

Начало работы с ESP8266

Представленный перевод выполнен для личных нужд администратором сайта radioham.ru и не является официальным. rhf-admin исправил текст так, как захотел (для себя).

Version 2.0
Copyright © 2016

Об этом руководстве

В качестве примеров по использованию ESP8266 SDK в этом документе используются ESP-LAUNCHER и ESP-WROOM-02.

Структура документа:

Глава	Название	Содержание
Глава 1	Обзор	Общее вступление к процедуре использования SDK и ознакомление с HDK, FW и инструментами для ESP8266
Глава 2	Подготовка «железа»	Вступление к конфигурированию и настройке «железа» перед программированием. Показано на двух примерах: ESP-LAUNCHER и ESP-WROOM-02.
Глава 3	Подготовка «софта»	Вступление к non-OS SDK и RTOS SDK. Информация по инструментам для компиляции SDK и загрузки прошивки.
Глава 4	Карта флэш-памяти	Описание адресации и разметки флэш-памяти для загрузки прошивки. Вступление к прошивкам FOTA и non-FOTA.
Глава 5	Компилирование SDK	Вступление к компилированию SDK с помощью инструментов компиляции
Глава 6	Загрузка прошивки	Вступление к загрузке прошивки с помощью инструментов загрузки
Приложение 1	Конфигурирование режима QIO для ISSI-флеш	Вступление к конфигурированию и режима QIO для ISSI-флеш

Замечания к выпуску

Дата	Версия	Замечания к выпуску
2016.04	V1.3	Первый выпуск

Связанные документы

Ссылки для загрузки связанных документов:

Официальный веб-сайт Espressif: <http://www.espressif.com/support/download/documents>

Официальная BBS Espressif: <http://bbs.espressif.com/viewtopic.php?f=67&t=225>

Категория	Документы
Руководства по HDK	<i>ESP8266 Hardware Description</i> <i>ESP-WROOM-02 Datasheet</i>
Руководства по SDK	<i>ESP8266 Non-OS SDK Getting Started Guide</i> <i>ESP8266 Non-OS SDK AT Instruction Set</i>

Содержание

1. Обзор
 - 1.1. Обзор методики
 - 1.2. ESP8266 HDK
 - 1.3. ESP8266 SDK
 - 1.3.1. Non-OS SDK
 - 1.3.2. RTOS SDK
 - 1.4. ESP8266 FW
 - 1.5. ESP8266 инструменты
 - 1.5.1. Компилятор
 - 1.5.2. Инструмент загрузки прошивки
 - 1.5.3. Инструмент отладки через последовательный порт
2. Подготовка «железа»
 - 2.1. ESP-LAUNCHER
 - 2.2. ESP-WROOM-02
3. Подготовка «софта»
 - 3.1. Non-OS SDK
 - 3.2. RTOS SDK
 - 3.3. ESP8266 Инструменты
 - 3.3.1. Компилятор
 - 3.3.2. Инструмент для загрузки прошивок
4. Карта флеш-памяти
 - 4.1. Non-FOTA
 - 4.1.1. Карта флеш-памяти
 - 4.1.2. Адреса загрузки
 - 4.2. FOTA
 - 4.2.1. Карта флеш-памяти
 - 4.2.2. Адреса загрузки
5. Компилирование SDK
 - 5.1. Подготовка
 - 5.1.1. Модификация файлов SDK
 - 5.1.2. Загрузка файлов SDK
 - 5.2. Компиляция
 - 5.2.1. Компиляция ESP8266_NONOS_SDK_v0.9.5 и более поздних
 - 5.2.2. Компиляция ESP8266_NONOS_SDK_v0.9.4 и более ранних
6. Загрузка прошивки
 - 6.1. Процедура загрузки
 - 6.2. Проверка log-файла
 - 6.2.1. ESP8266_IoT_Demo
 - 6.2.2. ESP8266_AT
 - 6.3. Настройка инициализации RF (опционально)
 - 6.3.1. Настройка опций RF InitConfig
 - 6.3.2. Настройка параметров RF InitConfig
 - 6.3.3. Примеры настройки
- I. Приложение – Настройка режима QIO для ISSI-флеш

1. Обзор

1.1. Обзор методики

На рисунке 1-1 показан общий обзор процедуры компиляции SDK.

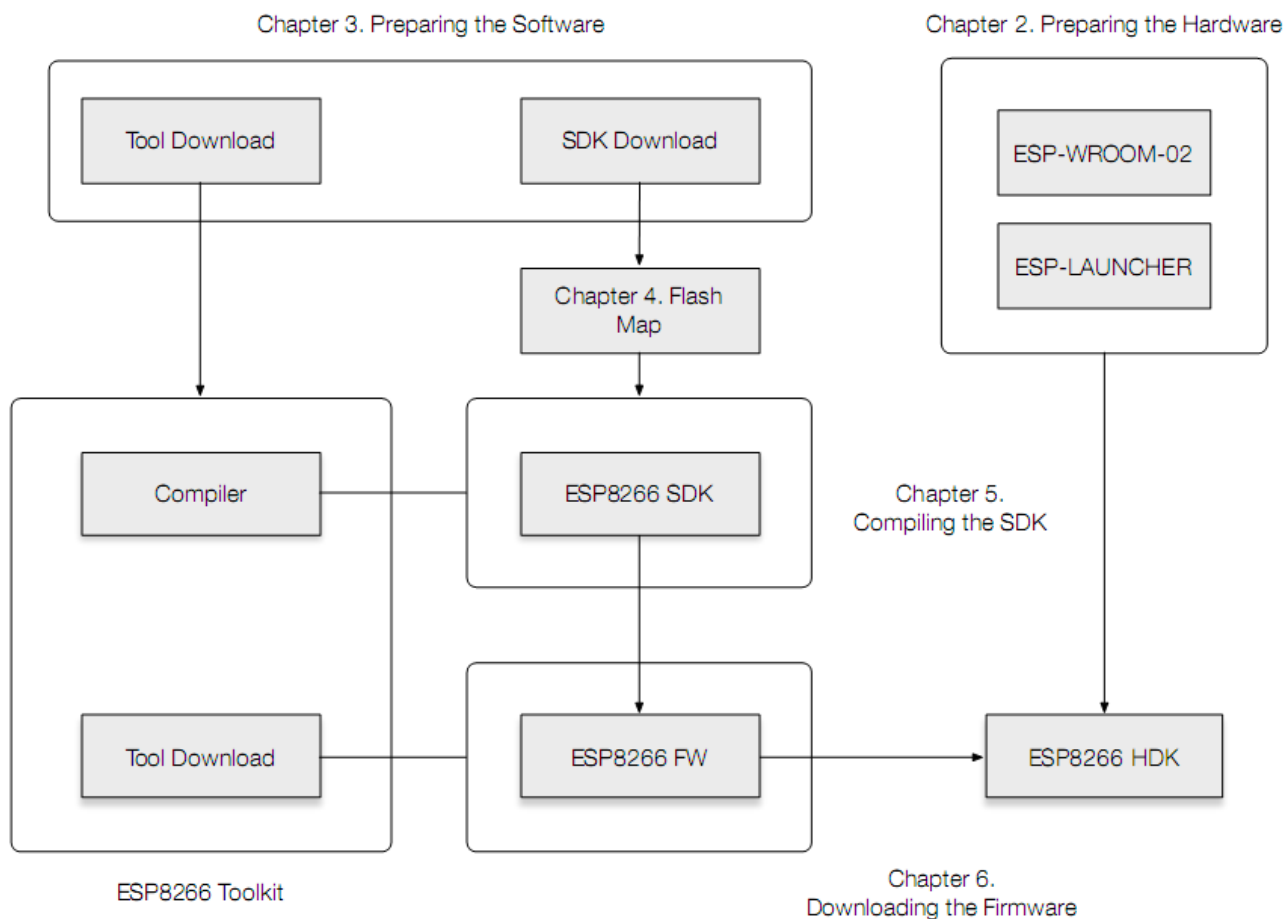


Рисунок 1-1 Обзор процедуры

1.2. ESP8266 HDK

ESP8266 HDK (Hardware Development Kit) включает чип – ESP8266EX, модуль – ESP-WROOM-02 и плату разработки – ESP-LAUNCHER. Пользователи могут загрузить предварительно скомпилированную прошивку используя ESP-WROOM-02 или ESP-LAUNCHER.

Замечания:

- Если вы используете другие платы разработки или модули в которые интегрирован чип ESP8266EX, пожалуйста используйте ПО для разработки, предоставляемое соответствующими производителями.
- Если вы хотите приобрести ESP-WROOM-02 или ESP_LAUNCHER, пожалуйста посетите официальный онлайн-магазин Espressif: <https://espressif.taobao.com>
- Больше информации о «железе» можно скачать по ссылкам:
 - Официальный веб-сайт Espressif: <http://www.espressif.com/support/download/documents>
 - Официальная BBS Espressif: <http://bbs.espressif.com/viewtopic.php?f=21&t=412&p=1545#p1545>

1.3. ESP8266 SDK

ESP8266 SDK (Software Development Kit) – это платформа разработки прошивок для IoT-устройств, предоставляемая Espressif разработчикам, которая включает базовую платформу и примеры ПО, разработанного для таких устройств как Умный Светильник и Умная Розетка.

В зависимости от того, базируется ли SDK на операционной системе или нет, SDK может быть разделено на две версии: Non-OS SDK и RTOS SDK.

