Master Thesis Report

PrithviRaj Narendra

1st August, 2014

Abstract

Hello Abstract. **1.Switch to biblatex.**

Contents

List of Abbreviations **3**

1 Introduction **4**
- 1.1 General introduction to the domain of the thesis 4
- 1.2 Problem definition 4
- 1.3 Goal 4
- 1.4 Outline of the report 4

2 Background **5**
- 2.1 Overview of Bluetooth Low Energy 5
  - 2.1.1 Design objectives of Bluetooth Low Energy (BLE) 5
  - 2.1.2 BLE network architecture 6
- 2.2 BLE stack overview 6
- 2.3 Overview of Contiki 7
- 2.4 Overview of 802.15.4 7
  - 2.4.1 Physical layer 7
  - 2.4.2 Medium Access Control (MAC) layer 7

3 Method **8**

4 Porting a BLE platform to Contiki Operating System (OS) **9**
- 4.1 BLE Hardware Platform 9
  - 4.1.1 Requirements of the hardware platform 9
  - 4.1.2 Comparison and selection of the hardware platform 9
  - 4.1.3 Overview of nrf51822 System-on-Chip (SoC) 10
- 4.2 Porting PCA10000 platform of nrf51822 to Contiki 10
  - 4.2.1 Folder structure of Contiki 10
  - 4.2.2 Peripherals required for Contiki 11
  - 4.2.3 Makefile structure of Contiki 11

5 Test Cases **12**
- 5.1 Performance Metrics Definition 12
- 5.2 Data Dump Test Design 12
- 5.3 Request Response Test Design 13
- 5.4 Data Rate Test 14
  - 5.4.1 Latency 15
  - 5.4.2 Energy Consumption 16
  - 5.4.3 Reliability 16

6 Results and Analysis **17**

6.1 Data Dump Test 17
6.2 Request Response Test 17
6.3 What we learnt 17

7 Other Contributions **18**
7.1 Firmware for demo application 18
7.2 Radio driver for BLE advertisement packet logger 18

List of Abbreviations

AFH Adaptive Frequency Hopping.

ATT Attribute Protocol.

BLE Bluetooth Low Energy.

CRC Cyclic Redundancy Check.

CSMA Carrier Sense Multiple Access.

GAP Generic Access Profile.

GATT Generic Attribute Profile.

GFSK Gaussian Frequency Shift Keying.

IC Integerated Circuit.

IoT Internet of Things.

ISM Industrial, Scientific and Medical.

kbps kilobit per second.

L2CAP Logical Link Control and Adaptation Protocol.

LL Link Layer.

MAC Medium Access Control.

MCU microcontroller.

OS Operating System.

PDR Packet Delivery Ratio.

RDC Radio Duty Cycle.

SIG Special Interest Group.

SM Security Manager.

SoC System-on-Chip.

1 Introduction

1.1 General introduction to the domain of the thesis

What is BLE? Why was it developed and who/where is it being used? [1]

What is Contiki OS? Why was it and who/where is it being used? Explain the amount of R&D done in terms of communication protocols based on 802.15.4 [2].

1.2 Problem definition

Need for BLE in Contiki as a platform for Internet of Things (IoT). To understand the characteristics of BLE and 802.15.4 so that we can identify the applications suitable for these protocols.

1.3 Goal

To choose and integrate a BLE hardware platform with Contiki. To compare BLE and 802 in various performance criteria of data-rate, latency, reliability and energy consumption in environments with and without external interference.

1.4 Outline of the report

2.method??

2 | Background*

2.1 Overview of Bluetooth Low Energy

Rest of overview, from what is missed in 1st chapter.

2.1.1 Design objectives of BLE

BLE was designed by Bluetooth Special Interest Group (SIG) from the ground up, which helped it achieve certain design goals. These design goals for this wireless personal area network were *low cost, worldwide operation, short range, robustness* and *low power*[3].

Low Power BLE aims to use a tiny batteries such as button cell to keep a device operating for months to years. To achieve this goal, BLE was optimized to communicate small amounts of data, such as the states of devices. Also BLE is optimized to have lower peak power requirements, which allows use of button cells to be used with BLE devices.

