TUTORIAL : CONFIGURING A BLE DEVICE (RN4871) AS A WIRELESS ACTUATOR FOR DRAGONBOARD 410C

[1] RN4871 MODULE CONFIGURATION

STEP	COMMAND / ACTIVITY	DESCRIPTION
1	Connect BLE RN4871 device to a USB port of the PC	**screen** /dev/tty.usbmodem1411 115200
2	$$$ +	Put the RN4871 into command mode
3	PZ	Clear all services
4	R,1	Reboot the RN4871
5	$$$ +	Put the RN4871 into command mode
6	Generate 3 UUIDs of 128 bits through **https://www.uuidgenerator.net/**	UUID0: 59c88760536411e7b114b2f933d5fe66 UUID1: 59c889e0536411e7b114b2f933d5fe66 UUID2: 59c88d6e536411e7b114b2f933d5fe66
7	PS,59c88760536411e7b114b2f933d5fe66	Create private service with given UUID0
8	PC,59c889e0536411e7b114b2f933d5fe66,12,01	Create GATT characteristic with UUID1 Property: Notification / Read Data Bytes: 1 Characteristic : 0072 (automatically defined)
9	PC,59c88d6e536411e7b114b2f933d5fe66,16,01	Create GATT characteristic with UUID2 Property: Notification / Read / Write without response Data Bytes: 1 Characteristic : 0075 (automatically defined)
10	SW,0A,00	Remove special functionality on PIN P12 of the module (pin index 0A)
11	SW,0B,00	Remove special functionality on PIN P13 of the module (pin index 0B)
12	SN,DRAGONWALLY	Set module name to “DRAGONWALLY”
13	R,1	Reboot the module

STEP	COMMAND / ACTIVITY	DESCRIPTION
14	$$$ +	Put the RN4871 into command mode
15	WC	Clear current script
16	R,1	Reboot the module
17	$$$ +	Put the RN4871 into command mode
18	WW	Enter script input mode
19	@PW_ON # power on event |O,18,00 # turn leds on |O,18,18 # turn leds off SHW,0072,57 # write “W” (0x57) in characteristic 0072 SHW,0075,18 # write 0x18 in characteristic 0075 SM,2,0064 # start timer 2 with T=1s @CONN # connection event SM,2,0064 # reset timer 2 with T=1s @DISCON # disconnection event SM,2,0064 # reset timer 2 with T=1s @TMR2 # timer 2 event $VAR1=SHR,0075 # read characteristic 0075 and \|O,18,$VAR1 # write the value to the led outputs SM,2,0064 # reset timer 2 with T=1s	Copy this script from a text editor and paste into the **screen** terminal
20	Type ESC	Exit Script input mode
21	R,1	Reboot the module
22	$$$ +	Put the RN4871 into command mode
23	SR,4040	Set module to run script after power on and with no prompt
24	R,1	Reboot the module

[2] BLUEPY INSTALLATION

BLUEPY is a convenient library for accessing the Bluetooth Low Energy features of the **DragonBoard 410c**.

Reference : https://github.com/IanHarvey/bluepy

To install the current released version, on most Debian-based systems:

```
$ sudo apt-get install python-pip libglib2.0-dev
$ sudo pip install bluepy
```

[3] BLUEPY DOCUMENTATION

Reference : http://ianharvey.github.io/bluepy-doc/

[4] PYTHON SCRIPT (EXAMPLE)

```
#! /usr/bin/python

#####################################
#  bluepy_dw_write.py               #
#####################################
from bluepy.btle import Scanner, DefaultDelegate
from bluepy.btle import Peripheral, UUID
import sys

#################################
# List of Known UUIDs           #
#################################
uuid = [ '59c88760-5364-11e7-b114-b2f933d5fe66',  # UUID0  (service UUID)
         '59c889e0-5364-11e7-b114-b2f933d5fe66',  # UUID1  (characteristic 0072)
         '59c88d6e-5364-11e7-b114-b2f933d5fe66']  # UUID2  (characteristic 0075)

###################################
# BLE Target                      #
###################################
ble_module_name = "DRAGONWALLY"
ble_target_uuid = uuid[2]   # UUID2
ble_target_value = [int(str(sys.argv[1]),16)] if (len(sys.argv)==2) else [0x00]

#############################
# convert Vetor to String   #
#############################
def makestring(bytes):
    return "".join(map(chr,bytes))

#################
# Scan Delegate #
#################
class ScanDelegate(DefaultDelegate):
    def __init__(self):
        DefaultDelegate.__init__(self)

    def handleDiscovery(self, dev, isNewDev, isNewData):
        if isNewDev:
            print "Discovered device", dev.addr
        elif isNewData:
            print "Received new data from", dev.addr
```

```
################
# Scan Devices #
################
print("[SCAN DEVICES]")
scanner = Scanner().withDelegate(ScanDelegate())
devices = scanner.scan(10.0)
mydevice = None

###################
# Analyze Devices #
###################
print("[ANALYZE DEVICES NAMES]")
for dev in devices:
    print "Device %s (%s), RSSI=%d dB" % (dev.addr, dev.addrType, dev.rssi)
    for (adtype, desc, value) in dev.getScanData():
        #############################
        # print Complete Local Name #
        #############################
        if (desc == "Complete Local Name"):
               print "  %s = %s" % (desc, value)
        ##################################
        # find target device by its name #
        ##################################
       if (desc == "Complete Local Name") and (value == ble_module_name) :
               mydevice = dev
               print "[Device %s found as %s]" % (mydevice.addr,value)

###########################
# Target Device was Found #
###########################
if mydevice is not None:
     ##########################
     # Open Peripheral and    #
     # Search Services        #
     ##########################
     p = Peripheral(mydevice.addr,mydevice.addrType)
     print "Get Services ..."
     services = p.getServices()
     print "Done."
     ##########################
     # Analyze Services       #
     ##########################
     for serv in services:
       print "Peripheral Addr=%s , UUID=%s" % (serv.peripheral.addr,serv.uuid)
       characteristics = serv.getCharacteristics()
       for ch in characteristics:
          print "   UUID=%s , Properties=%s , Value=%s" % (UUID(ch.uuid).getCommonName(),ch.propertiesToString(),ch.read() if ch.supportsRead() else
"-")
           ############################################
           # Write Value of the Target Characteristic #
           ############################################
           if (UUID(ch.uuid).getCommonName() == ble_target_uuid):
              ch.write(makestring(ble_target_value))
              print("[Writing Value to Target UUID]")
```