1.3.1. Non-OS SDK

Non-OS SDK не базируется на операционной системе. Этот SDK поддерживает компиляцию IoT_Demo и AT команды. Non-OS SDK использует таймеры и колбэки в качестве главного пути предоставления различных функций – вложенных событий, функций, срабатывающих при возникновении определённых условий. Non-OS SDK использует сетевой интерфейс espconn; пользователи должны разрабатывать своё ПО в соответствии с правилами использования интерфейса espconn.

1.3.2. RTOS SDK

RTOS SDK базируется на операционной системе FreeRTOS с открытым исходным кодом, выложенным на Github.

- FreeRTOS SDK базируется на FreeRTOS, многозадачной ОС. Вы можете использовать стандартные интерфейсы для управления ресурсами, повторных операций, задержек выполнения, внутризадачных сообщений и синхронизации, а также другие подходы, основанные на задаче-ориентированных процессах. Детальное описание интерфейсных методов можно найти на официальном web-сайте FreeRTOS или в документе «USING THE FreeRTOS REAL TIME KERNEL – A Practical Guide».
- Сетевой интерфейс в FreeRTOS SDK – это стандартный lwIP API. RTOS SDK предоставляет пакет, который включает интерфейс BSD Socket API. Пользователи могут напрямую использовать socket API для разработки программ, а также портировать программы с других платформ на платформу ESP8266, используя socket API, эффективно уменьшая стоимость обучения, возникающую при переходе на другую платформу.
- RTOS SDK включает библиотеку cJSON, которая содержит функции для легкого распарсивания пакетов JSON
- RTOS совместима с Non-OS SDK в части интерфейсов Wi-Fi, Smart Config, интерфейсов, связанных со сниффером, системных интерфейсов, интерфейса таймера, интерфейсов FOTA и интерфейсов драйвера периферии, но не поддерживает включение AT-команд..

1.4. ESP8266 FW

ESP8266 FW (firmware - прошивка) предоставляется в виде бинарных (.bin) файлов, которые могут быть загружены напрямую в HDK. Пользователи могут выбирать между прошивками FOTA (Firmware Over-The-Air) и Non-FOTA. Более детальную информацию можно найти в Таблице 1-1.

Таблица 1-1. Значения полей в зависимости от размера и карты флеш-памяти (kB)

Бинарный файл	Обязательный или опциональный	Описание	Non-FOTA	FOTA
master_device_key.bin	Опциональный	Скачивается из облака, нужен для работы с облаком Espressif	✓	✓
esp_init_data_default.bin	Обязательный	Параметры системы по умолчанию, предоставляется SDK	✓	✓
blank.bin	Обязательный	Параметры системы по умолчанию, предоставляется SDK	✓	✓
eagle.flash.bin	Обязательный	Пользовательская программа, скомпиленная из SDK	✓	✗
eagle.irom0text.bin	Обязательный	Пользовательская программа, скомпиленная из SDK	✓	✗
user1.bin	Обязательный для первого использования	Пользовательская программа, скомпиленная из SDK	✗	✓
user2.bin	Используется для апгрейда	Пользовательская программа, скомпиленная из SDK	✗	✓

Замечания:

- Про содержимое SDK подробнее можно почитать в Главе 3 – Подготовка софта.
- Про компиляцию SDK подробнее можно почитать в Главе 5 – Компилирование SDK.
- Про адреса загрузки бинарных файлов подробнее можно почитать в Главе 4 – Карта флеш-памяти

1.5. ESP8266 инструменты

1.5.1. Компилятор

Для компилирования ESP8266 SDK нужна операционная система Linux. Если вы используете ОС Windows, мы рекомендуем использовать виртуальную машину с Linux, установленную в VirtualBox. Для упрощения процедуры компиляции мы предоставляем образ с установленными инструментами для компиляции. Пользователи могут импортировать наш образ (OVA) в свою виртуальную машину и сразу начать работать с компилятором.

1.5.2. Инструмент загрузки прошивки

Официальный инструмент загрузки прошивки, разработанный Espressif – это ESP8266 DOWNLOAD TOOL. С его помощью пользователи могут одновременно загрузить все необходимые бинарники в SPI-флеш материнской платы ESP8266 (ESP-LAUNCHER или ESP-WROOM-02) в соответствии с актуальным режимом компиляции и размером флеша.

1.5.3. Инструмент отладки через последовательный порт

Инструмент отладки через последовательный порт может быть использован для прямой связи с модулем ESP8266 через стандартный порт RS-232. Если ПК не имеет физического последовательного порта, то может использоваться виртуальный com-порт (преобразователь USB-to-serial)

Замечания:

- В качестве средства отладки через последовательный порт (в качестве терминала) мы рекомендуем CoolTerm (для Windows и MacOS) и Minicom (для Linux).
- В качестве USB-to-TTL (USB-to-UART) преобразователя, лично я рекомендую использовать преобразователь RH-003 (<https://radioham.ru/product/rh-0003/>)

2. Подготовка «железа»

В зависимости от того, используете ли вы ESP-LAUNCHER или ESP-WROOM-02, вам понадобится оборудование, перечисленное в Таблице 2-1:

Таблица 2-1. Подготовка «железа»

ESP-LAUNCHER	ESP-WROOM-02
<ul style="list-style-type: none">• 1 ESP-LAUNCHER• 1 USB cable	<ul style="list-style-type: none">• 1 ESP-WROOM-02• 1 USB-to-TTL преобразователь (лично я рекомендую RH-0003)• 6 проводков-соединителей• 1 паяльник
	

или

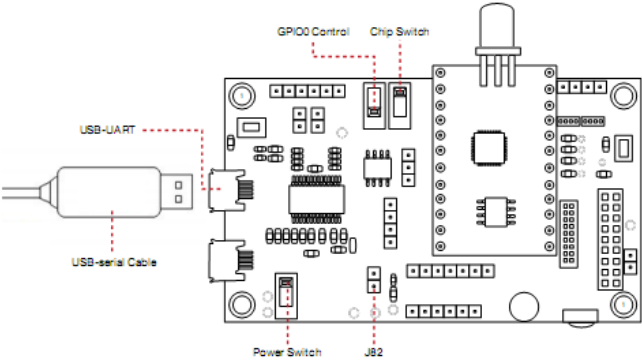
1 ПК с предустановленной ОС Windows

Замечание:

Для питания Wi-Fi модуля ESP8266 нужен источник постоянного напряжения 3.3В с минимальным током 500 мА.

2.1. ESP-LAUNCHER

1. Подключите ПК к USB-UART интерфейсу платы ESP-LAUNCHER используя USB кабель.
2. Переключите ESP-LAUNCHER в режим загрузки.

Шаги	Результаты
<ul style="list-style-type: none"> Переключите Power Switch в положение ближе к краю платы (состояние «выкл») Переключите переключатель GPIO0 Control в положение дальше от края платы для переключения ESP-LAUNCHER в режим загрузки 	
<p>Замечание: Контакты J82 должны быть закорочены джампером, в противном случае программа не сможет быть загружена в плату.</p>	

3. Подключите USB-to-TTL преобразователь к ПК.

Замечание:

Убедитесь, что соответствующий драйвер для USB-to-TTL преобразователя установлен и правильно распознан компьютером.

4. Запитайте ESP-LAUNCHER переключив Power Switch в положение дальше от края платы.

5. Запитайте чип, переключив Chip Switch в положение ближе к краю платы.

6. Загрузите прошивку во флеш с помощью ESP8266 DOWNLOAD TOOL.

Замечание:

Как и куда заливать прошивку читайте в **Главах 4 и 6**

7. После загрузки, переключите GPIO0 Control в положение ближе к краю платы для включения рабочего режима ESP-LAUNCHER.

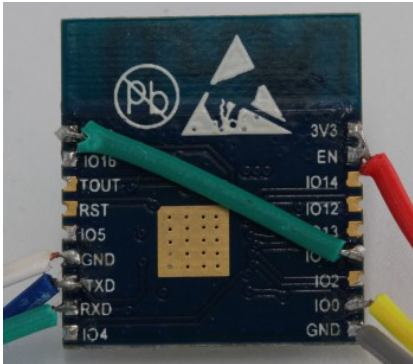
8. Сбросьте и снова запитайте чип с помощью переключателя Chip Switch и чип начнёт читать и выполнять программу из флеш-памяти.

Больше информации про ESP-LAUNCHER можно найти в документе **ESP8266 Hardware Description**.

2.2. ESP-WROOM-02

1. Припаяйте выводы для подключения в соответствии с Таблицей 2-2

Таблица 2-2. Выводы ESP-WROOM-02

Пин	Подключение пина	Картинка
EN	Подтянуть к питанию	
3V3	Подтянуть к питанию	
IO15	Подтянуть к земле	
IO0	Режим загрузки: подтянуть к земле Рабочий режим: подтянуть к питанию или оставить неподключенным	
GND	GND	
RxD	Подтянуть к питанию	
TxD	Подтянуть к питанию	

2. Подключите ESP-WROOM-02 к USB-to-TTL преобразователю, используя проводки-соединители как показано на Рисунке 2-1

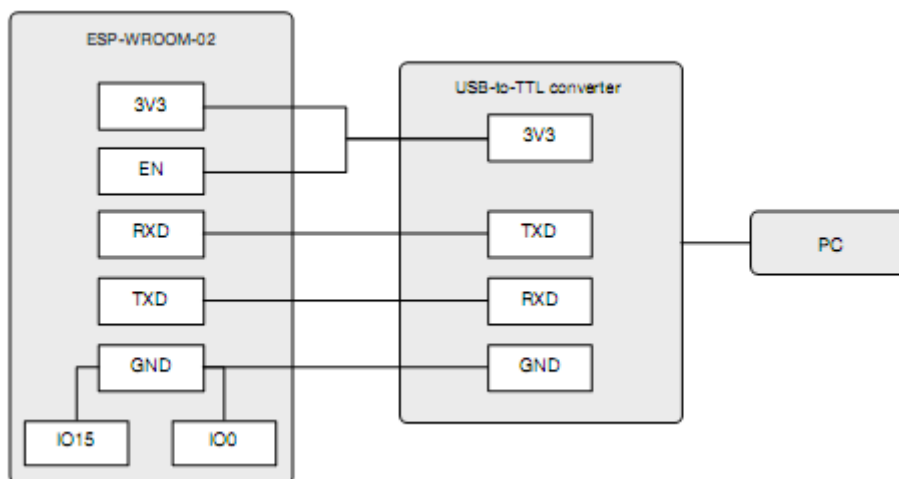


Рисунок 2-1. Подключение ESP-WROOM-02 в режиме загрузки

3. Подключите USB-to-TTL преобразователь к ПК

4. Загрузите прошивку во флеш при помощи ESP8266 DOWNLOAD TOOL.

