Application Note: Direction Finding Solution

Ingchips Technology Co., Ltd.

Contents

List of Tables

List of Figures

Welcome

Welcome to use *INGCHIPS* 91x Software Development Kit.

INGCHIPS 91x is a BLE 5.1 full feature SoC solution. This application note walks through the offered solution to help anyone get started with Direction Finding.

Chapter 1

Introduction

Bluetooth 5.1 introduced support for Direction Finding by adding the option to send and receive Constant Tone Extensions (CTEs) after Bluetooth packets in both advertising and connection modes. This makes it possible to do phase measurements on antenna arrays and ultimately to determine the direction of an incoming signal.

This application note is organized as following:

- Direction Finding Fundamentals talks about the basic theory of Direction Finding, that is how to determine the direction of incoming signals;
- Quick-Start Guide gives a step-by-step introduction on an AoA demo;
- SDK Support of Direction Finding provides detailed information on SDK design & relevant APIs, including connection & connectionless mode, and two proprietary modes.
- There are also Further Information on how to integrate reference AoA algorithm implementation into other systems.

1.1 Abbreviations & Terminology

Table 1.1: Abbreviations

Abbreviation	Notes
AoA	Angle of Arrival
AoD	Angle of Departure
BLE	Bluetooth Low Energy
CTE	Constant Tone Extension
SDK	Software Development Kit

1.2 References

1. Bluetooth SIG[1]

[1] https://www.bluetooth.com/

Chapter 2

Direction Finding Fundamentals

2.1 Positioning

Positioning technologies have many useful applications, one example being GNSS (GPS, Beidou, GLONASS, Galileo, etc.), which is widely used all over the world.

Unfortunately, GNSS systems do not work very well indoors, so there is a real need for better indoor positioning technologies.

There are two conventional methods to calculate the position of an asset: trilateration and triangulation.

Trilateration uses the distance of the asset from multiple fixed-position locators to determine the position by finding the point that satisfies all distance measurements best. The distance can be deduced from Received Signal Strength Indicator (RSSI) measurement or Time of Flight measurement, for example. Unfortunately, RSSI measurement can be very inaccurate and Time of Flight measurement needs highly accurate time measurement.

Triangulation uses the angles under which the fixed-position locators see the asset (or under which the asset sees the fixed-position locators). The position of the asset is then determined by the intersection point of the lines of sight. Take two locators as an example, with the two measured angles, one can construct a unique triangle by the angle-side-angle congruence theorem within a given plane, and the asset is sitting at the third vertex of this triangle.

The technology to determine these angles is called Direction Finding. Direction finding, or radio direction finding, is – in accordance with International Telecommunication Union (ITU) – defined as radio location that uses the reception of radio waves to determine the direction in which a radio station or an object is located. Direction finding has a very long history, for example, in World War II, it played an important role in combating German threats during both the Battle of Britain and the long running Battle of the Atlantic. It is even a sport, and there are amateur radio direction finding events held all over the world today.

Triangulation can give a more accurate position than trilateration with RSSI measurement, and requires less expensive time measurement hardware than Time of Flight measurement.

However, triangulation needs an antenna array and a method with which the direction (angle) of the incoming signal can be determined.

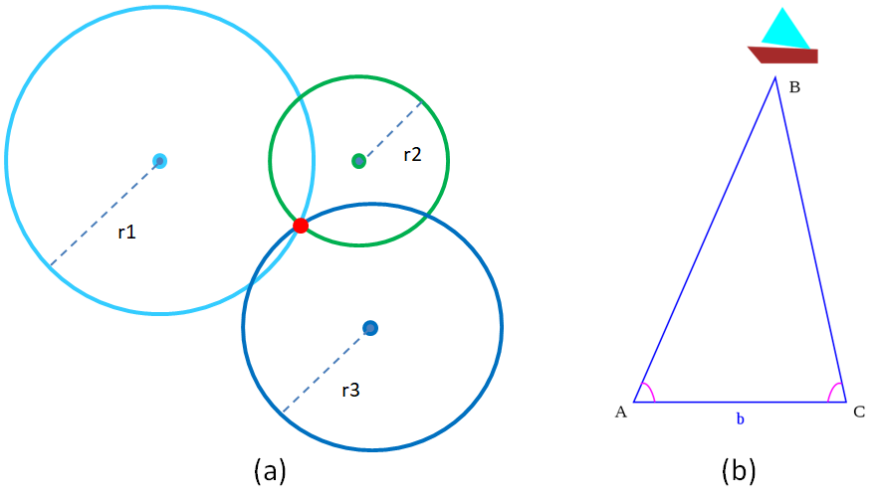

Figure 2.1: Trilateration (a) and Triangulation (b)

2.2 Direction Finding Methods

There are two direction finding methods available in BLE, called Angle of Arrival (AoA) and Angle of Departure (AoD) respectively.

