

Wireless Networking

Course code: CS4222/5422, Assignment #4/Project

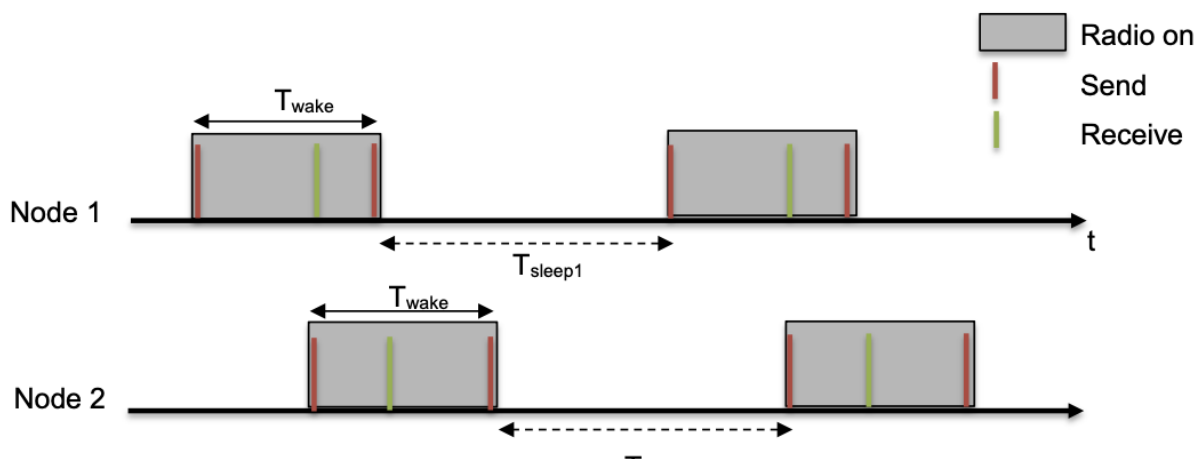
Objective

You will apply various concepts learned in the course and assignments, including sensor interfacing, proximity detection using received signal strength, delay-tolerant networking, and neighbor discovery. The project is structured into two parts:

- You will implement a neighbor discovery mechanism based on "birthday protocol."
- You will enhance this neighbor discovery mechanism to establish a delay-tolerant sensing application. In particular, you will investigate estimating link metrics.

Introduction

You are provided a program that implements "birthday protocol" In this protocol, each node wakes up at random intervals to transmit data and listen for transmissions from nearby devices. The following illustration demonstrates the steps involved in protocol's operation:



The identical code is executed on all nodes. Each node wakes up for a duration of **Twake**, sending two packets: one at the beginning of the wake-up period and another at the end of the wake-up period, just before returning to sleep. Following the initial transmissions, the node keeps its radio active (on state) to listen for potential incoming messages.

In the illustration above, Node 1 and Node 2 operate based on their timers. The intervals between two consecutive wake-up slots (**Tsleep1** and **Tsleep2**) are randomly selected from a uniform distribution, sharing the same mean **Tsleep** value. Adjusting **Twake** and **Tsleep** influences the latency experienced by nodes when discovering neighboring devices. The radio's duty cycle is calculated using the formula: **Twake / (Twake + Tsleep)**

You are provided with a C program, "nbr.c" which implements the fundamental logic outlined above. The modifiable parameters in the program are as follows:

WAKE_TIME (Twake): The default value is set to (RTIMER_SECOND / 10) or 100ms.

The maximum duration of a single sleep interval is constrained by the RTIMER count wraparound. Therefore, the total sleep interval between two wake-up periods is calculated as the product of the duration of a single sleep (SLEEP_SLOT) and the number of sleep cycles (SLEEP_CYCLE).

- **SLEEP_SLOT:** The default value is the same as WAKE_TIME.
- **SLEEP_CYCLE:** This represents the average number of sleep cycles, with a default value of 9. The duty cycle is calculated using the formula: $WAKE_TIME / (WAKE_TIME + SLEEP_CYCLE * SLEEP_SLOT)$.

Task 1:

Please try out various settings and observe your results while performing the tasks:

a) Using the default settings, observe and record how long the devices take to discover each other. Pick one of the devices as **device A** and plot the cumulative distribution of the intervals between packet receptions on **device A** hearing from **device B**.

b) Reset device B and observe how long it takes for **device A** to hear from the **device B** after **device B** reboots. You may need to modify the given code to observe this duration. Perform the experiments at least 10 times and plot the cumulative distribution.

Next, please modify the program (nbr.c) so that two-way discovery (A hears from B AND B hears from A) can be completed in a deterministic manner within 10 seconds. You should choose settings so that the radio power consumption is "**minimized**".

You must include the following in the report (detailed instructions below in the document):

- a) The algorithm you have implemented
- b) The parameters chosen
- c) The maximum two-way latency observed

Important: Please note that the two-way latency is the time it takes for node A to hear from node B, and then for node B to hear from A. In other words, it is the time it takes for performing two-way neighbor discovery.

