R-IoT

General Description

The R-IoT also embeds a 9 axis digital sensor featuring a 3-axis accelerometer, a 3-axis gyroscope and a 3-axis magnetometer, allowing for instance the onboard computation of the absolute orientation of the module in space. The sensor is attached to the SPI port to grab the 16 bit motion data at high speed.

The sensors data are locally processed, analyzed and streamed over WiFi (UDP, Open Sound Control) by the module using its firmware, a program compiled using the Energia tool chain.

The R-IoT module is based upon the CC3200 chip from Texas Instrument. Its core feature is to be compatible with Energia, a branch of the Arduino IDE that allows programming TI processors with the easiness of the Arduino look & feel.

Just like the Arduino system for Atmel microcontrollers, Energia[1] is essentially a code warper on top of the C API from Texas Instrument to program the core processor though R-IoT can still be programmed with TI tools chain & Code Composer.

The CC3200 is actually the combination of two ARM processors, one taking care of the WiFi modem and network stack, the second being the application processor running the user code.

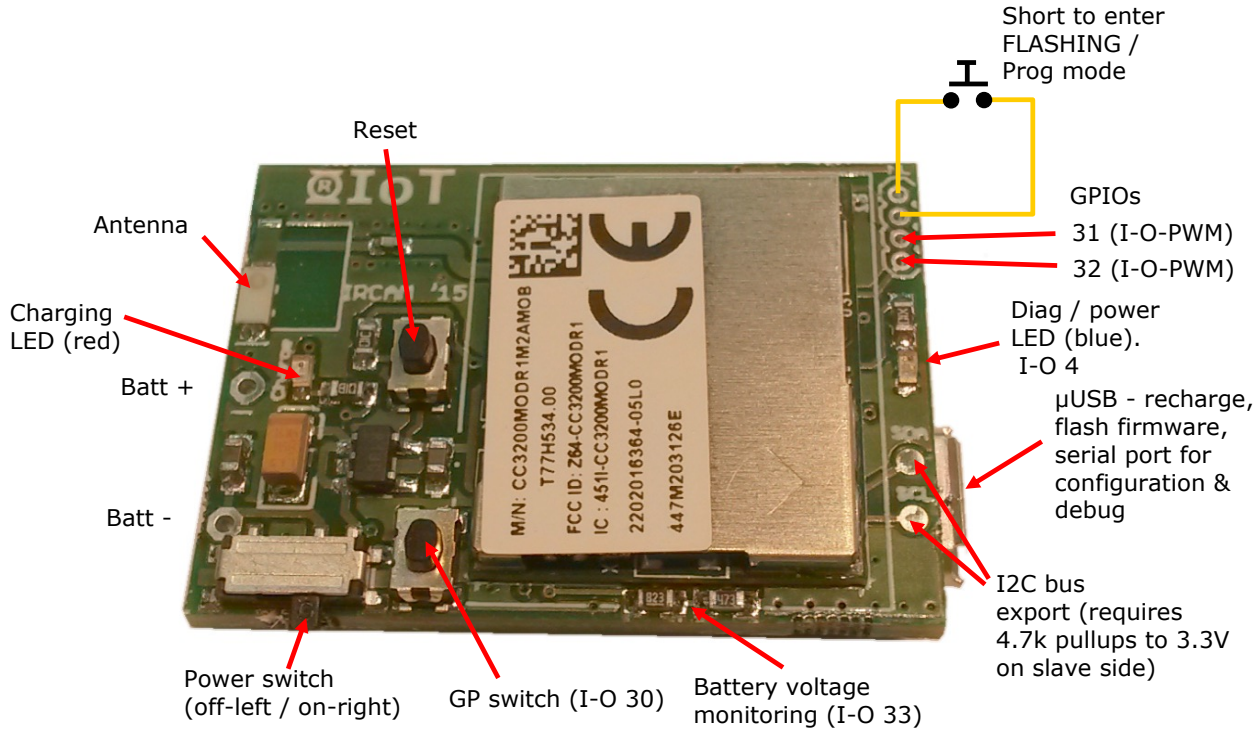
The CC3200 embeds a bootloader triggered upon reset / power on by an external momentary switch. The firmware is uploaded to the chip using a simple UART available as a USB serial port.

The R-IoT exports only some of the available I/O's from the CC3200 for matters of size / form factor. While not exactly a hacker friendly board, it remains small enough to be installed anywhere, on the body, on an instrument or in an object. With the additional I/Os which include 2 analog inputs and 1 PWM output, it can be adapted to various contexts.

The i2c bus is also exported allowing a complete chain of accessories to be talked to, from LED controllers to additional sensors or I/O extenders.

---

[1] http://energia.nu/



Specifications

- 34 x 23.5 mm – 7mm thick
- CC3200 processor – 80 MHz – 32 bits – 256 kB of RAM – 80 mA while transmitting WiFi.
- 2.4 GHz WiFi – Open Sound Control Oriented
- 9 DoF motion sensor LSM9DSO – 3D accel + gyro + magneto
- On-board web server for configuration
- On-board general purpose tactile switch
- 2 GPIO + 2 analog inputs (or GPIO) exported
- I2C bus SCL & SDA pins exported
- on-board li-ion / li-po charger via µUSB
- At least 6 hrs of runtime with a 16340 li-ion cell
- Battery voltage sample and report via WiFi
- USB UART for serial port debugging and configuration

Physical platform

We have developed the R-IoT platform using initially the Texas Instrument CC3200 launchpad, a development board for which Energia [2] has standardized and named / numbered all the I/Os based on the physical header and pinhead connectors of the dev board.