Замечание:

Как и куда заливать прошивку читайте в **Главе 4 – Карта флеш-памяти** и в **Главе 6 – Загрузка прошивки**

5. После загрузки переключите ESP-WROOM-02 в рабочий режим, оставив IO0 неподключенным или подтянув к питанию.

6. Переподключите питание на ESP-WROOM-02 и чип начнет читать и выполнять программу из флеш-памяти.

Замечание:

Пин IO0 имеет встроенную подтяжку к питанию через резистор. Больше информации по этому поводу можно найти в документах **ESP8266 Hardware Description** и **ESP-WROOM-02 Datasheet**.

3. Подготовка «софта»

3.1. Non-OS SDK

Пользователи могут загрузить Non-OS SDK (включая примеры программ) по следующим ссылкам:

- Официальный сайт Espressif <http://www.espressif.com/en/support/download/sdks-demos>
- Официальная BBS Espressif <http://bbs.espressif.com/viewtopic.php?f=46&t=851>

На Рисунке 3-1 показана структура папок Non-OS SDK.

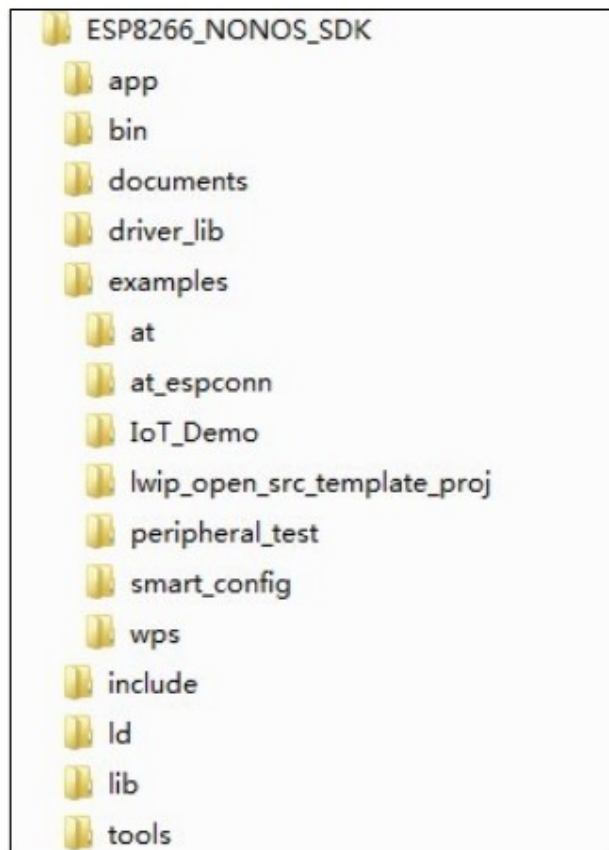


Рисунок 3-1. Структура папок Non-OS SDK

- **app**: главная рабочая директория, которая содержит файлы с исходными кодами и хидерами для компиляции.
- **bin**: скомпилированные бинарники для загрузки во флеш
- **documents**: связанные с SDK документы и ссылки
- **driver_lib**: библиотека файлов для работы с периферией, такой как UART, I2C и GPIO
- **examples**: примеры кодов для вторичной разработки, например, IoT_Demo
- **include**: файлы хидеров преинсталлированные в SDK. Файлы содержат соответствующие API функции и другие макроопределения. Пользователям не нужно менять эти файлы.
- **ld**: файлы для подключения программного обеспечения SDK. Мы рекомендуем пользователям не модифицировать их без особой необходимости.
- **lib**: файлы библиотек, предоставленных SDK
- **tools**: инструменты, необходимые для компилирования бинарников. Пользователям не нужно модифицировать их.

3.2. RTOS SDK

Пользователи могут загрузить RTOS SDK с примерами использующих этот SDK программ (ESP8266_IOT_PLATFORM) по следующим ссылкам:

- RTOS SDK https://github.com/espressif/ESP8266_RTOS_SDK
- ESP8266_IOT_PLATFORM https://github.com/espressif/ESP8266_IOT_PLATFORM

На Рисунке 3-2 показана структура папок RTOS SDK.

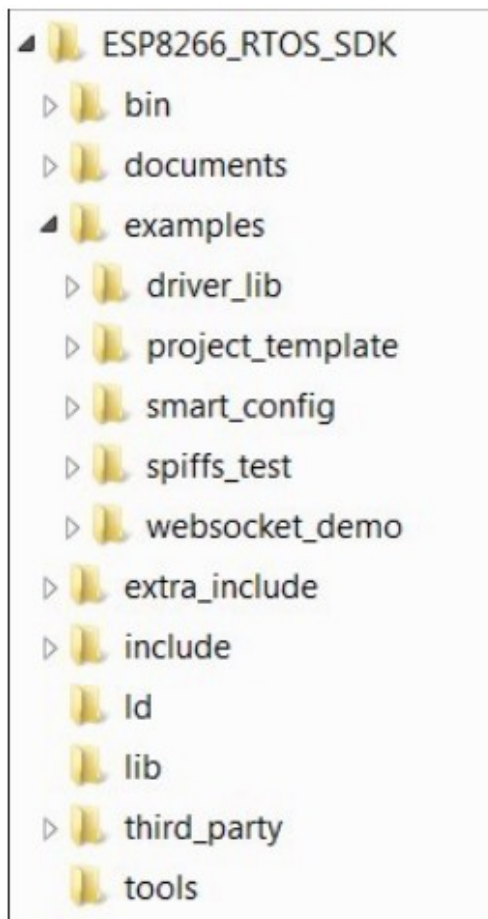


Рисунок 3-2. Структура папок RTOS SDK

- **bin**: скомпилированные бинарники для загрузки во флеш
- **documents**: связанные с SDK документы и ссылки
- **examples**: примеры кодов для вторичной разработки
 - **examples/driver_lib**: библиотека файлов для работы с периферией, такой как UART, I2C и GPIO
 - **examples/project_template**: шаблон папки проекта

Замечание:

Пользователи могут скопировать **project_template** в любую папку, например ~/workspace.

- **examples/smart_config**: примеры кодов, связанных со Smart Config
- **examples/spiffs_test**: примеры кодов, связанных со SPIFFS
- **examples/websocket_demo**: примеры кодов, связанных с веб-сокетами
- **extra_include**: файлы хидеров, предоставляемые Xtensa
- **include**: файлы хидеров, преинсталлированные в SDK. Файлы содержат соответствующие API функции и другие макроопределения. Пользователям не нужно менять эти файлы.
- **ld**: файлы для подключения программного обеспечения SDK. Мы рекомендуем пользователям не модифицировать их без особой необходимости.
- **lib**: файлы библиотек, предоставленных SDK
- **third_party**: библиотечный файл third party с открытым исходным кодом
- **tools**: инструменты, необходимые для компилирования бинарников. Пользователям не нужно модифицировать их.

3.3. ESP8266 инструменты

3.3.1. Компилятор

Скачайте VirtualBox по ссылке:

<https://www.virtualbox.org/wiki/Downloads>

Замечание:

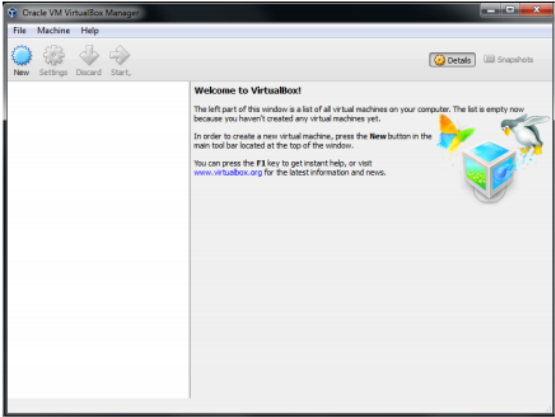
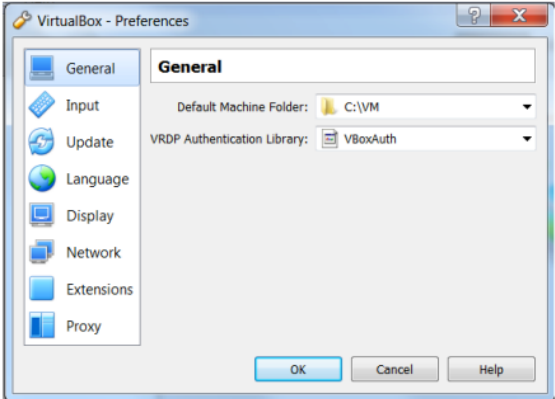
Пожалуйста выбирайте правильную версию VirtualBox, соответствующую установленной на вашей хост-машине ОС.

