智能生活 C-SDK 适配 FreeRTOS

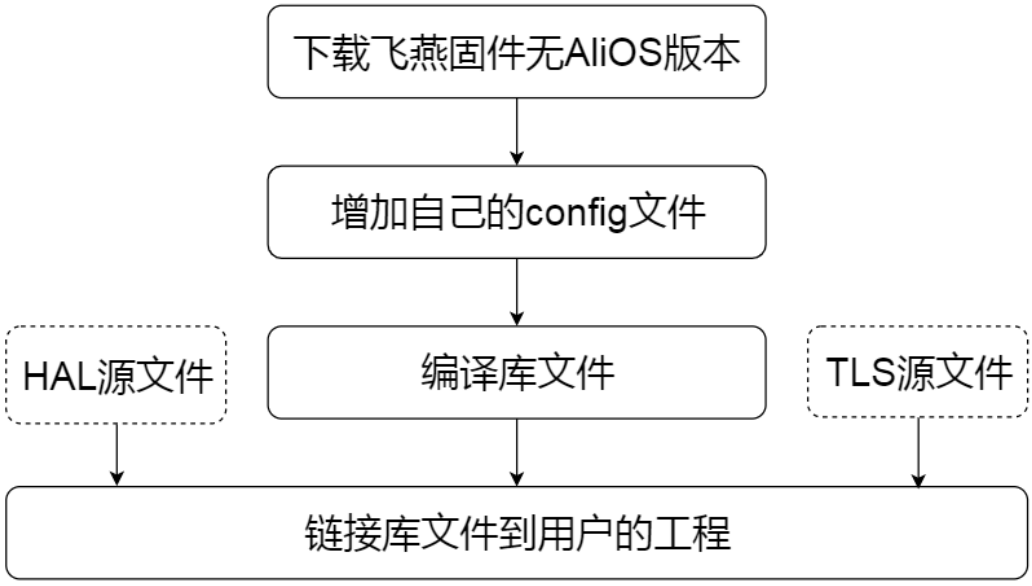
帮助文档

修订记录

修改时间	修改内容	修改人	备注
2020-05-21	第一个发布版本	远情	针对智能生活 SDKV1.6.0 版本

操作流程

整体集成流程如下图，其中 HAL 和 TLS 最好以源文件的形式在用户工程进行编译，否则如果在智能生活的工程中编译可能需要解决很多头文件的依赖问题。



移植步骤（以智能生活固件无 AOS 的 V1.6.0 版本为例）

第一步 下载代码

智能生活固件无 AOS 的版本 V1.6.0，下载地址参考智能生活帮助文档：

https://help.aliyun.com/document_detail/135278.html?spm=a2c4g.11186623.6.641.6ef444e3es97dK

第二步 编译智能生活 SDK 的代码

配置交叉编译器路径

文件 build-rules/settings.mk 中修改 TOOLCHAIN_DLDIR := /home/mytoolchain 配置编译器的文件夹所在的路径，然后修改 build-rules/funcs.mk 里面的函数 Relative_TcPath 增加编译器的相对路径：

```
define
( \
    case $(1) in \
        xtensa-lx106-elf-gcc ) \
            echo "gcc-xtensa-lx106-linux/main/bin" ;; \
        arm-none-eabi-gcc ) \
            echo "gcc-arm-none-eabi-linux/main/bin" ;; \
        nds32le-elf-gcc ) \
            echo "bin" ;; \
    esac \
)
endef
```