In order to remain 100% compatible with the former pin numbering, we kept the same numbering logic of the CC3200 I/Os in our code. The following picture details which I/Os are exported along with their #. That number is the same used to access to the physical pin from the program written in Energia.

As an example, using the blue LED pin as an output and turning it on in Energia would be done with those 2 lines:

```
pinMode(4, OUTPUT);
digitalWrite(4, HIGH);
```

R-IoT Sensor Fusion and Analysis

Sensor fusion

Calibration of the sensors and the Data fusion allowing for computing the quaternions and Euler angles are provided. The data fusion is implemented using the open-source code provided by Sebastian Madgwick[3]

R-IoT code repository

Several code examples dedicated to the R-IoT platform are available on Ircam's public GIT repository: https://github.com/ircam-rnd

The repository contains the main firmware, which achieves several analysis on the sensor data and allow the configuration of the module (IP address, UDP port, module ID) via a web server or the USB serial port.

Other simpler examples show how to write dedicated code for the R-IoT platform for specific motion analysis for instance.

Being a development platform, the R-IoT module role isn't bound to motion analysis or sensor data streaming. Using the additional I/Os and analog input and the WiFi modem, the unit can be turned into an efficient

---

[2] http://energia.nu/pin-maps/guide_cc3200launchpad/

[3] See http://www.x-io.co.uk/open-source-imu-and-ahrs-algorithms/ for more details

Internet Of Things (IoT) object, a miniature web server, a car alarm or a plant watering system.

Receive sensors data from the module

Below is an data receiving in max-msp. The module currently forges 3 OSC packets

/<ID>/raw <11 int data list> that contains the battery voltage, the GP switch state and the raw 9 channels from the motion sensor (all 16 bit)

/<ID>/quat <4 float data list> that contains the quaternion computation results based on Madgwick algorithm

/<ID>/euler <3 float data list> that contains the euler angles computed by the module from the quaternions mentioned above

/<ID>/analog <2 int data list> that contains the analog inputs digital conversion

Max abstraction Riot

Parse, resample and calibrate data received from the R-IoT board (Ircam's wireless inertial measurement unit sensor). The raw data is received using the OSC (OpenSoundControl) protocol.

input

[1] device ID (default value: 0)

[2] port number (default value: 8888)

[3] outputted resampling period (default value: 10 ms)

output

[1] acceleration (<float: x><float: y><float: z>), in **g**

[2] angular velocity (<float><float><float>), in deg/s

[3] magnetometer data (<float><float><float>), in G (= 100 µT)

[4] quaternions (<float><float><float><float>[-1 1])

[5] Euler angles (<float><float><float>), in deg

[6] GP switch state (<bool : 0 if pressed, 1 otherwise>)

[7] battery voltage (<float>), in V

[8] actual sampling period after resampling (<float>), in ms

components

Components

udpreceive : connects to the port where data are received, given in input

resampling : resamples the received data at the desired frequency, given in input

calibration :

(a) Gyroscope calibration

The board should be held completely still as the mean background noise is computed for all 3 axis simultaneously. It is then subtracted from the value received from the gyroscopes in output stream [2].

(b) Accelerometer calibration

The board should be slowly rotated so that all 6 faces of the board are successively oriented vertically to undergo gravitational acceleration. After this process, acceleration values have been comprised between -1 **g** and 1 **g** on all 3 axis. For a given axis, minimum and maximum accelerometer values are stored as calibration parameters and used to scale the output stream [1] accordingly.

For gravity to overcome acceleration components related to the in-hand manipulation of the board, as well as to avoid abrupt changes and high-frequency noise, the incoming acceleration data streams are heavily smoothed by lowpass filtering. As a consequence, it can take up to several seconds for calibration parameters to reach the correct value. A visual feedback (button) is provided to indicate whenever a parameter is updating. Typically, one should have the board oscillate around each vertical position for a few seconds, until the button doesn't blink anymore.

(c) store/recall and export/import calibration settings

There are 2 calibration parameters for each accelerometer axis and 1 for each gyroscope axis, giving a total of 9 calibration parameters that can be stored and recalled within the object **riot**, or exported and imported from a file.

visualisation :