Загрузите образ с установленным компилятором **ESP8266_lubuntu_20141021.ova** по ссылке:

Baidu: <https://pan.baidu.com/share/init?shareid=3541602653&uk=190196792&third=15>

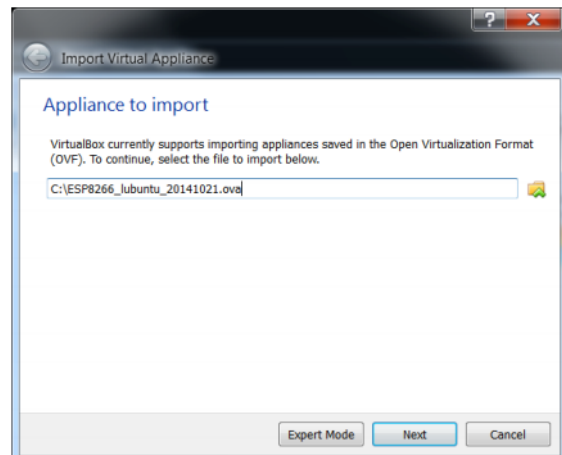
Пароль: qudl

Google: <https://drive.google.com/folderview?id=0B5bwBE9A5dBXaExvdDExVFNrUXM&usp=sharing>

Шаги	Результаты
<div>1. Запуск ОС Windows и установка виртуальной машины</div> <div><ul style="list-style-type: none">Двойной клик по VirtualBox-5.0.16.-105871-Win.exe и устанавливаем VirtualBox</div> <div>Замечание: Существуют различные версии VirtualBox. Мы используем для примеров версию Windows V5.0.16</div> <div><ul style="list-style-type: none">Двойной клик по Oracle_VM_VirtualBox.exe для запуска программы и система покажет главное меню →</div> <div>Совет: Виртуальная машина для ESP8266 занимает много места, пожалуйста зарезервируйте для неё достаточное место.</div>	<div></div>
<div>2. Настройка папки VirtualBox</div> <div><ul style="list-style-type: none">Создайте новую папку, например, C:\VMВыберите File -> Preferences, система покажет диалоговое окно →Выберите General, установите созданную папку (C:\VM) в качестве расположения по умолчанию (Default Machine Folder)</div>	<div></div>

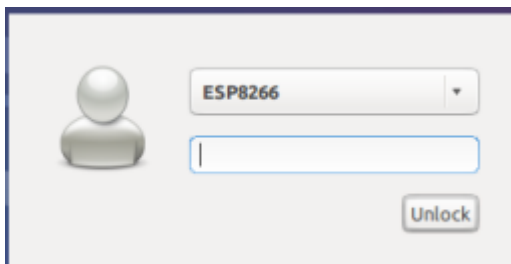
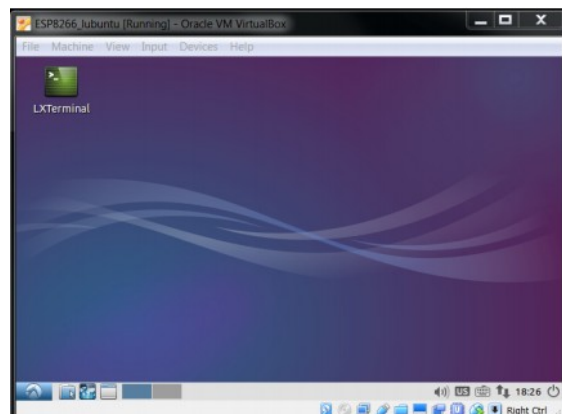
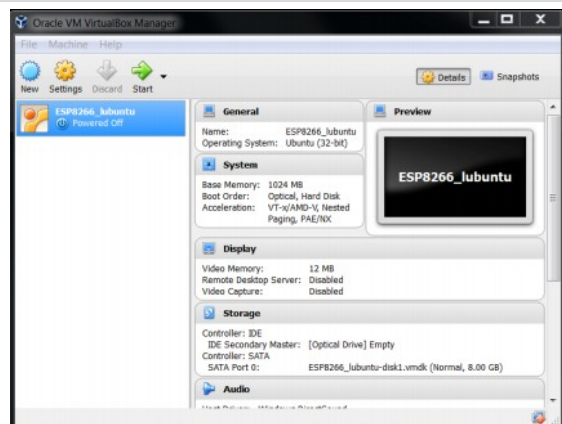
3. Импорт файла образа

- Выберите **File -> Import Appliance**, система покажет диалоговое окно →
- Выберите нужный файл образа, например, **C:\ESP8266_lubuntu_20141021.ova**, и нажмите **Next**.
- Нажмите **Import** чтобы принять настройки



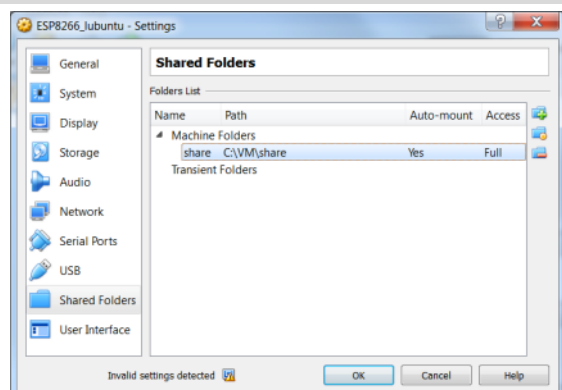
4. Запуск виртуальной машины

- После импортирования вы увидите виртуальную машину с именем **ESP8266_lubuntu** →
- Дважды кликните **ESP8266_lubuntu** или **Start** для запуска виртуальной машины
- После запуска, система покажет виртуальную машину **ESP8266_lubuntu** →
- Если всплывет показанное ниже диалоговое окно – введите пароль **espressif**.



5. Создание расшаренной папки

- Создайте в хостовой системе новую папку, с именем: **C:\VMshare**
- Выберите **Machine -> Settings -> Shared Folders...**, после чего вам будет показано диалоговое окно, такое как справа →
- Выберите созданную (общую для хоста и виртуальной машины) папку **C\VMshare**



3.3.2. Инструмент для загрузки прошивок

Пользователи могут скачать ESP8266 DOWNLOAD TOOL по адресу:

<http://www.espressif.com/support/download/other-tools>

4. Карта флеш-памяти

В этой главе рассказано о картах распределения памяти прошивок FOTA и Non-FOTA в чипах флеш-памяти различной ёмкости. Пользователи могут модифицировать карту при необходимости.

Карты памяти показаны на рисунке 4-1

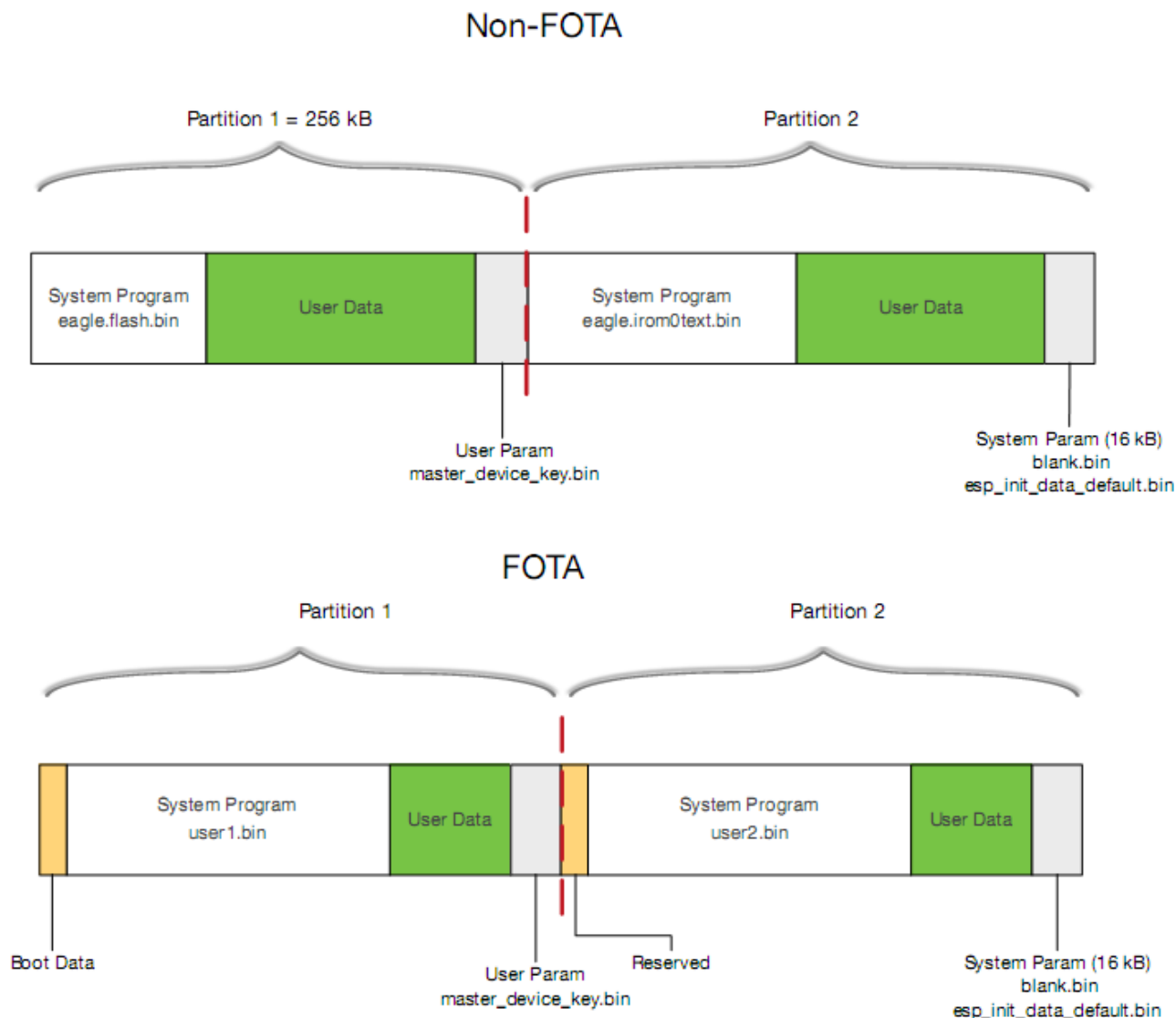


Рисунок 4-1. Карты памяти

Замечание:

О прошивке ESP8266 читайте в **1.3 ESP8266 FW**.

