Raptor Software Setup Guide

Xunxing DIAO, Hongling SHI and Kun Mean HOU
University Clermont Auvergne (UCA), LIMOS UMR 6158 CNRS, France

Table of Contents

List of figures

I. Introduction

Raptor platform is based on a three layer systems: end-device, local server and remote server. End-device samples environmental data (air temperature, air relative humidity, O3 and NO2) and sends sensory raw data through IEEE802.15.4 to a local server. Thereafter, the local server performs a first treatment and sends the results to the remote server (AWS 'Amazone Web Services' server).

In term of software, to be operational Raptor platform needs the following firmware and software:

- End-device firmware: Bootloader, LwMesh and Toolchains
- Local server: Linux (RASPBIAN).

II. LwMesh and Toolchains for End-device

To install and flash Raptor End-device firmware two methods may be used, local and remote.

II.1. Local build method

The firmware package used on End-dive and Coordinator is LwMesh 1.2.1 (AVR's Lightweight Mesh ZigBee mesh networking stack). To use that, users need to install the toolchains first.

II.1.1. Install Toolchains to compile and flash Raptor firmware

First you need to install the following software toolchains:

- C compiler
- LwMesh firmware.

II.1.1.1 Linux desktop: Ubuntu or Debian

Run:

```
$sudo apt-get update
$sudo apt-get install gcc-avr binutils-avr gdb-avr avr-libc avrdude
```

II.1.1.2. For MacOS:

1. Follow the link below to install "Brew"
http://brew.sh/

2. To install the entire AVR toolchain:

$brew tap larsimmisch/avr
$brew install avr-libc

3. To install Avrdude for AVR ROM and EEPROM Downloader/Uploader
$brew install avrdude --with-usb

4. To install AVRFuses as a simple interface of Avrdude
Download and install by: http://www.vonnieda.org/AVRFuses/AVRFuses_1.4.8.zip

5. Optional: to install other GNU core utilities (e.g. gmkdir):
$brew install coreutils

II.1.1.3. LwMesh Configuration Files

The LwMesh source codes are in the folder of "LwMesh/source". For basic usage, there is only one important file:

/<LwMesh>/source/apps/WSNDemo/config.h

In this file, users can change node address (APP_ADDR), caption (APP_CAPTION), sample interval (APP_SENDING_INTERVAL), etc.

If APP_ADDR is zero, it will be set as a Coordinator. If APP_ADDR is larger than 0x8000, it is an end-device.

For Raptors, the other parts are all set. No need to change.

II.1.1.4. Build LwMesh

Make sure the toolchain has been installed. Then, run the following command to build the LwMesh:

Make -f Makefile_Rcb128rfa1_ATmega128rfa1 all

After that, a binary file will be generated in:

/<LwMesh>/source/apps/WSNDemo/make/Debug/WSNDemo.srec

II.1.1.5. Flash the binary to Raptor

The finished Raptor normally comes with a bootloader. If not, users need to install it:

Bootloader:

Step 1: Use JTag with AVRFuses (or Atmel Studio on Windows) to change the Fuse to 0xFF, 0x9C, 0x62, as following figure:

Option	Common bootloader
BODLEVEL	Disabled
OCDEN	Disabled
JTAGEN	Enabled
SPIEN	Enabled
WDTON	Disabled
EESAVE	Disabled
BOOTSZ	Boot Flash size=1024 words start address=$FC00
BOOTRST	Enabled
CKDIV8	Enabled
CKOUT	Disabled
SUT_CKSEL	Int. RC osc.; Start-up time: 6 CK + 65ms
Resulting fuse bytes	***0xFF 0x9C 0x62***

Step 2: Use AVRFuses (or Atmel Studio on Windows) to write a bootloader on Raptor by JTag. The bootloader is in:

"/<LwMesh>/Bootloader_Atmega128rfa1.hex"

Flash:

There is a Python script for that at:

/<LwMesh>/flashWriter.py

Step 1: Set the jump to USB<-> uSu on Raptor:

Step 2: Connect Edu board with a USB to serial port cable. Check the device id, and change the following line in flashWriter.py to the right one:

For Ubuntu, normally:
MY_PORT = "/dev/ttyUSB0"

For MAC OSX, normally:

MY_PORT = “/dev/tty.usbserial-(xxx)”

Step 3: Run “python /<LwMesh>/flashWriter_PcToEdu.py”. The program will do the LwMesh building and flash rewriting. Just follow the command notice to click the RESET button when needed.

II.2. Remote build method

Remote build method is user-friendly. Users connect to our remote server and configure the parameters of Raptor End-device and coordinator. After compiling, the binary codes of firmware are downloaded automatically ready to be flashed on the uSu-Edu board: Raptor End-device or Raptor coordinator according to the configuration setting up by the user.

Remote configuration and build

First connect to our remote server. A login and password are required. You can send email to peterdiao@gmail.com to obtain one. Thereafter, a configuration set up menu is displayed. User can build remotely the firmware (Figure 1 and Figure 2).