(a) multisliders for data streams

(b) sampling period at reception and after resampling

(c) toggle for switch state

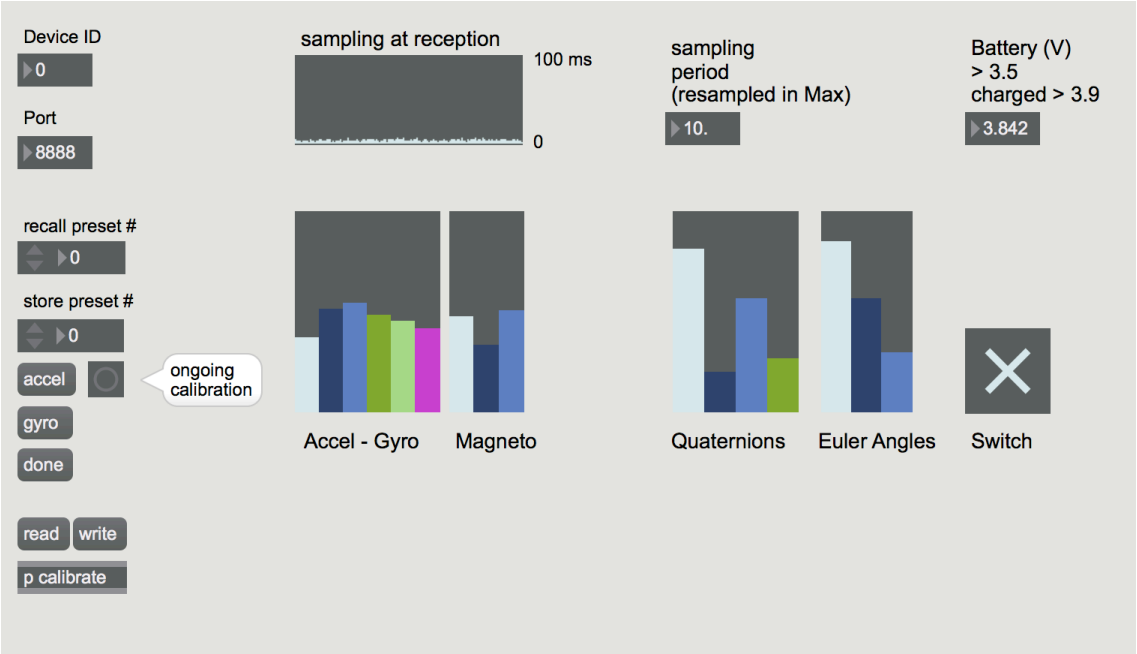


figure 1. Max Abstraction RIoT

Programming the R-IoT

Install the Energia IDE

1. visit http://energia.nu/ and look for the download section
2. download energia version 16
3. On windows, simply unzip the Energia folder on your main hard drive. On Mac, place the folder-program in the Application folder.

Install the USB serial port driver

The R-IoT unit uses a FTDI serial port. Install the drivers by visiting FTDI download section (look for VCP driver)

http://www.ftdichip.com/Drivers/VCP.htm

On windows, it will create a COM port. On Mac OS, the user must create the port by going in the network preferences (select standard NULL modem).

Customize the IDE

In order to compile the "full" firmware (currently named R-IoT 1.4), the default linker file used by the Energia tool chain must be modified as the reserved heap size / stack is too high when compiling big programs.

On windows:

- open the location of your Energia folder, such as C:\Program Files (x86)\energia-0101E0016\
- keep going into hardware\cc3200\cores\cc3200
- edit the file cc3200.ld

On Mac OS:

- open the location of your Energia folder, usually in the Application folder
- OSX applications are actually a folder. Right click on the app icon and select "show package contents"
- browse to Contents/Resources/Java/hardware/cc3200/cores/cc3200
- edit the file cc3200.ld

In the .ld file, simply edit the first line and change the heap size to 0x0008000 and save the file:

HEAP_SIZE = 0x0008000;

Use Energia IDE

Launch Energia. A blank sketch appears. A sketch is composed of two essential functions, setup() and loop() which is equivalent to the main function in traditional C language. Energia, just like Arduino uses C and C++, along with a basic API and set of classes to access the hardware in what became the Arduino standard.

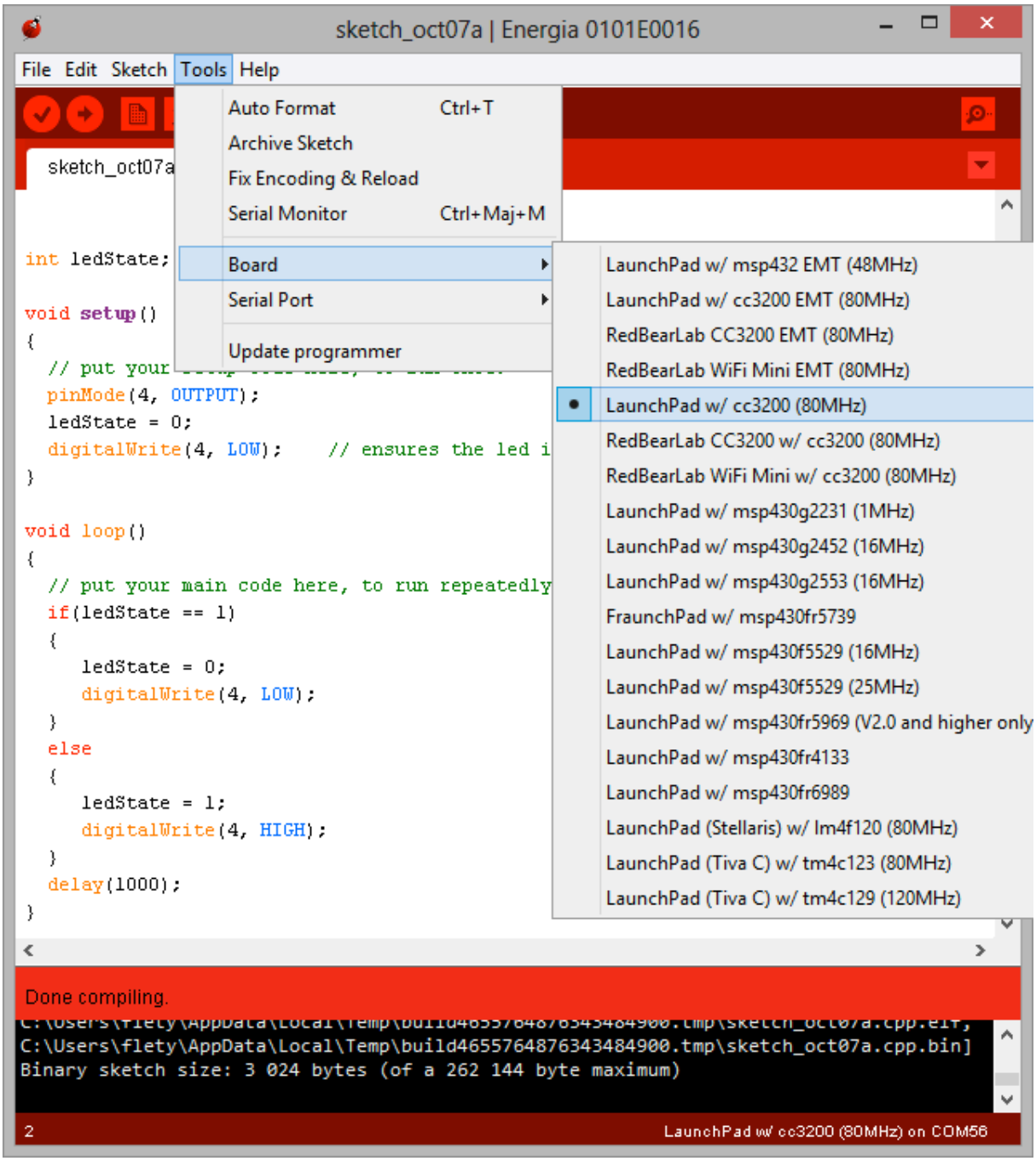
https://www.arduino.cc/en/Reference/HomePage

