-	Project V	Vorkspace					
roiec	Open Project								- 1			
rojec &	· _				Close Proje	ect						
E	Pro	oject	t: h€		Export							- N
Ē	- *	hel	low		Export							
	٠		atŗ		Manage							
	٠	- 	cli		Select Devi	ce for Tar	get					
	٠	-	de		Remove Ite	em						
	٠	-	dig	N.	Options for	r Target '	helloworld@d	evelope	rkit'			Alt+F7
	٠	-	ha		Clean Targ	ets						
	٠		he		Build Targe	 .t						F7
	Ð	-	kei	1255	Debuild all	taraat file						
	٠		kv		Rebuild all	larget me	5					
	Đ	-	ne	2	Batch Build	1						
			loc	1	Batch Setu	p						
	1		100		Translate							Ctrl+F7
	+		arr		Stop build							
	•		str	~	1 D:\code\i	iot\aos\d		Things\r	projects	\autogen\k	nelloworld@develonerkit\keil.nroject\helloworld@develonerkit.uvorojv	

b. 在Device页签选择对应的Device。

🔣 Options for Target 'helloworld@developerkit'	×
Device Target Output Listing User C/C++ Asm Linker Debug Utilities	
Software Packs	
Vendor: STMicroelectronics	
Device: STM32L475VGTx Pack: Keil.STM32L4xx_DFP.2.0.0	
Toolset ARM URL: <u>http://www.keil.com/pack</u>	
Search: stm32l496vgtx	
STMicroelectronics ST has built a new architecture to reach best-in-class ultra-low-power figures thanks to its high flexibility. STM32L4 Series STM32L496 STM32L496 STM32L496VG STM32L496VG STM32L496VG STM32L496VG Moreover, the STM32L4 series shatters performance limits in the ultr Iow-power world. It delivers 100 DMIPS based on its ARM Cortex-M4 core with FPU ar ART Accelerator at 80 MHz.	r ▲ a- nd ST 🗄
STM32L4 microcontrollers offer dynamic voltage scaling to balance consumption with processing demand, low-power peripherals (LP U LP timers) available in Stop mode, safety and security features, sma numerous peripherals, advanced and low-power analog peripherals as op amps, comparators, LCD, 12-bit DACs and 16-bit ADCs (hardw	power ART, Int and s such vare
OK Cancel Defaults	Help

3. 调试设置。

打开工程设置,选择Debug页签,根据芯片的不同选择相应的debug工具,例 如developerkit开发板应选择ST-Link Debugger,如下图所示。

🔀 Options for Target 'helloworld@developerkit'							
Device Target Output Listing User C/C++ Asm	Linker Debug Utilities						
C Use Simulator with restrictions Settings □ Limit Speed to Real-Time	Use: ULINK2/ME Cortex Debugger Settings ULINK2/ME Cortex Debugger UINK Pro Cortex Debugger						
Image: Second and Second							
Restore Debug Session Settings Image: Breakpoints Image: Toolbox Image: Breakpoints Image:	Restore PEMicro Debugger VULink Debugger VBre Stellaris ICDI SiLabs UDA Debugger Vater Windows V Memory Display						
CPU DLL: Parameter:	Driver DLL: Parameter:						
SARMCM3.DLL -REMAP -MPU	SARMCM3.DLL -MPU						
Dialog DLL: Parameter:	Dialog DLL: Parameter:						
DCM.DLL -pCM4	TCM.DLL -pCM4						
Warn if outdated Executable is loaded	Warn if outdated Executable is loaded						
Manage Component Vie	ewer Description Files						
OK Car	ncel Defaults Help						



在烧写代码前,需要先在Debug页签中配置工具,否则会出现烧写失败的问题。

在工程设置中添加设备并完成调试设置后,即可以用keil来编译、烧写代码和调试了。

5 自动生成 IAR 工程

本文主要介绍如何生成并编译IAR工程。

准备工作

- ・ 下载安装IAR Embedded Workbench
- · 添加IAR的toolchain路径到windows的环境变量Path中。



IAR的toolchain路径默认为C:\Program Files (x86)\IAR Systems\Embedded Workbench 8.0\arm\bin。关于如何添加环境变量请参见How to add a folder to Path

 $environment \ variable_{\circ}$

生成IAR工程

本示例主要介绍如何生成helloworld@developerkit的IAR工程。

1. 执行如下命令,编译生成IAR工程。

```
// v2.1.0及之后
aos make IDE=iar
// v2.1.0之前
aos make helloworld@developerkit IDE=iar
```

2. 将生成的IAR工程目录存在projects\autogen\helloworld@developerkit\

IAR_project文件夹中,具体文件内容如下。

使用IAR IDE编译调试工程

1. 用IAR打开工程。

进入projects\autogen\helloworld@developerkit\IAR_project目录,找到名称为 helloworld@developerkit,文件类型为workspace的文件,双击该文件即可以用IAR直 接打开工程。

