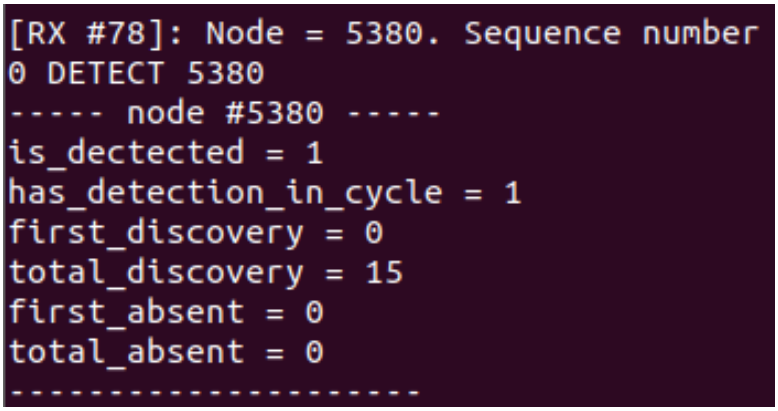
Experimental Settings and Test Cases

With our parameters properly defined, we went on to test the program for its reliability and robustness. To do this, we defined 6 test cases to cover normal scenarios as well as edge cases as described below:

Context for the characteristics of each nodes and their debug statements

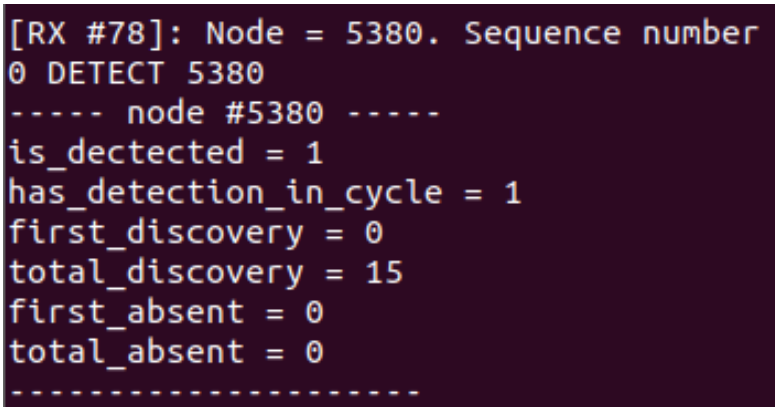
Example Output:

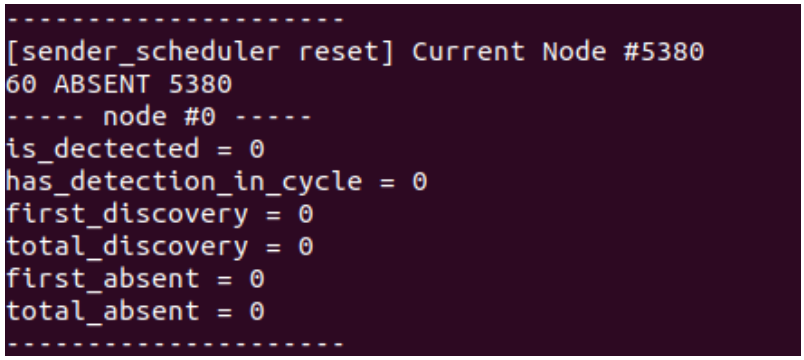
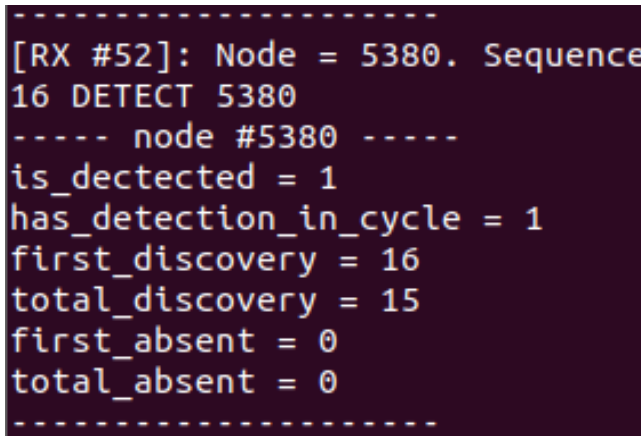