The setup function, called prior the main loop, is used to configure the module hardware, default behaviors and everything that requires some initialization before executing the main program.

Once returning from the setup function, the loop function is called repetitively and is equivalent to any traditional endless program loop used on embedded platforms (a while(1) statement within the main function).

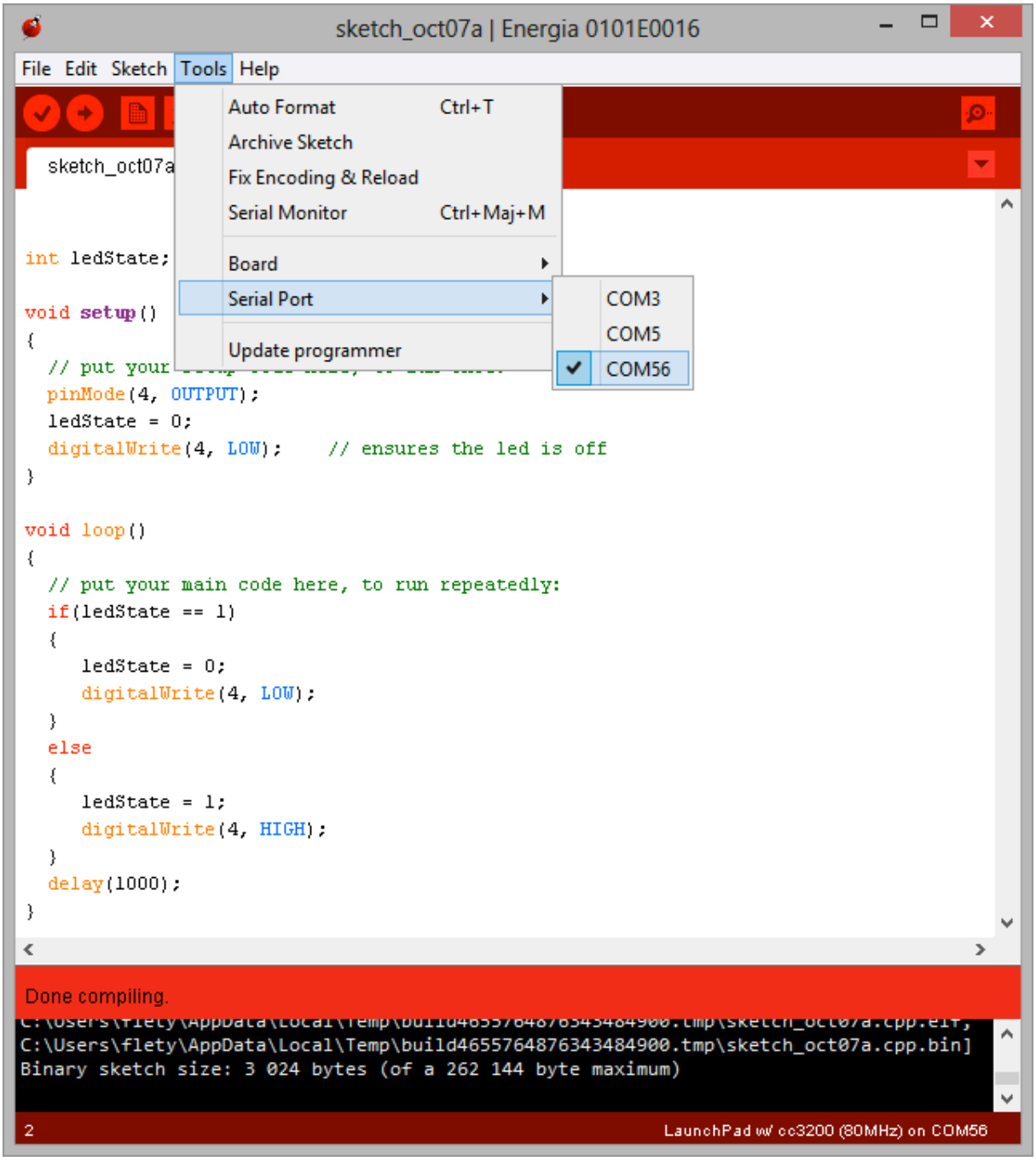
Below an example of blinking endlessly the blue LED of the module (I/0 #4):

Before compiling, select the proper target in the Tools -> Board menu and select the launchPad w/ CC3200 (80 MHz).