Worldwide Operation For a technology to be adopted, it is important that there is uniform conformity to the regulations around the world. The 2.4 GHz Industrial, Scientific and Medical (ISM) radio band is the only one available license free worldwide. The technology to develop wireless devices in this band is mature making it the suitable radio band for BLE.

Short Range BLE was designed to be for personal area network like Classic Bluetooth, which means that it is not a network to work with a cellular base station network. This design criterion goes hand in hand with *low power*.

Low Cost Lower power requirements mean that the batteries in BLE devices need to smaller and have to be replaced less frequently, both resulting in a reduction of cost for both the manufacturer and the customer. The use of the ISM band for communication levels removes the licensing entry barrier for start-ups to develop BLE devices. BLE embraces simplicity in its pursuit to lower the cost. BLE supports only single-hop communication in a star network, which reduces the memory and processor requirement for supporting the protocol. Simplicity was the key factor for the choosing of Gaussian Frequency Shift Keying (GFSK) as the modulation scheme for BLE to result in low cost, small radio implementation of the radio in the Integerated Circuits (ICs) for BLE.

*Includes text and images from minor thesis 'Business ideation for BLE'

Robustness The 2.4 GHz space is crowded with devices communicating with various standards as well as spurious noise making the robustness a key criteria in developing BLE standard. BLE uses a multi-channel hopping mechanism called Adaptive Frequency Hopping (AFH) to detect, avoid and recover from interference. In addition to AFH, BLE uses Cyclic Redundancy Check (CRC) to detect and recover from bit-errors due to background noise.

2.1.2 BLE network architecture

Single and Dual devices, Advertisers and scanners, Master and Slave, Star Network, figure 2.1

Figure 2.1: Typical BLE network

2.2 BLE stack overview

Explain host and controller division. More details about Link Layer (LL).

Image of the stack

Physical layer

Link Layer (LL)

Logical Link Control and Adaptation Protocol (L2CAP)

Generic Access Profile (GAP)

Attribute Protocol (ATT)

Generic Attribute Profile (GATT)

Security Manager (SM)

2.3 Overview of Contiki

??????????????? Which aspects of Contiki here?

2.4 Overview of 802.15.4

Mention that in this thesis the Contiki specific implementations of the 802.15.4 layers will be tested.

2.4.1 Physical layer

2.4.2 MAC layer

2.4.2.1 Radio Duty Cycle (RDC) layer

2.4.2.2 Carrier Sense Multiple Access (CSMA)

3.Seperate section Previous work done on comparing BLE and 802.15.4

3 Method

- The first task of porting a BLE hardware platform to Contiki is qualitative?????
  - This task is a per-requisite for the next task
- The task of comparing two communications protocols running on Contiki OS uses quantitative methodology.
  - Deductive approach
    - Hypothesis ?????????????????????
  - Define the metrics to measure
  - Explain the design of the tests to be carried out
  - Carry out the tests and collect the specified data
  - Descriptive statistics with graphical representation for summarizing the data acquired.
  - Univariate analysis of the different metrics defined to compare the two protocols
  - Drawing inferences from this analysis.

4 | Porting a BLE platform to Contiki OS

4. 5to10 pages

4.1 BLE Hardware Platform

What is meant by hardware platform?

4.1.1 Requirements of the hardware platform

The *mandatory* requirements for the platform would be:

- Must have a well supported and documented processor with good specifications.
- For a open source project such as Contiki, a free (and preferably open source) development toolchain must support the platform.
- Availability of well documented datasheet and user manual.
- Availability of an evaluation/development kit.
- Enough memory to accommodate Contiki and BLE stack's requirements.
- Presence of basic peripherals such as timers and serial port required for Contiki.

The non-mandatory, although *nice to have* requirements for the platform would be:

- Presence of flexible power modes with low active and sleep power consumption.
- Availability of a good set of peripherals.
- Availability of BLE stack from the vendor, preferably with source code.