- **System Program:** в этой области хранится прошивка, необходимая для запуска системы.
- **User Data:** если системные данные не занимают всю флеш-память, то оставшиеся области могут использоваться для хранения данных пользователя.
- **User Param:** адрес этой области могут определять пользователи. В IoT_Demo под область пользовательских параметров отведены четыре сектора, начинающиеся с 0x3C000. Пользователи могут назначать для этой области любые свободные адреса.

- **master_device_key.bin**: в IoT_Demo этот файл расположен в третьем секторе области пользовательских параметров
- **System Param**: эта область занимает последние 4 сектора флеш-памяти
 - **blank.bin**: адрес загрузки – второй с конца сектор флеш-памяти
 - **esp_init_data_default**: адрес загрузки – четвертый с конца сектор флеш-памяти
- **Boot Data**: располагается в Partition 1 прошивки FOTA и хранит данные связанные с FOTA
- **Reserved**: зарезервированная область в Partition 2 прошивки FOTA, соответствует области **Boot Data** в Partition 1 прошивки FOTA.

Замечания:

- Каждый сектор флеш-памяти имеет размер 4 KB (0x1000).
- Детальное описание адресов смотрите в главах 4.1.1 Карта флеш-памяти и 4.2.2 Карта флеш-памяти

4.1. Non-FOTA

4.1.1. Карта флеш-памяти

Для чипов флеш-памяти различных размеров, предельный объем памяти для сохранения **eagle.irom0text.bin** составляет 200 kB. Пользователи могут изменить предельный объем, модифицировав файл **ESP8266_NONOS_SDK/ld/eagle.app.v6.ld**.

Вы можете модифицировать поле **len** в **irom0_0_seg** как показано на рисунке 4-2.

```
MEMORY
{
    dport0_0_seg :                org = 0x3FF00000, len = 0x10
    dram0_0_seg :                 org = 0x3FFE8000, len = 0x14000
    iraml_0_seg :                 org = 0x40100000, len = 0x8000
    irom0_0_seg :                 org = 0x40240000, len = 0x32000
}
```

Рисунок 4-2. Расположение поля **len**

Замечание:

- **len** – это просто размер области в 16-ричном виде, так, например, 0x32000=204800 байт = 204800/1024 = 200 килобайт.

В Таблице 4-1 даны предельные размеры **eagle.irom0text.bin** при различных размерах **len**.

Таблица 4-1. Карта памяти Non-FOTA (единицы измерения: kB)

Ёмкость флеша	eagle.flash.bin	eagle.irom0text.bin	User data	len	User/System Param
512	≤ 64	≤ 240	≥ 176	0x3C000	16
1024	≤ 64	≤ 752	≥ 176	0xBC000	16
2048	≤ 64	≤ 768	≥ 176	0xC0000	16
4096	≤ 64	≤ 768	≥ 176	0xC0000	16

Замечание:

В настоящее время максимальный размер области **System Param** для ESP8266 может составлять только 1024 kB.

4.1.2. Адреса загрузки

В Таблице 4-2 перечислены адреса загрузки для прошивки Non-FOTA.

Таблица 4-2. Адреса загрузки для прошивки Non-FOTA (единицы измерения: kB)

Бинарный файл	Адреса загрузки для флеш-памяти разного размера			
	512	1024	2048	4096
master_device_key.bin		0x3E000		
esp_init_data_default.bin	0x7C000	0xFC000	0x1FC000	0x3FC000
blank.bin	0x7E000	0xFE000	0x1FE000	0x3FE000
eagle.flash.bin		0x00000		
eagle.irom0text.bin		0x40000		

4.2. FOTA

4.2.1. Карта флеш-памяти

В Таблице 4-3 показаны размеры различных частей прошивки FOTA.

Таблица 4-3. Карта памяти FOTA (единицы измерения: kB)

Ёмкость флеша	boot	user1.bin	user2.bin	User/System Param	User data
512	4	≤ 236	≤ 236	16	≥ 0
1024	4	≤ 492	≤ 492	16	≥ 0
2048 (Раздел 1 = 512)	4	≤ 492	≤ 492	16	≥ 1024
2048 (Раздел 1 = 1024)	4	≤ 1004	≤ 1004	16	≥ 0
4096 (Раздел 1 = 512)	4	≤ 492	≤ 492	16	≥ 3072
4096 (Раздел 1 = 1024)	4	≤ 1004	≤ 1004	16	≥ 2048

4.2.2. Адреса загрузки

В Таблице 4-4 перечислены адреса загрузки для прошивки FOTA.

Таблица 4-2. Адреса загрузки для прошивки FOTA (единицы измерения: kB)

Бинарный файл	Адреса загрузки для флеш-памяти разного размера					
	512		1024		2048	
	512	1024	512+512	1024+1024	512+512	1024+1024
master_device_key.bin	0x3E000	0x7E000	0x7E000	0xFE000	0x7E000	0xFE000
esp_init_data_default.bin	0x7C000	0xFC000	0x1FC000		0x3FC000	
blank.bin	0x7E000	0xFE000	0x1FE000		0x3FE000	
boot.bin			0x00000			
user1.bin			0x01000			
user2.bin	0x41000	0x81000	0x81000	0x101000	0x81000	0x101000

Замечания:

- В случае с прошивкой FOTA нет необходимости загружать файл **user2.bin**, он предназначен для обновления прошивки через облачный сервер.
- Детали функционирования прошивки FOTA можно найти в документе **ESP8266 Non-OS SDK Upgrade Guide**.

5. Компилирование SDK

Замечания:

- В этой главе описано, как скомпилировать SDK на примере проекта **ESP8266_NONOS_SDK/examples/IoT_Demo**.
- Проект **IoT_Demo** может быть настроен в качестве одного из трех устройств: Умный Светильник, Умная Розетка или Датчик. Настройка задаётся в файле **/examples/IoT_Demo/include/user_config.h**. В одно и то же время может быть выбрано только какое-то одно устройство. По умолчанию выбран Умный Светильник.

5.1. Подготовка

5.1.1. Модификация файлов SDK

Замечание:

Пользователям необходимо модифицировать файлы SDK при использовании прошивки FOTA.

- Запустите ОС Windows
- Модифицируйте файлы в **ESP8266_NONOS_SDK/examples/IoT_Demo/include** в соответствии с размером вашей флеш-памяти.
 - Модифицируйте **#define PRIV_PARAM_START_SEC** в файлах **user_light.h** и **user_plug.h**.

```
/* NOTICE !!! ---this is for 512KB spi flash.*/  
/* You can change to other sector if you use other size spi flash. */  
/* Refer to the documentation about OTA support and flash mapping*/  
#define PRIV_PARAM_START_SEC      0x3C  
#define PRIV_PARAM_SAVE          0
```

- Модифицируйте **#define ESP_PARAM_START_SEC** в файле **user_esp_platform.h**.

```
/* NOTICE---this is for 512KB spi flash.  
 * you can change to other sector if you use other size spi flash. */  
#define ESP_PARAM_START_SEC      0x3D
```

В Таблице 5-1 перечислены значения, которые нужно вписать.

Таблица 5-1. Значения для указанных выше файлов (единицы измерения: kB)

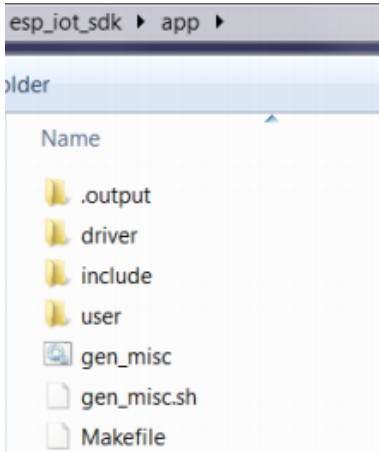
Значение по умолчанию (512)	Адреса загрузки для флеш-памяти разного размера					
	512	1024	2048		4096	
			512+512	1024+1024	512+512	1024+1024
0x3C	-	0x7C	0x7C	0xFC	0x7C	0xFC
0x3D	-	0x7D	0x7D	0xFD	0x7D	0xFD

Замечание:

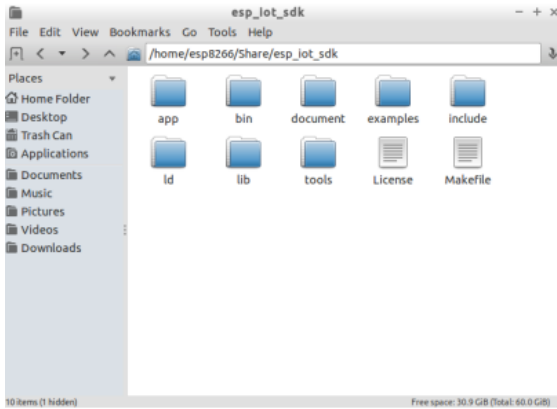
Пользователям, использующим флеш-память, объёмом 512 kB не нужно модифицировать файлы SDK.