To compile to program, simply click on the left-most icon (tick).

To flash the code on the platform first plug the unit on a USB port then select the matching COM port of the R-IoT

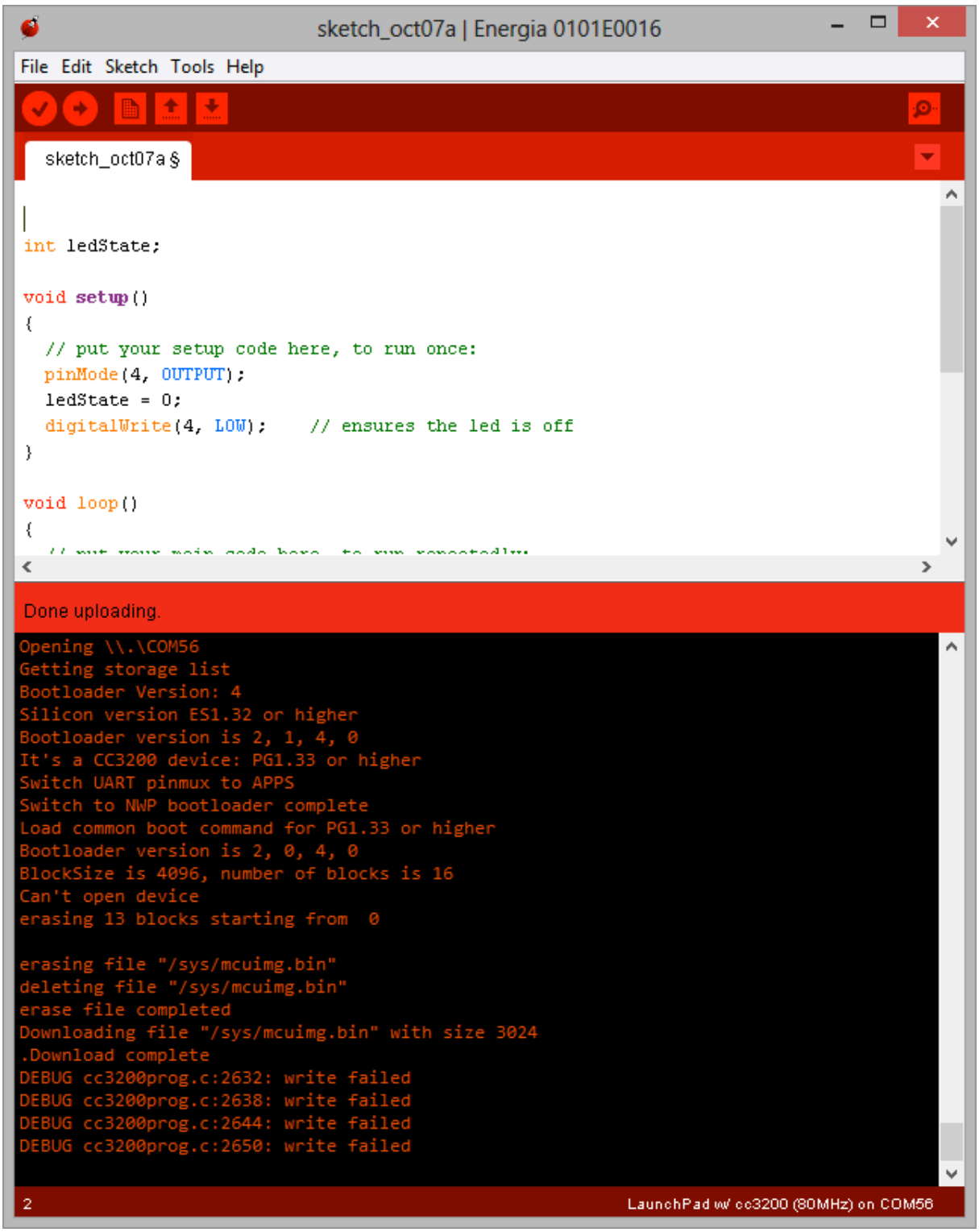


With the module powered on, even already executing the code present in the chip, keep the flashing switch pressed and click on the upload ircon (arrow). It will first compile the code then self reset the chip into bootloading mode. After a while, you'll get a done uploading message with the following log in the console of the IDE.

Sometimes, if the bootloader cannot be triggered properly, first click on the upload after pressing on both the flashing switch and the reset. Keep

the flashing switch pressed, and release only the reset switch once Energia moves from compiling to uploading stage in the log console.

The DEBUG write fail messages are normal, as the bootloading program tries also to set the chip in debug mode while our platform doesn't have any JTAG debugger / port exported.



Once there, the module can be reset (cycle the on-off switch, or press the reset onboard momentary switch) to leave flashing mode and execute the freshly uploaded code. With the above code, the blue LED should flash once per second.

WiFi Access point Configuration

While it's possible to connect the computer by WiFi to the AP, we strongly advise to use **wired Ethernet** as it maximizes the throughput and reduces both the latency and data jitter.
When taken straight out of the box, the AP is usually configured with DHCP enabled for clients and with a fixed IP address which can be (most of the time) 192.168.1.1 or 192.168.0.1 **or in the case or the mini TP-link 3G/Wifi router, 192.168.0.254.**

Set your computer in DHCP (aka "obtain an IP automatically") and connect it to the router by wired ethernet. Based on the address type you'll receive (either 1.xxx or 0.xxx) connect to either 192.168.1.1 or 192.168.0.1 or… RTFM ;-) Sometimes DHCP is disabled by default, and a manually set IP address must be assigned to the computer using the same scheme of the router's default address. In the case of the small TP-link pocket router above, the computer can be set to 192.168.0.100 to first access the admin page located on 192.168.0.254.