4.1.2 Comparison and selection of the hardware platform

With these requirements, based on the comparison **5.(Appendix A)** of the available (BLE + microcontroller (MCU)) solutions available today, a platform based on nRF51822 from Nordic-Semiconductors would be a suitable option. DA14580 and BCM20732 would also require mention here since from a technical point of view they meet the requirements, but since they are released recently they are not supported well in terms of availability of the IC, evaluation kit, tool-chain and BLE stack.

4.1.3 Overview of nrf51822 SoC

4.2 Porting PCA10000 platform of nrf51822 to Contiki

Contiki was initially developed in a Linux based operating system, although Windows is also supported now. The porting of PCA10000 platform to Contiki was done in Ubuntu 13.10. The compiler used is **6. ... explain all the other details of the development setup**

4.2.1 Folder structure of Contiki

When porting a new platform to Contiki, the three folders inside a Contiki distribution where additions need to be made are cpu, examples and platform. The content in these folders are as described below. The path of the Contiki distribution is in the path CONIKI.

CONTIKI/cpu In this folder, all the files contain implementation which is solely dependent on the SoC is present. This includes the code for the processor abstraction, the drivers of the peripherals of the SoC, makefile with commands for compiling and linking the code, linker file and the documentation of these implementation. The boot-loader or start-up code, if required is also present here. The peripheral drivers for peripherals such as timers and serial port must use the API format of Contiki so that the modules of Contiki using these peripherals can operate correctly. The exact implementations of these drivers is explained in section 4.2.2.

CONTIKI/platform An SoC over time has increasing number of boards or systems that are based on it and these are referred as platforms in this folder. Every board has its own folder in this platform folder which contains files which contain implementation which is dependant on the particular board. These include the specification of the connections to the LEDs, buttons, sensors and serial ports power system, the clock sources, memories present and any other board specific details. Any peripheral driver implementation specific to the board is present here. The default project specifications such as the serial baud-rate and source and frequency of clock are defined here. The main function where the execution of the program starts and initialization of all the peripherals and modules used by Contiki happens is present here.

CONTIKI/examples The examples folder contains all the files related to projects that are implemented using Contiki with any of the supported hardware platforms. The makefile where the make command is executed is present here. The compiled object and binary files are also stored here.

4.2.2 Peripherals required for Contiki

4.2.2.1 Contiki Clock

4.2.2.2 R-Timer

4.2.2.3 LED(s)

4.2.2.4 Radio

4.2.2.5 Button(s)

4.2.2.6 Serial Port

4.2.3 Makefile structure of Contiki

Introduction to makefile and include. There are multiple makefiles present across different folders that are included in each other to form the complete makefile. In the most basic form these are the makefile in the example folder, 'makefile.include' in the Contiki root folder, 'makefile.TARGET' in the specific platform folder and 'makefile.CPU' in the specific cpu folder, where TARGET and CPU are specific to the project. This structure by which this inclusion happens is illustrated better in **figure**.

Makefile where the project is present i.e. in the example folder -Makefile.include in root CONTIKI directory –Makefile indicating the target (if not already specified) –Makefile of applications, if any are required –Makefile of the target, present in the platform directory —Makefile of the SoC, present in the cpu directory

The makefile in the example folder is where the make command is called. The operations that can be performed with this make command depends on the makefile. Usually the operations are cleaning (removing the object and binary files), compiling the source files into object files , linking these object files to create an executable file, creating a binary file from this executable file, uploading the binary to the SoC to start execution and so on.