[4.1] PYTHON SCRIPT DESCRIPTION

This example script (“**bluepy_dw_write.py”**) scans all nearby BLE devices, looking for its **addresses** and **complete local names**.

Once the “target name" (**ble_module_name**) is found, the scripts traverses all its services and characteristics, looking for the “target UUID" (**ble_target_uuid**).

If a match occurs, the “target value” (**ble_target_value**) is written to the characteristic.

Any Python script accessing the **BLUEPY** library must run as “**superuser**”:

```
$ chmod +x  bluepy_dw_write.py
$ sudo ./bluepy_dw_write.py  <value>
```

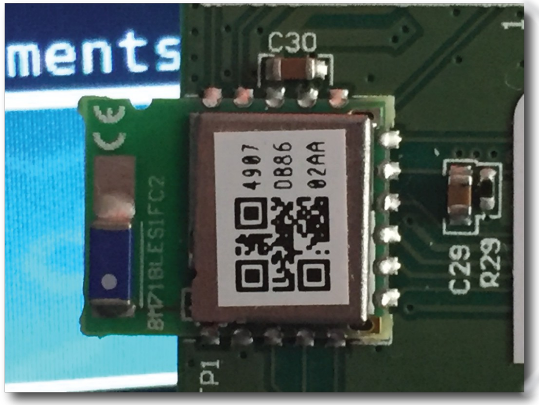
<value>	LED1	LED0
0x00	ON	ON
0x18	OFF	OFF
0x08	ON	OFF
0x10	OFF	ON

Usage example:

```
$ sudo ./bluepy_dw_write.py 0x00        (turn both LEDs ON)
```

[5] SOME PICTURES

The RN4871 BLE Module

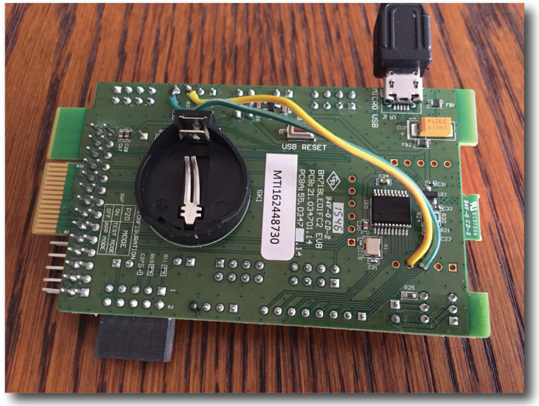


(very tiny : 9 mm x 11.5 mm)

The RN4871 Evaluation Board

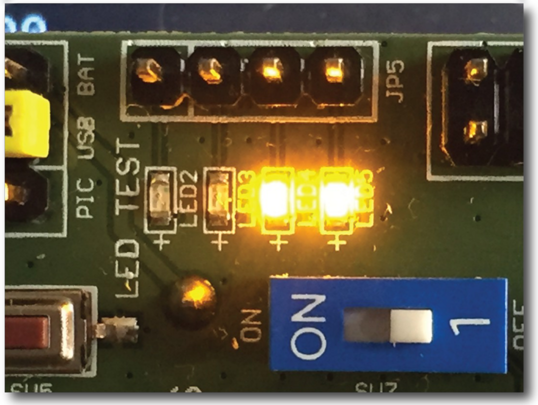
DragonWally Test Setup

Required Wire Connections



RN4871 P12 (GREEN, LED0)
RN4871 P13 (YELLOW, LED1)

LED1 (ON) and LED0 (ON)



The **DragonWally** project uses Bluetooth Low Energy (BLE) for the communication between the **DragonBoard 410c** and wireless actuators and sensors. This demonstration shows the **RN4871** module acting as a two output digital actuator (for access grant or deny)

[6] ADITIONAL REFERENCES

[6.1] DragonBoard 410c

https://developer.qualcomm.com/hardware/dragonboard-410c

[6.2] RN4871 Bluetooth Low Energy PICtail / PICtail Plus Daughter Board

http://www.microchip.com/DevelopmentTools/ProductDetails.aspx?PartNO=RN-4871-PICTAIL

[6.3] RN4871 Bluetooth Low Energy Module

http://www.microchip.com/wwwproducts/en/RN4871

Cezar Menezes
cezar.menezes@live.com