When accessing the AP, you'll be prompted for a login and password, the provided routers are left with **admin / admin**. Once in the administration page of the access point, there are only a few things to setup. All those things can be changed and tuned to match any equipment, we're just covering here the basics.

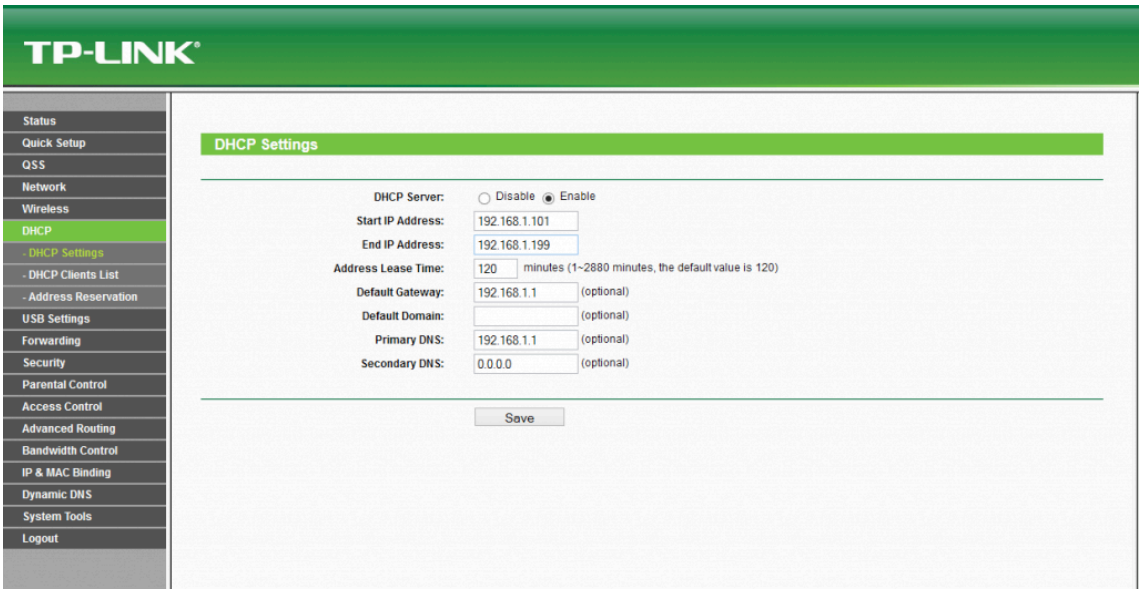
– The default configuration of R-IoT is to connect to a computer with the fixed IP address 192.168.1.100. If the router / AP configuration is set to

192.168.0.1, there will be an address mismatch, therefore go in the Network -> LAN page and change the AP address to 192.168.1.1 or 192.168.1.254 which is kind of standard

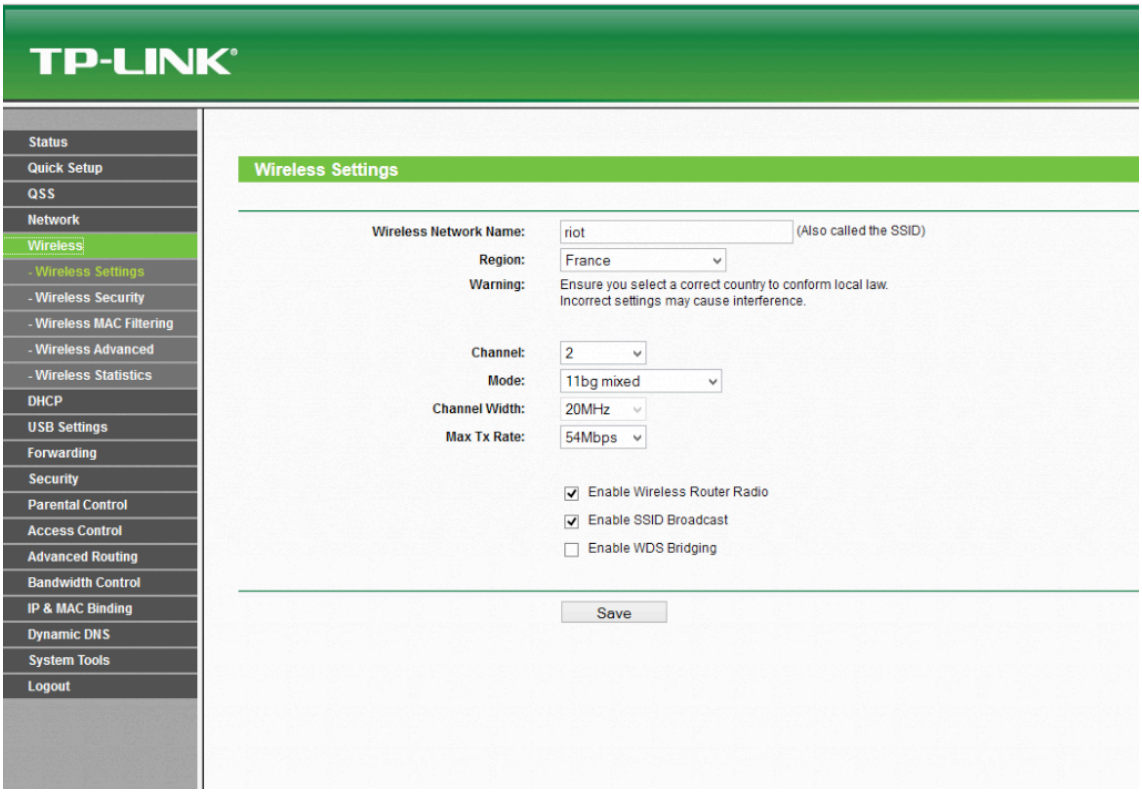
$

That will usually require a reboot of the AP, proceed and log in once again.