增加 config 文件

参考 src/board/文件夹里面的 config 增加一个自己的 config，例如增加 config.freertos.esp8266 文件，内容如下：

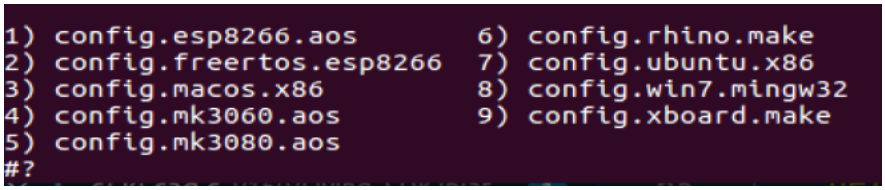
```
#添加芯片平台相关的配置
CONFIG_ENV_CFLAGS += \
```

```
    -DBOARD_ESP8266 -u call_user_start \
    -fno-inline-functions \
    -ffunction-sections \
    -fdata-sections \
    -mlongcalls \
    -Wl,-static

#配置编译器选项
CONFIG_ENV_CFLAGS += -Os

#配置不需要编译的子模块
CONFIG_src/ref-impl/tls         :=
CONFIG_src/ref-impl/hal         :=
CONFIG_examples                 :=
CONFIG_tests                    :=
CONFIG_src/tools/linkkit_tsl_convert :=

#交叉编译器的前缀，这里不要带路径
CROSS_PREFIX := xtensa-lx106-elf-
```

编译库文件

首先根据功能需要配置 SDK 的功能，有两种方法配置 SDK 的功能，方法一是直接编辑根目录下面的 make.settings 文件，方法二是执行 make menuconfig 进行配置，配置会修改 make.settings 文件。

然后编译 SDK，先执行 make clean，然后执行 make reconfig 选择你的配置，如下图输入数字 2 就是选择 config.freertos.esp8266 配置



然后执行 make，如果没有编译错误生成的文件在 output/release 文件夹下面，用户将库文件 output/release/lib/libiot_sdk.a 链接到自己的工程中，同时将目录 output/release/include 下面的头文件放置到自己的工程并配置头文件路

径，在用户的应用程序中 include "iot_import.h"即可，这个头文件会包含其他 SDK 的头文件，注意如果修改了文件 config.freertos.esp8266 都需要再次执行 make reconfig 才能生效。

常用 make 相关命令：

命令	解释
make distclean	清除一切构建过程产生的中间文件，使当前目录仿佛和刚刚 clone 下来一样
make	使用默认的或已选中的平台配置文件平台配置文件开始编译
make env	显示当前编译配置，非常有用，比如可显示交叉编译链，编译 CFLAGS 等
make reconfig	弹出多平台选择菜单，用户可按数字键选择，然后根据相应的硬件平台配置开始编译
make config	显示当前被选择的平台配置文件
make menuconfig	以图形化的方式编辑和生成功能配置文件 make.settings，直接编辑 make.settings 文件也是有效的
make help	打印帮助文本

补充说明

如果希望将 hal 在智能生活的工程中编译，需要先注释掉 config.freertos.esp8266 中的 CONFIG_src/ref-impl/hal := 同时将 hal 源文件放在文件夹 src/ref-impl/hal/os/freertos 文件夹中，依赖的头文件放到 prebuild/freertos/include 文件夹中，智能生活固件 V1.6.0 之前的版本这个头文件依赖有个问题，需要修改根目录下面的 project.mk 里面的变量 IMPORT_DIR 的值改为 IMPORT_DIR := $(TOP_DIR)/prebuilt，建议直接在用户自己的工程中编译 HAL 和 TLS 的代码，这样可以减智能生活 SDK 的编译报错。

第三步 链接库文件到用户工程

1. 为了减少编译 SDK 的错误，在 config.freertos.esp8266 里面增加了下面两行内容，就是不生成 tls 和 hal 的库，这个时候使用用户工程里面的 tls，并在用户自己的工程里面实现 HAL_xxx 一系列函数

```
CONFIG_src/ref-impl/tls         :=
CONFIG_src/ref-impl/hal         :=
```

2. 将 example/linkkit/living_platform 下面的文件放到用户的工程中编译，然后启动一个任务执行 living_platform_main 就可将整个 SDK 运行起来了
3. 解决适配过程的编译错误
4. 解决适配的运行错误，例如 coap 出错检查 HAL_UDP_xxx.c 是否适配好等

关于用户适配 HAL 的说明

1. 如果是 linux 系统，可以直接参考 src/ref-impl/hal/os/ubuntu 目录下面的 c 文件，大部分文件可以直接使用；
2. 如果不是 linux 的系统，例如 freertos 系统，参考 src/ref-impl/hal/os/ubuntu 下面的文件实现各个 HAL 函数的功能；
3. 配网相关的 HAL 函数比较多，关于 HAL 的说明可以参考智能生活官方帮助文档：
   https://help.aliyun.com/document_detail/155232.html?spm=a2c4g.11186623.6.670.67dc2c0b5kANxM