Figure 1. Remote build configuration

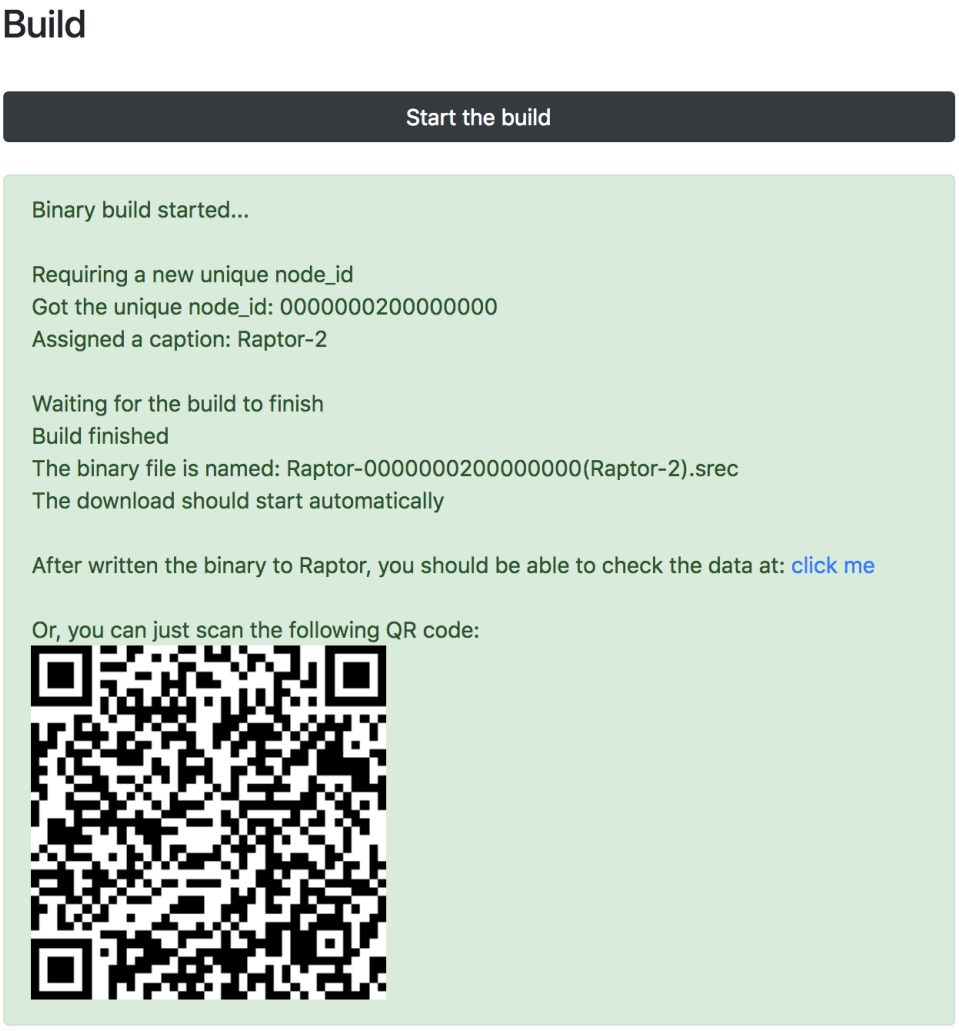

Figure 2. Raptor binary code generation

Then you just follow the step in the section “Flash” to write the downloaded binary to the Raptor. Please notice that by local or remote method, you need to configure the End-dive and Coordinator in the same IEEE 802.15.4 channel and PAN ID.

III. Local server

Raspberry Pi

Assume users has install LwMesh on a Coordinator and an end-device, the next step is to install software on a Raspberry Pi. The Coordinator will need to plug in a Raspberry Pi to work as a local server and the Raspberry Pi will forward the packets to our remote server.

The image file of Raspberry Pi can be downloaded it from:

http://www.raptor.im/store/pi.zip

Users need a micro SD card to install the image file. For Linux and MacOS, unzip the pi.zip then run the following command:

sudo dd if=/<path to> /pi.img of=/dev/<the sd card writer dev id> bs=1m

The image file is built based on "2018-03-13-raspbian-stretch-lite". The main Raptor-related software are a local server and a remote updater. It is a plug and play software that requires no user intervention. The software will upload the packets to our web server, and later users read it by web APIs.

IV. Web API

The web API is developed by using Node.js and MongoDB. It is based on the RESTful design. Mainly users will only need to use the GET functions:

GET on http://www.raptor.im/api/v2/:node_group/:node_id/items

It is used to get items (i.e. messages from Raptors in our case) from a database collection.

URL Parameters

Required:

- `:node_group` for Raptor project, it is "Raptor"
- `:node_id` the unique id of the Raptor

Optional (after the "?" in the URL):

- limit=[number] (default is 10) the number of the item will be sent back
- countonly=[0|1] (default is 0) if countonly=1, only the total number of items will be sent back