– Tune the DHCP settings. Most of the time the DHCP will be active on addresses from 192.168.1.100 to 199. Change this from 110 to 150, anything to avoid having the default destination IP address (192.168.1.100) to be in the DHCP lease pit. Alternatively you can leave it in the DHCP area and reserve the address based on the MAC address of your computer (see DHCP address reservation)



– Finally go in the Wireless settings, change the SSID of your network. Each provided module has a sticker on with with its serial #. Each number matches a SSID name "riotXX" (module number 5 wants to connect to riot5). Select a wifi channel that is possibly different than your neighbor. Finally, go in the security section and disable encryption (WPA2 encryption is supported but not recommended on first use).



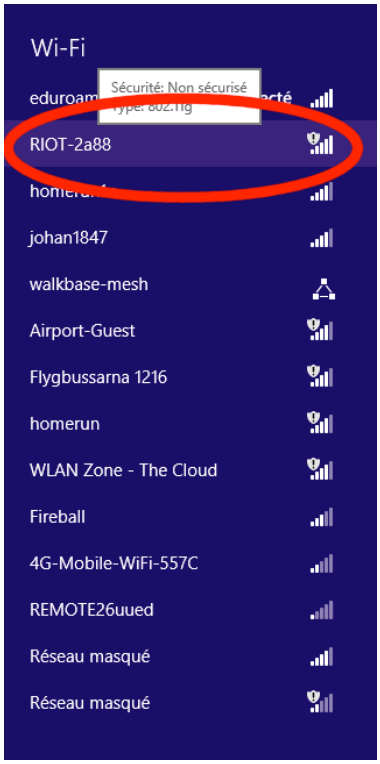
Once the router is configured with this 192.168.1.xxx address scheme and DHCP, the computer can be set to 192.168.1.100, the default destination address the R-IoT module sends data to.
For #MFT scandi, we had pre configured alls provided pocket

routers. The can be already accessed via 192.168.1.254 and have a SSID / wifi network name matching the # of the provided R-IoT module (riot3 for module #3 etc). Overall, there's no standard for naming your SSID, the only requirement is to have the R-IoT SSID name configuration matching the AP SSID.

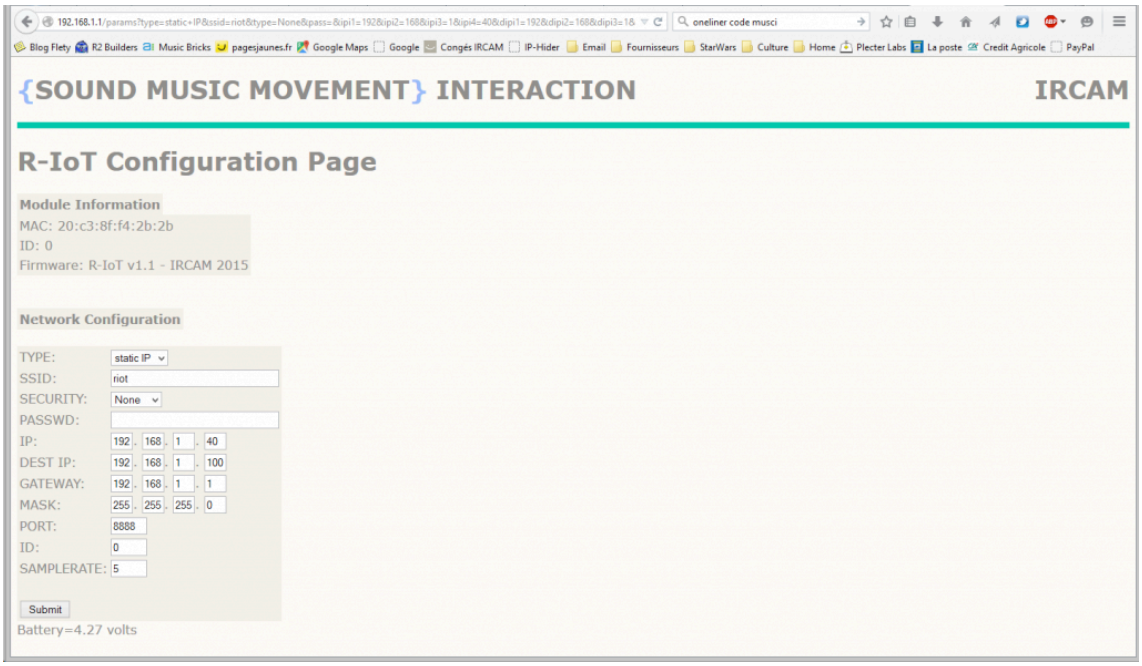
The WiFi radio channel can be adjusted in order to limit bandwith occupation. Each router has been assigned to a different channel (double check). Use a scanner or just use any WiFi survey app out there to diagnose the RF space around your spot and select the less busy channel.