The makefile in the project specific example folder, the different project source files are specified and the 'makefile.include' present in the root CONTIKI folder is added. 'makefile.inlcude' glues together all the required components of Contiki and provides the default implementation of compiling and linking. The target makefile in the specific platform directory is included here. In 'makefile.TARGET' the source files present in this directory are added, any Contiki modules required are added and the SoC makefile in the specific directory is added. The SoC

5 | Test Cases

5.1 Performance Metrics Definition

Data Rate

Latency

Reliability

Energy Consumption

5.2 Data Dump Test Design

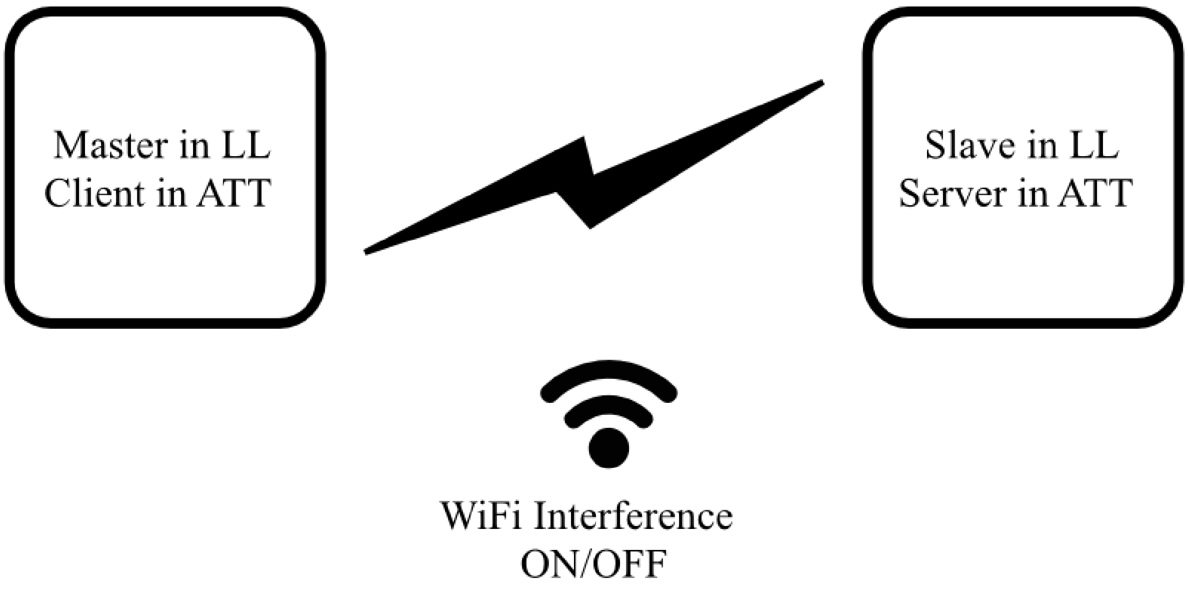

Figure 5.1: Setup to measure data rate with BLE

- Send packets of maximum size for one minute in the fastest way possible
  - Do one test without CCA for 802.15.4
- Repeat with WiFi
  - Choose a channel interfered by WiFi for 802.15.4 and one channel without WiFi interference.

  - For BLE do this with channel map with all channels and channel map with WiFi free channels.

- Do this test 10 times

In this test, to measure the various performance criteria in an environment with interference, WiFi interference will be introduced using the tool 'iperf'. The interference pattern generated will simulate streaming of large data over WiFi. **7.Explain the exact configuration of UDP packets sent**

Metrics evaluated

- Data Rate
- Reliability
- Energy Consumption

5.3 Request Response Test Design

- Explain what is meant by request response
- Different cases in BLE and 802.15.4

Metrics evaluated

- Latency
- Energy Consumption

The objectives of the tests are to compare the link layer of BLE and 802.15.4 in the aspects mentioned in the following sections.

The LL of 802.15.4 consists of MAC layer on top of a RDC layer. For the RDC layer the Null-RDC and ContikiMAC driver will be tested in each of the test with CSMA as the MAC layer. In case of Contiki-MAC the receiving node switches on periodically to sense if there are any packets that need to be received. The default time of this period is 125 ms. In case of null-RDC the radio receiver is never switched off, as the name suggests.