5.1.2. Загрузка файлов SDK

1. Запустите ОС Linux.
2. Запустите LXTerminal на рабочем столе виртуальной машины.
3. Скопируйте файлы, которые нужно скомпилировать в расшаренную папку.

Шаги	Результаты
<ul style="list-style-type: none">• Копируем папку ESP8266_NONOS_SDK в расшаренную папку, например, C:\VM\share.• Копируем папку IoT_Demo в C:\VM\share\ESP8266_NONOS_SDK, как показано на рисунке справа →.	

4. Загрузите расшаренную папку.

Шаги	Результаты
<ul style="list-style-type: none">• Выполните <code>./mount.sh</code>• Введите пароль <code>espressif</code>. Загрузка расшаренных файлов завершена.• Откройте расшаренную папку ESP8266_NONOS_SDK в виртуальной машине и убедитесь, что загрузка успешно завершена<ul style="list-style-type: none">◦ если всё в порядке – папка содержит файлы как на картинке справа →◦ если нет – папка будет пустой и нужно будет повторить шаг загрузки ещё раз	

Замечание:

Если вы используете *RTOS SDK*, пожалуйста продолжайте выполнять следующие шаги; если вы используете *Non-OS SDK*, пропустите шаг 5.

5. Пропишите переменные среды `PATH` для указания на SDK и бинарники

```
export SDK_PATH=~/.share/ESP8266_RTOS_SDK
export BIN_PATH=~/.share/ESP8266_RTOS_SDK/bin
```

Замечание:

Вы можете добавить эти пути в файл **.bashrc**, в противном случае вам нужно будет повторять шаг 5 каждый раз после перезагрузки компилятора.

5.2.1. Компиляция ESP8266 NONOS SDK v0.9.5 и более поздних

```
cd /home/esp8266/Share/ESP8266_NONOS_SDK/app
./gen_misc.sh
```

gen_misc.sh version 20150511
Please follow below steps(1-5) to generate specific bin(s):

```

graph TD
    FOTA{FOTA?} -- N --> S1_2[2]
    FOTA -- Y --> NewVer{New version?}
    NewVer -- N --> S1_0[0]
    NewVer -- Y --> S1_1[1]
    S1_0 --> FirstTime{First-time usage?}
    S1_1 --> FirstTime
    FirstTime -- N --> S2_2[2]
    FirstTime -- Y --> S2_1[1]
    S2_2 --> S3_2[2]
    S2_1 --> S3_2
    S3_2 --> S4_0[0]
    S4_0 --> S5_3[3]
    S5_3 --> End(( ))
  
```

STEP 1: choose boot version
(0=boot_v1.1, 1=boot_v1.2+, 2=none)
enter(0/1/2, default 2)

STEP 2: choose bin generate
(0=eagle.flash.bin+eagle.rom0text.bin, 1=user1.bin, 2=user2.bin)
enter (0/1/2, default 0)

STEP 3: choose spi speed
(0=20MHz, 1=26.7MHz, 2=40MHz, 3=80MHz)
enter (0/1/2/3, default 2)

STEP 4: choose spi mode
(0=QIO, 1=QOUT, 2=DIO, 3=DOUT)
enter (0/1/2/3, default 0)

STEP 5: choose spi size and map
0= 512 KB(256KB+ 256KB)
enter (0/2/3/4/5/6, default 0)

Example Option

Рисунок 5-1. Компиляция SDK

Замечания:

- Опции, используемые в примере, помечены зелёным. Пользователи могут выбрать правильные опции, если это необходимо.
- Подробнее о прошивках FOTA и Non-FOTA читайте в **1.4 ESP8266 FW**.
- Опции 5 и 6 в шаге 5 поддерживаются только *sdk_v1.1.0 + boot 1.4 + flash download tool_v1.2* и выше.
- После компиляции **user1.bin** сначала выполните **make clean** для удаления временных файлов, сгенерированных при компиляции, и только потом компилируйте **user2.bin**
- По поводу карты флеш-памяти для шага 5 читайте **Главу 4 – Карта флеш-памяти**

3. В результате компиляции будут сгенерированы бинарники и на экране будет написан адрес, куда эти бинарники надо загрузить (лог сообщений показан ниже)

Generate user1.2048.new.3.bin successfully in folder bin/upgrade.

boot.bin----->0x000000

user1.2048.new.3.bin--->0xSupport boot_v1.2 and +
01000

!!!

Замечание:

Скомпилированные бинарники находятся в папке
/home/esp8266/Share/ESP8266_NONOS_SDK/bin

5.2.2. Компиляция ESP8266_NONOS_SDK_v0.9.4 и более ранних

Для ESP8266_NONOS_SDK_v0.9.4 и предыдущих версий процесс компилирования проводится так, как описано ниже:

1. Выполните **./gen_misc_plus.sh 1** для генерирования файла **user1.bin** в папке **/ESP8266_NONOS_SDK/bin/upgrade**
2. Выполните **make clean** для удаления временных файлов, сгенерированных при компиляции.
3. Выполните **./gen_misk_plus.sh 2** для генерирования **user2.bin** в папке **/ESP8266_NONOS_SDK/bin/upgrade**

Замечания:

- Детальное описание функционала FOTA читайте в документе **ESP8266 Non-OS SDK Upgrade Guide**
- ESP8266_NONOS_SDK_v0.7 и более ранние не имеют варианта FOTA

6. Загрузка прошивки

6.1. Процедура загрузки

1. Запустите ОС Windows
2. Дважды кликните ESP_DOWNLOAD_TOOL.exe чтобы запустить утилиту загрузки

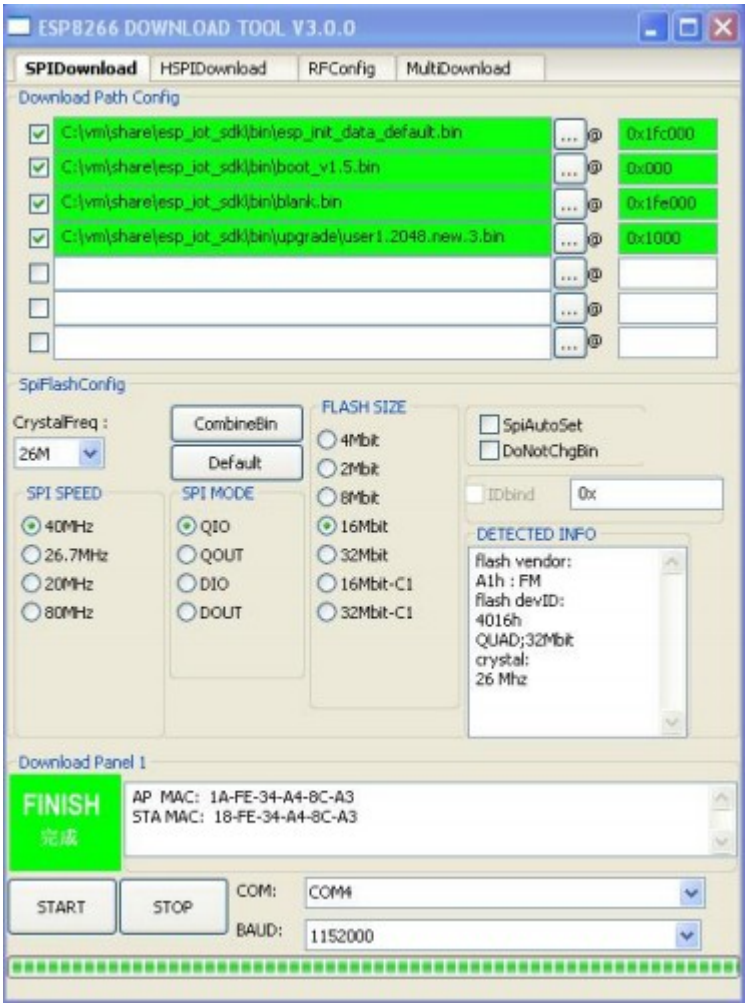



Рисунок 6-1. ESP8266 DOWNLOAD TOOL – SPIDownload

SPIDownload	Для загрузки в SPI-флеш
HSPIDownload	Для загрузки в HSPI-флеш
RFConfig	Настройки для инициализации RF
MultiDownload	Для мультиматеринских плат

3. Дважды кликните  в панели **Download Path Config** чтобы выбрать бинарники для загрузки. Установите соответствующие адреса загрузки в **ADDR**.
4. Сконфигурируйте страницу SPIDownload.

Замечание:

Бинарники для загрузки и соответствующие адреса меняются в зависимости от размера SPI флеш-памяти и фактических требований. Детали читайте в **Главе 4 – Карта флеш-памяти**.