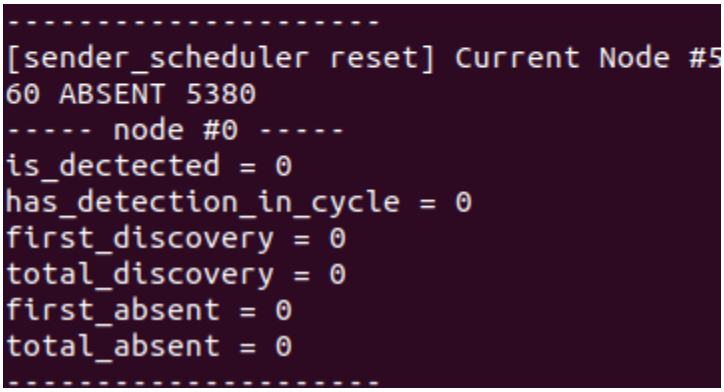
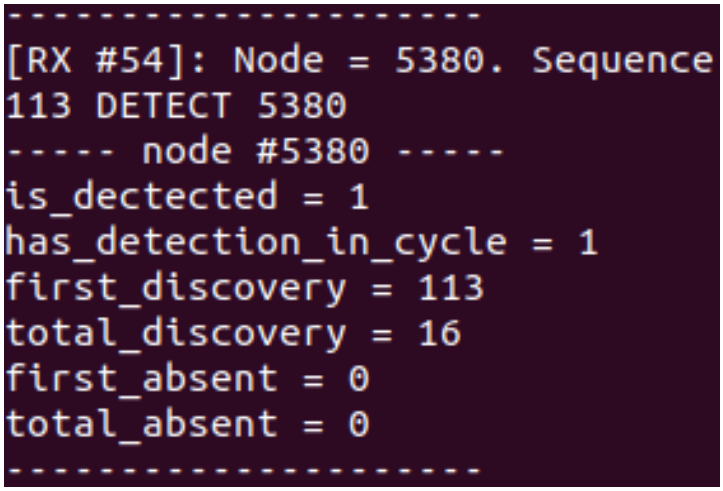
Example 1: Example Debug Output

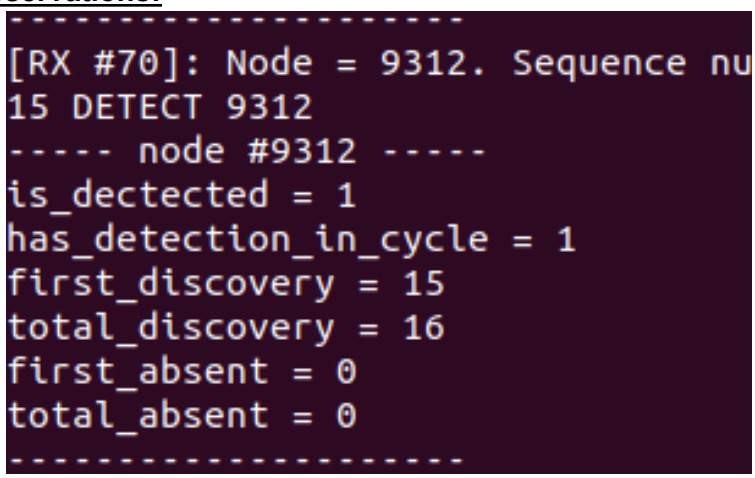
To provide context, from Example 1, the characteristics of each nodes in the above terminal screenshot are as follows:

1. Is_detected: what is the state of the node?
    a. 1 → DETECT
    b. 0 → ABSENT.
2. has_detection_in_cycle: has the node been detected within the 100 time slots?
    a. 1 → Detected
    b. 0 → Not detected.
3. first_discovery: number of seconds at which the two nodes first come into proximity.
4. total_discovery: number of seconds at which the two nodes are in proximity.
5. first_absent: number of seconds at which the two nodes first go out of range.
6. total_absent: number of seconds at which the two nodes are out of range.

Test Case	Test Case Description
Test Case 1	**Scenario:** The two nodes come in range of each other. **Theoretical Observations:** When the time >= 15s, the DETECT statement should be printed in the console. **Actual Observations:**  Figure 5: Test Case 1 Figure 5 shows a screenshot of the terminal when the program was run with the intention of “Test Case 1”. We observed that node #5380 was first detected at t=0, and after 15s, the state transitioned from ABSENT to DETECT. Furthermore, the DETECT statement was printed into the console as well. Hence, the test case worked as expected.
Test Case 2	**Scenario:** The two nodes go out of range of each other. **Theoretical Observations:** When the time >= 30s, the ABSENT statement should be printed in the console. **Actual Observations:**

	 Figure 6: Test Case 2 From Figure 6, we observed that node #5380 was first absent at t=60, and after 30s out of range, the ABSENT statement is printed. Subsequently, the node ID was removed and its characteristics are resetted as shown. Hence, the test case worked as expected.
Test Case 3	**Scenario:** The two nodes come in range of each other for 15s. Subsequently, the two nodes go out of range of each other for 30s. **Theoretical Observations:** When in range, if time >= 15s, the DETECT statement should be printed in the console. Subsequently, when out of range, if time >= 30s, the ABSENT statement should be printed in the console. **Actual Observations:**  Figure 7: Test Case 3 (Detect) From Figure 7, we observe that Node #5380 was first detected at t=16. After 15s of proximity, the device changes state from ABSENT to DETECT, and the DETECT statement is printed into the console.