Accessing the Web Server for configuring the module

- Turn the module off, press the GP switch, and power the module on while keeping the GP switch depressed
- The power Led will start blinking fast. When it becomes static / solid, the module is now a WiFi access point
- Set your computer in DHCP and lookup the wifi network proposed by the module, which will be RIOT-XXXX (4 digits, will change each time)



- Connect to the network (no security needed)
- Open your web browser and open the URL http:/192.168.1.1
- That should take you to the web page hosted by the module where you can configure its network behavior & parameters

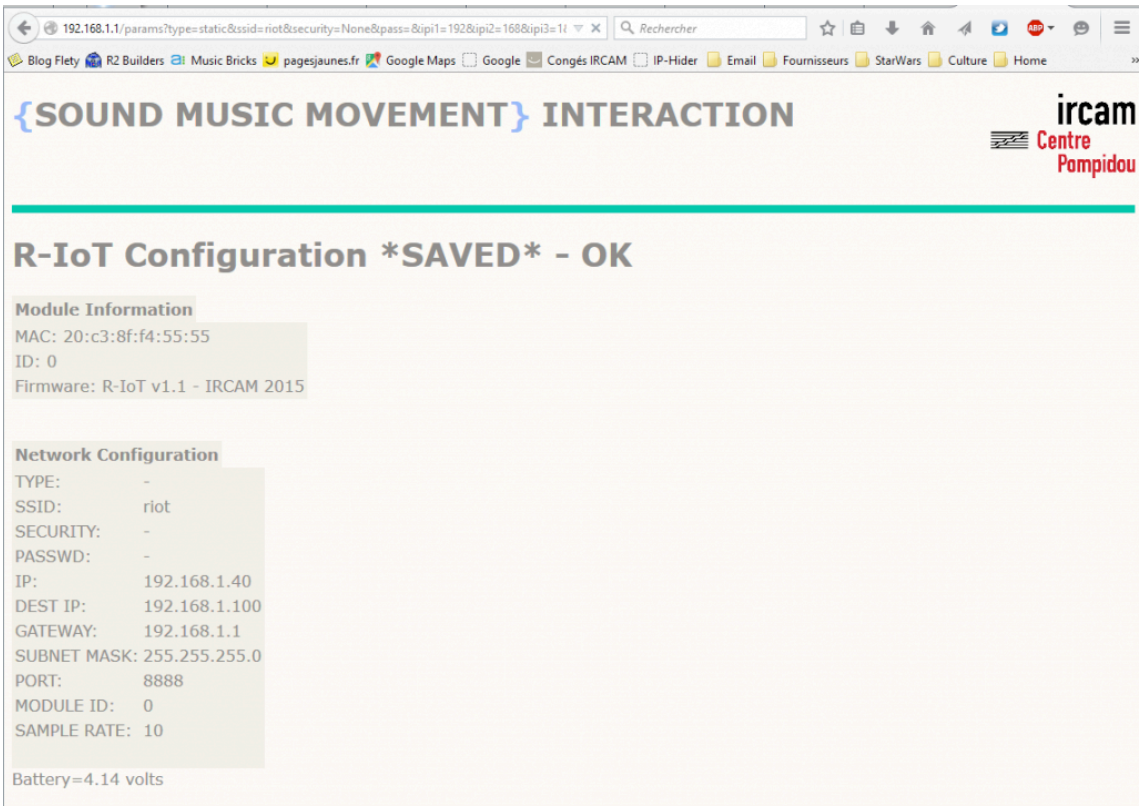


Currently, the IP address type remains DHCP in the current version of the firmware until Energia stabilizes its API, and security isn't implemented yet neither. The main parameters that can be adjusted are the IP address of the computer to connect to (DEST IP), the UDP Port, the ID of the module and the sample rate (min 3 ms).

The ID is used to compose the Open Sound Control address scheme which looks like /<ID>/<stream> followed by a data list. When using several modules on the same WiFi access point, the best practice is to :

- use one UDP / OSC receiver per module, on a dedicated port. This ensures to have a dedicated thread (in like max-msp) to receive the data flow and avoid incoming packets from different modules to be interlaced in the same data pipe. Beware, using several instances of a port listener / OSC receiver (the udp-receive object in max msp) on the same port actually creates a single listener which can be a bottleneck.
- eventually set a different ID for each module to ease up the patching and routing, even though each OSC receiver sends out its own data flow, so routing doesn't matter much there, it's more a convenient way to identify the data origin.

Once the module configuration is finished, press the submit button, settings will be saved in the non volatile memory of the module that then has to be rebooted to apply the new configuration.


192.168.1.1/params?type=static&ssid=riot&security=None&pass=&ipi1=192&ipi2=168&ipi3=1
Rechercher
Blog Flety
R2 Builders
Music Bricks
pagesjaunes.fr
Google Maps
Google
Congés IRCAM
IP-Hider
Email
Fournisseurs
StarWars
Culture
Home
{SOUND MUSIC MOVEMENT} INTERACTION
ircam
Centre
Pompidou
R-IoT Configuration *SAVED* - OK
Module Information
MAC: 20:c3:8f:f4:55:55
ID: 0
Firmware: R-IoT v1.1 - IRCAM 2015
Network Configuration
TYPE: -
SSID: riot
SECURITY: -
PASSWD: -
IP: 192.168.1.40
DEST IP: 192.168.1.100
GATEWAY: 192.168.1.1
SUBNET MASK: 255.255.255.0
PORT: 8888
MODULE ID: 0
SAMPLE RATE: 10
Battery=4.14 volts