Таблица 6-1. SPIDownload Configuration

Элемент настройки	Описание
SPI FLASH CONFIG	
CrystalFreq	Выбор частоты кварца в соответствии с используемым кварцем
CombineBin	Объединить выбранные бинарники в target.bin с адресом 0x0000
Default	Установить для SPI флеш-памяти настройки по умолчанию
SPI SPEED	Выбор скорости чтения/записи по SPI (максимум 80 MHz)
SPI MODE	<p>Выбор режима SPI в соответствии с используемой SPI флеш-памятью. Если используемая флеш-память поддерживает режим Dual SPI – выберите DIO или DOUT. Если флеш-память поддерживает режим Quad SPI – выберите QIO или QOUT</p> <p>Замечание: Если вы используете ISSI флеш-память – читайте <i>Appendix – Configure ISSI Flash QIO Mode</i></p>
FLASH SIZE	Выбор размера флеш-памяти в соответствии с используемым чипом
SpiAutoSet	<p>Мы рекомендуем не использовать SpiAutoSet, а конфигурировать настройку флеш-памяти вручную при необходимости.</p> <p>Если пользователи выберут SpiAutoSet, бинарники будут загружены в соответствии с картой памяти по-умолчанию. Карты памяти для флешек 16 Mbit и 32 Mbit будут 512 Kbyte + 512 Kbyte</p>
DoNotChgBin	<ul style="list-style-type: none"> Если пользователи выберут DoNotChgBin, то используемая рабочая частота флеш-памяти, режим и карта флеш-памяти будут основаны на конфигурации при компиляции Если пользователи не выберут DoNotChgBin, то используемая рабочая частота флеш-памяти, режим и карта флеш-памяти будут основаны на конфигурации, указанной в компиляторе
Download Panel	
START	Кликните START для начала загрузки. Когда загрузка завершится – в зелёной области слева появится надпись FINISH
STOP	Кликните STOP чтобы прервать загрузку
MAC Address	Если загрузка успешна, система покажет MAC-адреса ESP8266 STA (станции) и ESP8266 AP (точки доступа)
COM PORT	Выбор com-порта для загрузки данных в ESP8266
BAUDRATE	Выбор скорости загрузки. Значение по умолчанию 115200.

- После загрузки, переключите GPIO0 Control на ESP-LAUNCHER в положение ближе ко внешней стороне платы и запитайте плату для включения рабочего режима.

6.2. Проверка log-файла

После загрузки прошивки, вы можете проверить лог, выведенный на терминал с помощью инструмента отладки через последовательный порт

Вы должны сконфигурировать следующие опции инструмента отладки через последовательный порт

Таблица 6-2. Настройка инструмента отладки через последовательный порт

Элемент настройки	Описание настройки
Protocol	Последовательный порт
Port Number	Установите номер порта в соответствии с подключенным устройством
Baud rate	<p>Скорость на которой запущено устройство, в соответствии с частотой кварца</p> <ul style="list-style-type: none"> • 69120 (24 М кварц) • 74880 (26 М кварц) • 115200 (40 М кварц) <p>Пример ESP8266 AT поддерживает по умолчанию скорость 115200. Пользователи не могут её изменять.</p> <p>Пример ESP8266 IOT Demo поддерживает по умолчанию скорость 74880. Пользователи могут её изменять.</p>
Data bit	8
Calibration	None
Flow control	None

6.2.1. ESP8266 IoT Demo

Если загружена прошивка ESP8266 IOT Demo, система в рабочем режиме покажет инициализационную информацию, включая версию SDK и т.д. «Finish» означает, что прошивка работает правильно.

```

SDK version:X.X.X(e67da894)
IOT VERSION = v1.0.5t45772(a)
reset reason: 0
PWM version: 00000003
mode: sta(18:fe:34:a4:8c:a3) + softAP(1a:fe:34:a4:8c:a3)
add if0
add if1
dhcp server start:(ip:192.168.4.1,mask:255.255.255.0,gw:192.168.4.1)
bcn 100
finish

```

6.2.2. ESP8266 AT

Если загружена прошивка ESP8266 AT или прошивка по умолчанию в ESP-LAUNCHER или ESP-WROOM-02, система напишет «Ready» в конце загрузки в рабочем режиме. Если ввести в терминале «AT», то система ответит «OK», что означает, что система работает правильно.

Замечание:

Скорость в прошивке AT принудительно сконфигурирована на 115200, а скорость ESP8266 по-умолчанию равна 74880. По этой причине в самом начале информация о инициализации системы будет отображаться как кракозябры. Это нормальный феномен, но в конце концов (после инициализации правильной скорости обмена) система должна показать «Ready». Больше информации по AT командам можно найти по ссылке ***ESP8266 AT Commands***.

6.3. Настройка инициализации RF (опционально)

Перед заливкой бинарников во флеш-память, пользователи могут модифицировать настройки инициализации RF в таблице **RF InitConfig**. Новый сгенерированный файл **esp_init_data_setting.bin** может быть загружен во флеш-память вместо **esp_init_data_default.bin**. Пользователи могут менять опции и параметры в настройках RF.

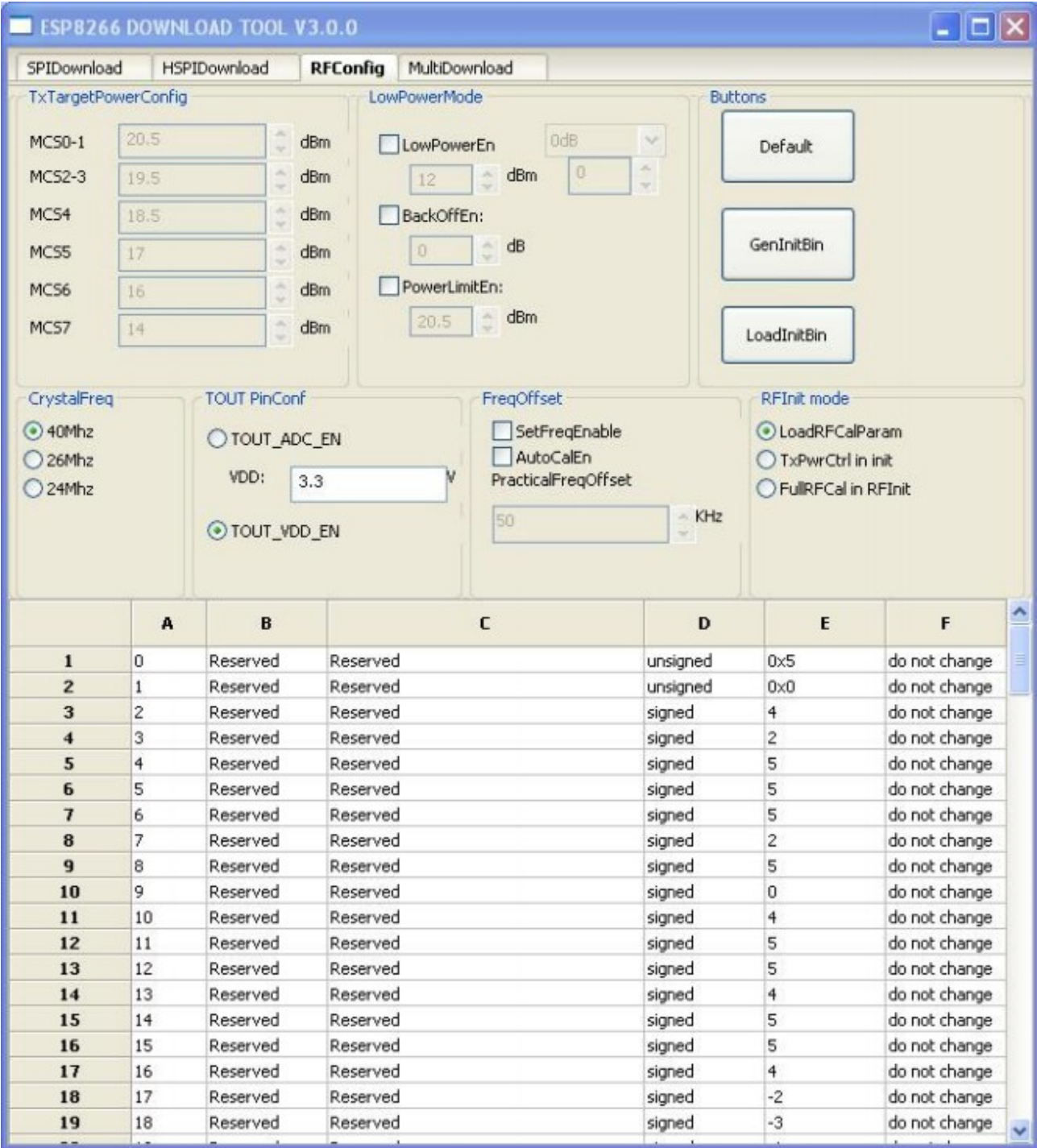


Рисунок 6-2. ESP8266 DOWNLOAD TOOL – RF InitConfig

6.3.1. Настройка опций RF InitConfig

Опции **RF InitConfig** перечислены в верхней части Рисунка 6-2. Описание опций приводится в Таблице 6-3.