名称	•	修改日期	类型	大小
👢 opts		2018/11/28 15:58	文件夹	
helloworld@developerkit.ewd		2018/11/28 15:56	EWD 文件	97 KB
helloworld@developerkit.ewp		2018/11/28 15:56	EWP 文件	243 KB
helloworld@developerkit		2018/11/28 15:56	IAR IDE Workspa	1 KB
utility_digest_algorithm_crc		2018/11/28 15:56	C 文件	2 KB

2. 打开工程设置。

打开工程后,选择Project > Options for node 'helloworld@developerkit',打开工程设置。



3. 选择正确的device。

选择general options > Target > Device,然后选取正确的芯片,即可以在IAR里编译此工程。

Options for node "hellowor	ld@develope	erkit"			×
Category: General Options Static Analysis Runtime Checking C/C++ Compiler Assembler Output Converter Custom Build Build Actions Linker Debugger Simulator CADI CMSIS DAP GDB Server I-jet/JTAGjet J-Link/J-Trace TI Stellaris PE micro ST-LINK Third-Party Driver	Librar Target Process © Core © Devi © CMS: Endi an © Litt © Big © B © B	y Options Output for varia ce CS-Pack mode le E32 E8	2 Libr nt Cortex ST STW None Flox FPU D	MISRA-C:2004 ary Configuration -M4 I32L496VG Iting point setting VFPv4 sin 16 Idanced SIMD (NEON) SP Extension	MISRA-C:1998 Library Options 1
TI MSP-FET TI XDS					OK Cancel

至此, AliOS Things在一块新的开发板的IAR工程已建立完毕, 开发者可以在此基础上使用IAR IDE环境开发更多的功能。

6 生成IAR/KEIL工程常见问题

6.1 确认是否支持生成keil/IAR工程

介绍生成keil/IAR工程失败时的处理方法

问题描述

生成keil/IAR工程失败。

问题解决

在要开发的board目录下打开.mk文件,确认是否有如下内容,且有相关的文件。

```
ifeq ($(COMPILER), armcc)
$(NAME)_SOURCES += startup_stm32l496xx_keil.s
else ifeq ($(COMPILER), iar)
$(NAME)_SOURCES += startup_stm32l496xx_iar.s
else
$(NAME)_SOURCES += startup_stm32l496xx.s
endif
ifeq ($(COMPILER),armcc)
GLOBAL_LDFLAGS += -L --scatter=board/developerkit/STM32L496.sct
else ifeq ($(COMPILER),iar)
GLOBAL_LDFLAGS += --config board/developerkit/STM32L496.icf
else
```

・如果有,表示已支持生成keil/IAR工程。

・如果没有,则需要按照生成keil/IAR工程的说明,完成keil/IAR工程的支持文件的添加。

6.2 L6218E: Undefined symbol _Heap2Base (referred from soc_impl.o)

介绍L6218E提示错误信息为Undefined symbol _Heap2Base (referred from soc_impl.o)时,如何和处理。

问题描述

L6218E提示错误信息:Undefined symbol _Heap2Base (referred from soc_impl.o)。

问题解决

按照AliOS Things移植文档里中的说明配置内核的堆,问题得到解决。文档说明内容如下。

2.2.6.3 内核使用堆的配置 如果要使用内存申请功能,则需要打开RHINO_CONFIG_MM_TLF宏,来使能k_mm模块,并且配 置对应的堆空间。 堆空间定义有三种方式:链接脚本定义、汇编定义、数组定义。推荐方式:链接脚本定义。 其基本原则是要预留一个内存空间作为堆使用,并将其交给g_mm_region管理。

该问题主要由soc_impl.c文件里的代码造成,该soc_impl.c文件的内容如下所示。

```
extern void *__HeapBase;
extern void *__HeapLimit;
extern void *__Heap2Base;
k_mm_region_t g_mm_region[] = {
{(uint8_t *)&__HeapBase, (uint32_t)0x8000},
{(uint8_t *)&__Heap2Base, (uint32_t)0x8000}};
int g_region_num = sizeof(g_mm_region)/sizeof(k_mm_region_t);
```

您可以根据移植文档,修改soc_impl.c文件和.sct文件来解决。也可以使用数组定义和汇编定义的 方法进行修改,具体请参见platform/mcu/stm32l4xx_cube/aos/soc_impl.c中的方法。

其它的芯片可能也会遇到相同的问题,也可以参见以上的方法进行解决。

6.3 armcc: not found

介绍armcc提示错误信息为not found时,如何处理

问题描述

armcc提示错误信息: not found。

问题解决

添加keil的bin目录到环境变量里,具体请参考自动生成Keil MDK工程 > 准备工作。

6.4 C3903U: Argument '/hardfp' not permitted for option 'apcs'

介绍C3903U提示错误信息为Argument '/hardfp' not permitted for option 'apcs'时,如何 处理

问题描述

C3903U提示错误信息: Argument '/hardfp' not permitted for option 'apcs'。

问题解决

keil没有注册,所以默认不支持浮点型编译,注册keil后,问题解决。