Setting up Contiki

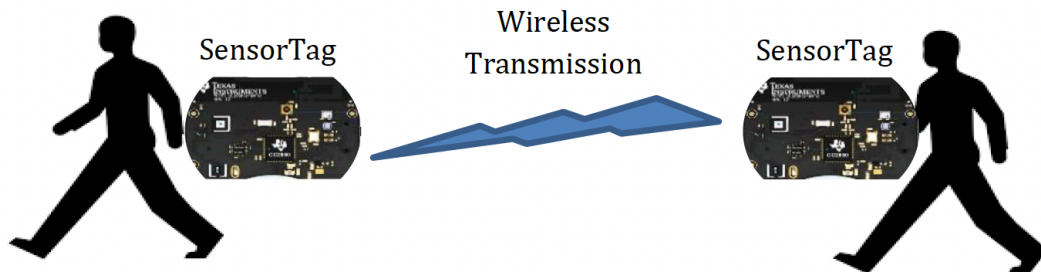
- Extract provided [neighbour.zip](#) in the contiki-ng/examples directory
- Compile the program using command "make TARGET=cc26x0-cc13x0 BOARD=sensortag/cc2650" in the directory "contiki-ng/examples/nbr".
- Use uniflash program to burn the binary file to the SensorTag.
- Observe the output of the program through the USB serial port.

You need to run the program on 2 devices to perform the experiment.

Task 2:

There has been significant interest in offloading/backhauling networks. These are useful for offloading sensor data from scenarios where traditional network connectivity, such as cellular towers, may not be available. In these networks, nearby devices, like smartphones, laptops, and tablets, pick up transmissions from sensors and offload them to a cloud-based server whenever they are connected to a traditional network. Examples of

such networks are Apple AirTags and the Find My network. In this case, nearby Apple devices act as relays to track the AirTag's location using Bluetooth and ultra-wideband technology. This allows AirTag to be located even when out of range of its paired device.



In this project, you will design a basic implementation of a backhauling network. One node will sense light readings and collect them in an array. You can sense the light readings at a sampling rate of 1 Hz for a period of 60 seconds. Let us call this as a Node A. Please refer to the second assignment for details regarding the sampling of the light sensor. Next, Whenever another node, which we may call Node B discovers this Node A, and it detects that the link quality between the nodes is good, it transfers the light readings to the Node B. You can decide if the Node A should delete the stored light readings, or if it can continue to maintain. Nonetheless, the transfer should be of the previously unsent light readings.

There are different ways to determine that the link quality is good; it could be that the RSSI is sufficiently high for received packets, or the Packet Received Ratio or ETX is high. You can use an appropriate metric, but please justify your choice in the report. Furthermore, please also justify any threshold used. If you use signal strength, what RSSI value would you consider to indicate a good link with sufficiently strong signals? Explain these choices in the report, and we will account for your answers when grading you for particular task.

Your code should output (write to stdout using printf) the time a device first detects another device in the following format:

```
Timestamp (in seconds) DETECT nodeID
```

DETECT is a keyword for the detection of a NEW node.
A single whitespace separates the fields.

```
For example: 123 DETECT 34567
```

This means that at 123 seconds, a node with ID 34567 is detected.

When the node determines that the link quality is good and it is ready to perform transmission of collected sensor data, please print the information in the following format:

```
345 TRANSFER 34567
```

This means that at 345 seconds, the node with ID 34567 is in the vicinity with good link quality and is ready for the transfer of sensor data.

Notes:

- TRANSFER is only printed when the node has determined that the link quality between nodes is good.
- Please keep the energy consumption for performing the neighbor discovery and other tasks as low as possible.

Finally, after the node has been successfully detected, also print the light sensor readings that were transferred. These should be printed at the NODE B. Please print them as follows:

"Light: Reading 1, Reading 2, ..., Reading 10"

Bonus marks: We will consider you for bonus marks, if you initiate the light sampling process when the node has detected significant motion. Thus, when there is significant motion, the node A starts the light sensing processing and it would capture for 60 seconds. The next light sensing capture would occur again when node detects significant motion.

Submission Guidelines and Deadline

The deadline for submission of the assignment is **April 26th, 2024**. This assignment needs to be done with members of the designated group. We provide implementation of neighbour discovery program here: [neighbour.zip](#)

1. Report as a Single PDF File (report_group_#.pdf)

a. Task 1:

- i. Description of the algorithm you have implemented.
- ii. The parameters chosen.
- iii. The maximum two-way latency observed.

b. Task 2:

- i. The neighbour discovery protocol you have implemented to "duty-cycle" your radio in order to reduce power consumption.
- ii. The logic that you have implemented for link quality detection.
- iii. Results showing the evaluation of your system.
- iv. The logic of the code used to sense motion, capture light readings, and transfer these captured light readings to another sensor tag.

2. Your Code in a Single Directory Named Source-Code

- a. Source code for Task 2 (task_2_group_#.c). You can either have one src file for both transmitter and receiver, or you can have two separate files (one for RX and one for TX)
- b. A README file to compile and execute your programme(s).

3. A video that demonstrates the working of the project.

4. A statement of work indicating the contributions of each member of the group.

You can submit after due date of **26th April 2024** without penalty till **30th April 2024**. Afterwards, we will not accept the submission.