Таблица 6-3. Настройка опций RF InitConfig

Опция	Описание
TxTargetPowerConfig	Пользователям не нужно настраивать эту опцию, она связана с опциями в LowPowerMode
LowPowerMode	<p>Настраивает режим низкого энергопотребления:</p> <ul style="list-style-type: none"> • LowPowerEn: включить режим низкого энергопотребления, установить значение мощности сигнала для всех скоростей передачи • PowerLimitEn: установить предел выходной мощности сигнала • BackOffEn: установить значение мощности сигнала в отвале (при отсутствии связи) для каждой скорости передачи данных <p>Замечание: Нельзя настроить LowPowerEn и PowerLimitEn в одно и то же время</p>
CrystalFreq	<p>Выбор частоты генератора в соответствии с используемым кварцем</p> <p>Замечание: Если при загрузке выбрать другую опцию, то конфигурация будет перезаписана (то есть вот этот выбор будет затёрт)</p>
TOUT PinConf	<p>Настройка пина TOUT в соответствии с актуальным статусом пина. Мы рекомендуем дефолтное значение.</p> <ul style="list-style-type: none"> • TOUT_ADC_EN: когда пин TOUT подключается ко внешнему контуру и измеряет внешнее напряжение с помощью встроенного АЦП. • TOUT_VDD_EN: когда пин TOUT не подключен (висит в воздухе) и через него можно измерить напряжение VDD33 с помощью функции uint16 system_get_vdd33(void) <p>Замечания:</p> <ul style="list-style-type: none"> • Нельзя выбрать одновременно TOUT_ADC_EN и TOUT_VDD_EN • Когда используется TOUT_ADC_EN, вы должны подключить актуальное напряжение к VDD3P3 пинам 3 и 4.
FreqOffset	<ul style="list-style-type: none"> • SetFreqEnable: установка сдвига частоты вручную <ul style="list-style-type: none"> ◦ PracticalFreqOffset: опция доступна при выборе SetFreqEnable. • AutoCalEn: установка частоты сдвига автоматически
RFInt mode	<p>Пользователи могут выбрать режим инициализации RF:</p> <ul style="list-style-type: none"> • LoadRFCalParam: во время инициализации RF, данные RF загружаются напрямую из флеш-памяти, без какой-либо калибровки. Это занимает около 2 мс и требует минимального начального тока. • TxPwrCtrl in init: во время инициализации RF производится только калибровка мощности Tx (передатчика), а другие данные загружаются из флеш-памяти. Это занимает около 20 мс и требует небольшого начального тока. • FullRFCal in RFin: во время инициализации RF калибруется всё. Это занимает 200 мс и требует большого начального тока.

6.3.2. Настройка параметров RF InitConfig

Параметры **RF InitConfig** перечислены в нижней части Рисунка 6-2. Описание параметров приводится в Таблице 6-4.

Таблица 6-4. Настройка параметров RF InitConfig

Параметр	Описание
A	Байт в <i>esp_init_data_setting.bin</i> (0 ~ 127 байт). Например, A=0 соответствует байту 0 в <i>esp_init_data_setting.bin</i>
B	Имя параметра. Пользователи не могут менять имя, если оно помечено как Reserved.
C	Имя параметра. Пользователи не могут менять имя, если оно помечено как Reserved.
D	Тип данных параметра конфигурации, включая знаковые и беззнаковые типы данных.
E	Шестнадцатиричное значение параметра конфигурации

Замечание:

Пожалуйста не модифицируйте параметры, помеченные как Reserved.

Ниже описано, как модифицировать параметры, хранящиеся в байтах со 112-го по 114-й. На Рисунке 6-3 перечислены начальные значения.

A	B	C	D	E	F
112	tx_param42	freq_correct_en	unsigned	0	bit[0]:0->do not correct fre
113	tx_param43	force_freq_offset	unsigned	0	signed, unit is 8khz
114	tx_param44	rf_cal_use_flash	unsigned	0	0: RF init no RF CAL, using

Рисунок 6-3. 112-й ~ 114-й байты параметров

Модифицирование параметров инициализации RF

Байт 114 предназначен для управления инициализацией RF после включения питания ESP8266. Различные варианты его значений перечислены в Таблице 6-5.

Замечание:

Поддерживается SDK версий ESP8266_NONOS_SDK_V1.5.3 и ESP8266_RTOS_SDK_V1.3.0 и более поздними.

Таблица 6-5. Модификация параметров RF InitConfig

Параметр	Описание
114-й байт = 0	Во время инициализации RF происходит только калибровка VDD33. Это занимает около 2 мс и требует минимального начального тока.
114-й байт = 1	Значение по умолчанию равно 1. Во время инициализации RF происходит калибровка VDD33 и мощности передатчика. Это занимает около 18 мс и требует небольшого тока инициализации.
114-й байт = 2	Аналогично случаю, когда 114-й байт = 0
114-й байт = 3	Во время инициализации RF происходит калибровка всего. Это занимает около 200 мс и требует большого тока инициализации.

Модифицирование Frequency Offset (смещения частоты)

112-й и 113-й байты служат для того, чтобы назначить смещение частоты. Параметры конфигурации описаны в Таблице 6-6.

Замечание:

Поддерживается SDK версий ESP8266_NONOS_SDK_V1.5.3 и ESP8266_RTOS_SDK_V1.3.0 и более поздними.

Таблица 6-6. Модификация параметров Frequency Offset

Параметр	Описание
112-й байт, значение по умолчанию = 3	
bit 0	Высший приоритет <ul style="list-style-type: none">bit 0 = 0: не изменять frequency offsetbit 0 = 1: изменять frequency offset
bit 1	Значение 0 означает, что bbp11 = 168 М. Пользователи могут менять смещение в обе стороны, как в положительную, так и в отрицательную. Это действие может влиять на производительность цифровой периферии, так что мы не рекомендуем его использовать. Значение 1 означает, что bbp11 = 160 М. Пользователи могут менять смещение в положительную сторону.
{bit 3, bit2}	Значение 0 означает, что чип будет отслеживать и назначать смещение частоты автоматически. Начальное значение смещения частоты равно 0. Значение 1 означает, что чип будет принудительно назначать значение смещения частоты равным значению байта 113, и не будет отслеживать и назначать смещение частоты автоматически. Значение 2 означает, что чип будет отслеживать и назначать смещение частоты автоматически. Начальное значение смещения частоты равно значению байта 113.
113-й байт, значение по умолчанию = 3	
113-й байт	Значение смещения частоты для принудительной коррекции или значение начального смещения частоты для автоматической коррекции. Тип данных – signed int8 (знаковое целое, 8 бит), единицы измерения (шаг) = 8 кГц.

6.3.3. Примеры конфигурации

Конфигурация байтов 112 и 113 зависят от специфических требований пользователя. Ниже мы предоставляем несколько примеров для справки:

- Модуль работает при постоянной температуре и не нуждается в подстройке частоты.**
 - Установите 112-й байт = 0, 113-й байт = 0
- Модуль работает при постоянной температуре и не нуждается в отслеживании и автоматической подстройке смещения частоты, но смещение частоты большое. В этом случае, мы рекомендуем принудительно задать смещение частоты.**
 - Если смещение частоты составляет +160 кГц (при постоянной температуре), можно установить 112-й байт = 0x07, 113-й байт = (256 – 160/8) = 236 = 0xEC

- Если смещение частоты составляет -160 кГц (при постоянной температуре), можно установить 112-й байт = 0x05, 113-й байт = $160/8 = 20 = 0x14$. Это действие может влиять на производительность цифровой периферии, так что мы не рекомендуем его использовать.

3. **Модуль работает при температурах от -40 °C до 125 °C как Умный Светильник и нуждается в отслеживании и автоматической подстройке частоты. Смещение частоты при постоянной температуре мало и пользователям не нужно задавать начальное смещение.**

- Установите 112-й байт = 0x03, 113-й байт = 0

4. **Модуль работает при температурах от -40 °C до 125 °C как Умный Светильник и нуждается в отслеживании и автоматической подстройке частоты. Смещение частоты при постоянной температуре большое и пользователям нужно задавать начальное смещение.**

- Если смещение частоты составляет +160 кГц (при постоянной температуре), можно установить 112-й байт = 0x0B, 113-й байт = $(256 - 160/8) = 236 = 0xEC$
- Если смещение частоты составляет -160 кГц (при постоянной температуре), можно установить 112-й байт = 0x09, 113-й байт = $160/8 = 20 = 0x14$. Но это действие может влиять на производительность цифровой периферии, так что мы не рекомендуем его использовать.

Мы рекомендуем пользователям ссылаться на пример 3.

Когда настройка инициализации RF будет закончена – кликните кнопку **GenInitBin** для генерирования файла *esp_init_data_setting.bin*.

Дополнительно пользователи могут кликнуть кнопку **Default** чтобы установить значение по умолчанию или кликнуть кнопку **LoadInitBin** чтобы импортировать двоичный файл конфигурации.

I. Приложение – настройка режима QIO для ISSI-флеш

Замечание:

При загрузке нужно выбирать режим DIO или DOUT, иначе могут произойти ошибки.

При использовании для ISSI-флеш режима QIO, вы должны изменить первые два байта в файле *blank.bin* как показано в Таблице I-I. Во время инициализации ESP8266 автоматически проверит первые два байта файла *blank.bin* и переключится в режим QIO для чтения ISSI-флеш. Структура файла *blank.bin* показана ниже.

```
struct boot_hdr{
    char user_bin:2;    //low_bit
    char boot_status:1;
    char to_qio:1;
    char reverse:4;
    char version:5;     //low bit
    char test_pass_flag:1;
    char test_start_flag:1;
    char enhance_boot_flag:1;
}
```

Таблица I-I. Конфигурация blank.bin

Параметр	Описание
Без вторичного бут-загрузчика	Измените значение to_qio на 0
Со вторичным бут-загрузчиком	<p>Измените значение use_bin на 0 и to_qio на 0. Измените версию на текущую версию бута.</p> <p>Примеры:</p> <p>Если вы используете вторичный <i>boot_v1.5.bin</i>, измените первые два байта FF FF в файле <i>blank.bin</i> на F4 E5.</p>

Замечание:

Вам не нужно модифицировать бинарники при использовании режимов ISSI-флеш DIO или DOUT.