For BLE, communication will be done at the Generic Attribute (GATT) layer because the binary from Nordic Semiconductor used in this project does not provide APIs to access the lower layers in both peripheral and central devices. In all the tests the packet structure in 802.15.4 will be same as in BLE above the link layer. In this test, to measure the various performance criteria in an environment with interference, WiFi interference will be introduced using the tool 'iperf'. The interference pattern generated will simulate streaming of large data over WiFi. **8.Explain the exact configuration of UDP packets sent** In BLE, the devices can assume different roles in the different layers of the protocol. In the LL a device can be a 'Master' or a 'Slave'. In the ATT layer, a device can be a 'Client' and/or 'Server'. A server contains data and the client can request data from the server.

In some of the tests, to measure the various performance criteria in an environment with interference, WiFi interference will be introduced using the tool 'iperf'. The interference pattern generated will simulate streaming of large data over WiFi.

5.4 Data Rate Test

This experiment aims to measure the maximum data rate of BLE and 802.15.4 at the LL with and without WiFi interference. For the BLE test the devices are configured as shown in the diagram below.

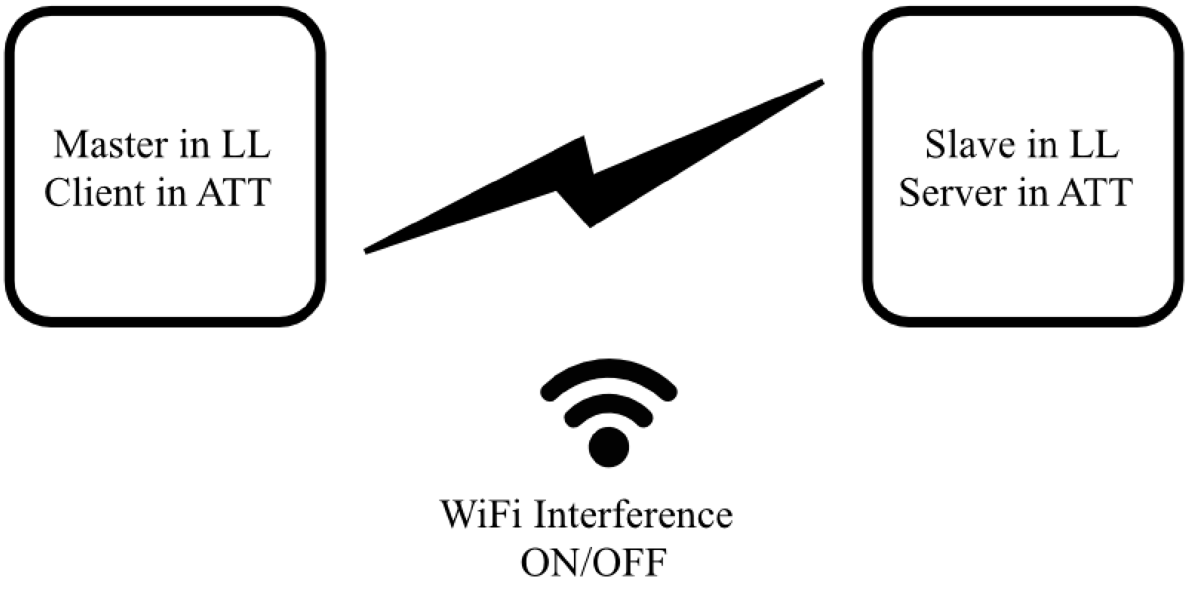

Figure 5.2: Setup to measure data rate with BLE

In this test the data rate is measured by sending data from the server to the client which is not acknowledged in the ATT layer. The data rate is measured at the receiver i.e the Master device. The packet structure will be used such that the complete packet size of BLE is used. Each run of the test will last of one minute to collect enough data to measure the average data rate.

In case of 802.15.4 one node is configured as transmitter and the other as receiver. The data-rate will be measured at the receiver. In these tests two channels will be used, one which interferes with WiFi and one which does not. This means that each channel will be tested with and without interference. The complete packet size of 802.15.4 will be used when conducting this test.

For data rate measurement tests in both the standards with and without WiFi, the data rate will be measured by sending data for one minute. The data rate will be expressed in both packets per second and kilobit per second (kbps).