	 Figure 8: Test Case 3 (Absent) Next, as seen in Figure 8, node #5380 is first seen absent at t=60s, and after 30s, the node is removed from memory and its characteristics are resetted.  Figure 9: Test Case 3 (Detect) Lastly, at t=113s, from Figure 9, node #5380 was first detected and after 16s, the state of the node transitioned from ABSENT to DETECT. Hence, the test case worked as expected.
Edge Case 1	**<u>Scenario:</u>** Irregular detection of nodes. The two nodes come in range of each other for less than 15s and go out of range for at least 2s. The two nodes come into contact again for the full 15s. This is to simulate an event where the two nodes are in close proximity, but is only for a short period for less than 15s. **<u>Theoretical Observations:</u>** As the two nodes did not stay in proximity for more than 15s, the state of the two nodes will not change from ABSENT to DETECT. Hence, no DETECT statement should be printed. After 2s, the two nodes come into range again and remain for 15s, the DETECT statement should be printed out in the console.

	Actual Observations:  Figure 10 Edge Case 1 For this experiment, we walked out of range at around t=13s, and walked back into range at around t=15s. After 16s, the nodes are in DETECT state and the DETECT statement is printed into the console as shown in Figure 10. Hence, the test case worked as expected.
Edge Case 2	**Scenario:** Irregular detection of nodes. The two nodes are in DETECT state. The nodes go out of range for 2s and come back into range. This is to test the limits of the device at a maximum range of 3m. As the RSSI received at the maximum range is spurious, there may be packet loss which results in false detection of ABSENT states. **Theoretical Observations:** The two nodes should recognise that the nodes are out of range for the 2s. When the device comes into range again, the nodes should recognise this and continue to classify the two nodes in DETECT state. No ABSENT statements should be printed. **Actual Observations:**

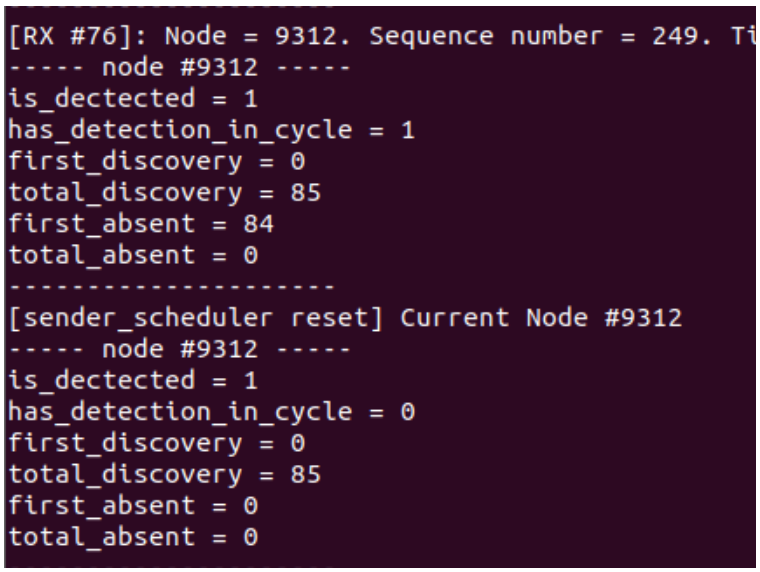
<table>
<tr>
<td></td>
<td>


Figure 11: Edge Case 2

We notice from Figure 11 that there is a false detection of first_absent at t=84s due to packet loss. However, in the next cycle, packets are received as indicated by has_detection_in_cycle = 1. This cause the program to reset the first_absent and total_absent timings and continue classifying the node #9312 in DETECT state.

Hence, the test case worked as expected.
</td>
</tr>
<tr>
<td>Edge Case 3</td>
<td>
<u>Scenario:</u>
Multiple nodes in the same area to simulate a multi-node network.

<u>Theoretical Observations:</u>
When within range, all nodes should discover each other and print the DETECT statement. When out of range, all nodes should print the ABSENT statements too.