5.4.1 Latency

This experiment aims to measure the latency for a read request in case of BLE and 802.15.4, i.e. the time difference between sending the read request packet and receiving the packet with the value requested. In case of both the protocols there are intervals at which the radio is active, where communication can take place. To make sure that the read request packets are sent randomly at any point in this interval, timers with random values would be used to initiate sending these packets. The table below shows the connection parameters used for the various tests to measure the latency with BLE. The latency is measured at the device sending the packet '1' in the 'Packet' column and is defined as the time from sending the packet '1' to receiving the packet '2'.

9.Redo the correct table

Connection Interval	Connection Configuration	Devices' Configuration		
		LL	GATT	Packet
7.5 ms	Slave latency = 0	Master	Client	1. Read request
	Supervision timeout = 1 s	Slave	Server	2. Read response
	Slave latency = 120	Master	Client	1. Read request
	Supervision timeout = 1 s	Slave	Server	2. Read response
	Slave latency = 120	Slave	Server	1. Indication
	Supervision timeout = 1 s	Master	Client	2. Indication ACK
125 ms	Slave latency = 0	Master	Client	1. Read request
	Supervision timeout = 1 s	Slave	Server	2. Read response
	Slave latency = 15	Master	Client	1. Read request
	Supervision timeout = 3 s	Slave	Server	2. Read response
	Slave latency = 15	Slave	Server	1. Indication
	Supervision timeout = 3 s	Master	Client	2. Indication ACK

Table 5.1: BLE latency measurement test cases

Where,

Connection Interval: After a BLE connection is established, the master device must always send a packet to the slave periodically after the time specified in connection interval. These connection events provide an opportunity for the slave to communicate with the master.

Slave Latency: The maximum number of connection events that a slave can choose not to respond to the master. This is intended to save power on the slave while also providing an opportunity to communicate with the master if necessary.

Supervision Timeout: A BLE connection is said to be lost if there is no bi-directional communication between the master and the slave for the duration specified by supervision timeout. After a tiemout the master will not send a packet to the slave in every connection event.

Indication: A packet containing an (attribute) value sent from a GATT server to a GATT client, which the client should acknowledge (ACK). This packet is sent without the client requesting this data.

In the case of 802.15.4, one node is configured to send unicast messages and another is configured to receive these and send a response. The latency using both ContikiMAC and NullRDC will be measured. The latency is measured as the time difference between sending a unicast message and receiving its response.

The BLE test cases are designed such that the 7.5 ms tests can compare with NullRDC test of 802.15.4 while the 125 ms tests can compare with the ContikiMAC test of 802.15.4. There are tests with Slave Latency as zero and non-zero, which will help in identifying its effect on latency and the energy consumption of both the master and slave device.

To get an overall sense of the latency over a period of time, the average and standard deviation of the delay for 1000 transactions will be calculated for each test. In all the tests the payload will be of 20 bytes.

5.4.2 Energy Consumption

Energy consumption will be indirectly measured by logging the radio activity in all the tests described above in both the devices communicating. This will provide the duty cycle of when the radio is on. The possibility of logging the sleep cycle of the processor using the binary for BLE software needs to be verified. If the state of the processor is available, it will also be logged.

5.4.3 Reliability

Similar to measuring the energy consumption, reliability will be evaluated by correlating the logs of when the radio is switched on versus when the packets are logged. This will provide an idea of how many times a packet has been dropped and hence find out the Packet Delivery Ratio (PDR).

6 Results and Analysis

6.1 Data Dump Test

Add graph

6.2 Request Response Test

Add graph

6.3 What we learnt

7 | Other Contributions

7.1 Firmware for demo application

7.2 Radio driver for BLE advertisement packet logger

Bibliography

[1] C. Gomez, J. Oller, and J. Paradells, “Overview and evaluation of bluetooth low energy: an emerging low-power wireless technology.,” *Sensors (Basel, Switzerland)*, vol. 12, pp. 11734–53, Jan. 2012.

[2] Contiki, “Contiki: The Open Source Operating System for the Internet of Things.”

[3] R. Heydon, *Bluetooth Low Energy: The Developer’s Handbook*. Prentice Hall, 1 ed., 2012.