<u>Actual Observations:</u>
</td>
</tr>
</table>

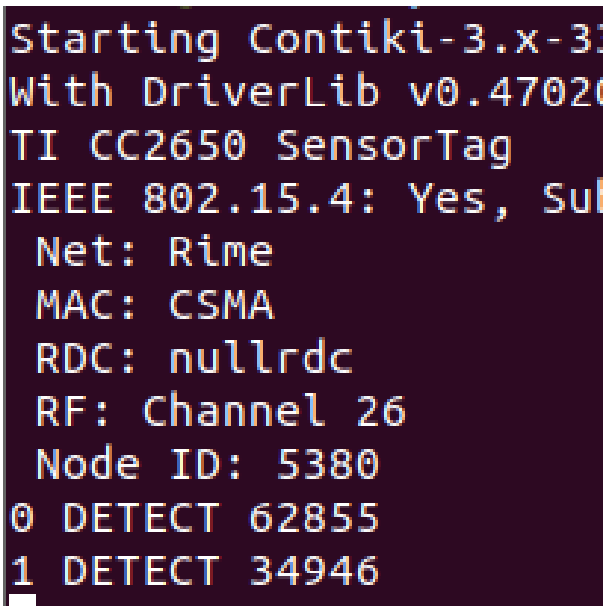


Figure 12: Edge Case 3 (Detect)

From Figure 12, we can see the simultaneous detection of two nodes, #62855 and #34946 when in proximity at t=0s and t=1s respectively.

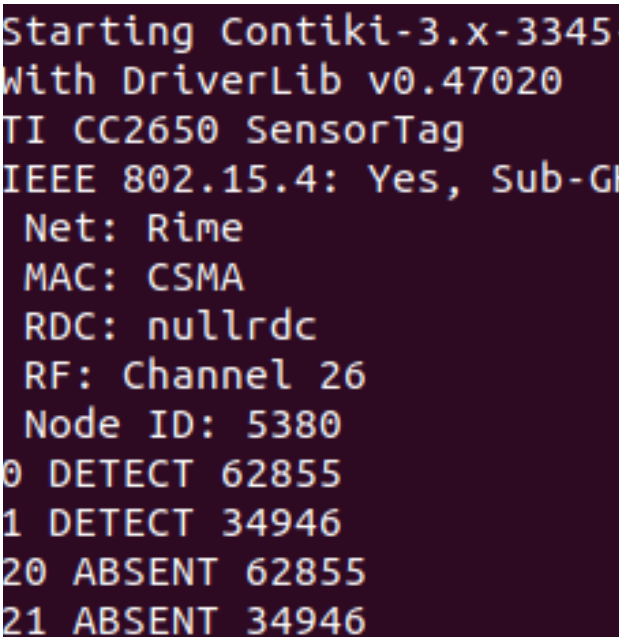
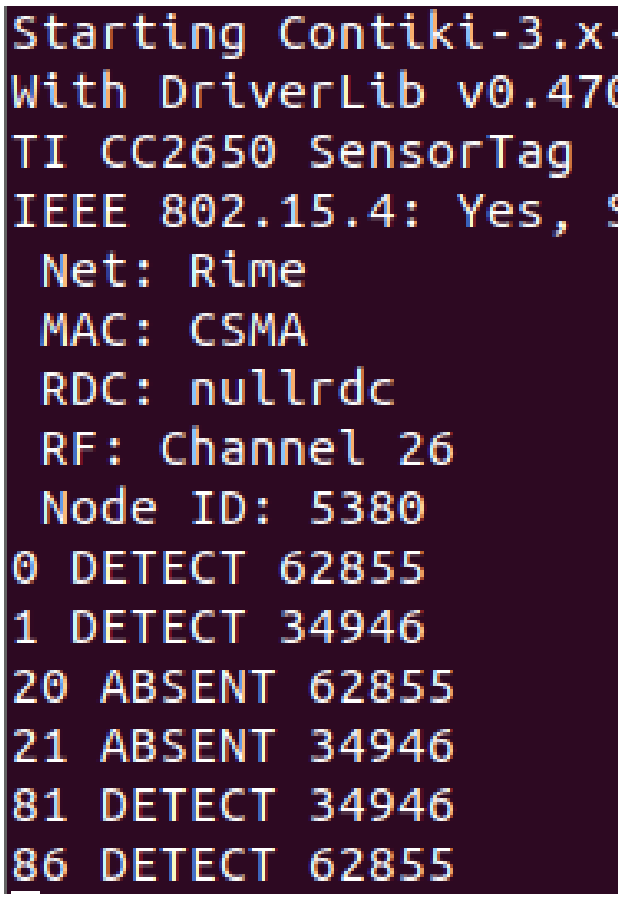


Figure 13: Edge Case 3 (Absent)

From Figure 13, we can see the simultaneous printing of the ABSENT statements of two nodes, #62855 and #34946 when out of proximity at t=20s and t=21s respectively.

	 Figure 14: Edge Case 3 (Detect) From Figure 14, we can see the simultaneous detection of two nodes, #62855 and #34946 when in proximity at t=81s and t=86s respectively. Hence, the test case worked as expected.

Summary

In summary, our system is able to achieve the following:

1) Discover a pair of devices with contact times of 15s or more with high probability.
2) Able to estimate the total duration in which two nodes are in proximity.
3) Discover that a node in proximity has moved away for 30s or more with high probability.

Furthermore, we have also tried to reduce the power consumption by playing with parameters such as duty cycle and transmitting and receiving powers of the SensorTags.