In each method, from the received samples, phase differences can be deduced, and then, direction can be estimated.

2.2.1 Angle of Arrival

In this method, an antenna array is deployed in the receiver. An un-modulated radio wave emitted by a single transmitter will reach different antennas in different phases.

There two methods to measure phases on antennas:

- Sampling all antennas simultaneously

  This requires separated and dedicated RF path for each antenna, which is too expensive for BLE systems.

- Sampling each antenna periodically

  This needs one or more RF switches, but only one RF path, which is much more affordable. When receiving, the receiver will sampling each antenna periodically controlled by a list which is called a *switching pattern*.

2.2.2 Angle of Departure

In this method, the setup is reversed: an antenna array is deployed in the transmitter. An single un-modulated radio wave is generated, and emitted on antennas periodically controlled by a switching pattern by the transmitter.

If multiple transmitter antennas are placed close enough to each other, and the wave is transmitting within a short duration, we can assume that the signals coming from each antenna will reach the receiver's single antenna inheriting the same propagation path. Therefore, phases for each transmitting antenna can be measured and utilized for direction estimation, too. Obviously, the receiver needs to **know all information** about the antenna array used by the transmitter a priori.

2.3 Direction Finding Theory

Antenna arrays and algorithm are both essential for a direction finding system. This section discusses the basics of some typical antenna arrays and estimation algorithms. Here, direction finding refers to the general problem of estimating AoA and AoD.

2.3.1 Antenna Arrays

There are several types of arrays, mounted on a line (Linear Array), on a flat surface (Rectangular Array, Circular Array, L-Array), or on a curved surface (Conformal Array, 3D Array).

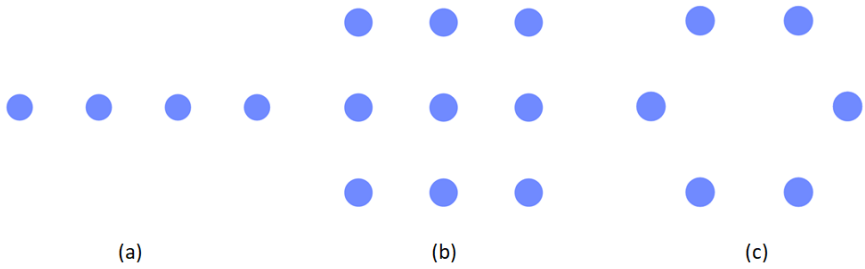

Figure 2.2: Uniform Linear Array (a), Uniform Rectangular Array (b) and Uniform Circular Array (c)

By using a one-dimensional antenna array, one can measure only the azimuth angle, while with two-dimensional arrays, one can measure both azimuth (θ) and elevation (ϕ) angles in the 3D half-space splitted by the plane in which the arrays are embedded. If the array is extended to 3D, then it is possible to measure in the full 3D space.

2.3.2 Algorithm with A Trivial Example

Figure 2.4 shows how a change in the direction of the incoming signal causes a difference in the phase of the received signal on two antennas.

In Figure 2.5 the direction of the incoming signal is θ, and the distance between the two antenna is d.

Then, the time difference of the incoming signal on the two antennas is:

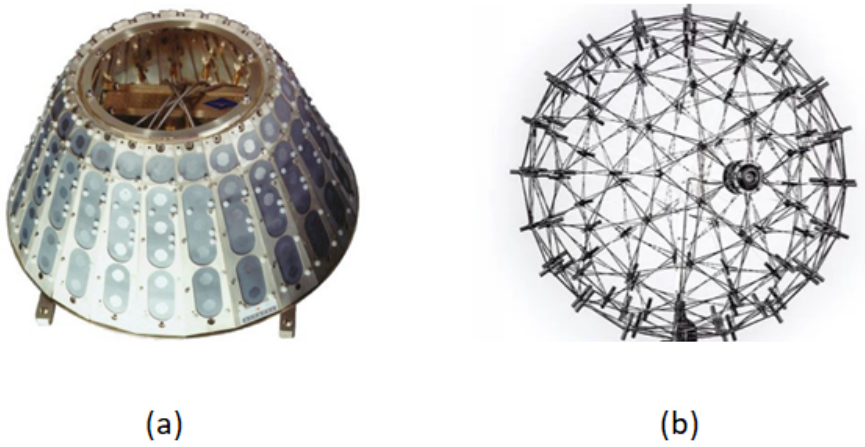

Figure 2.3: Conformal Array on LEO satellites (a) and 3D Microphone Array in a commercial product (b)

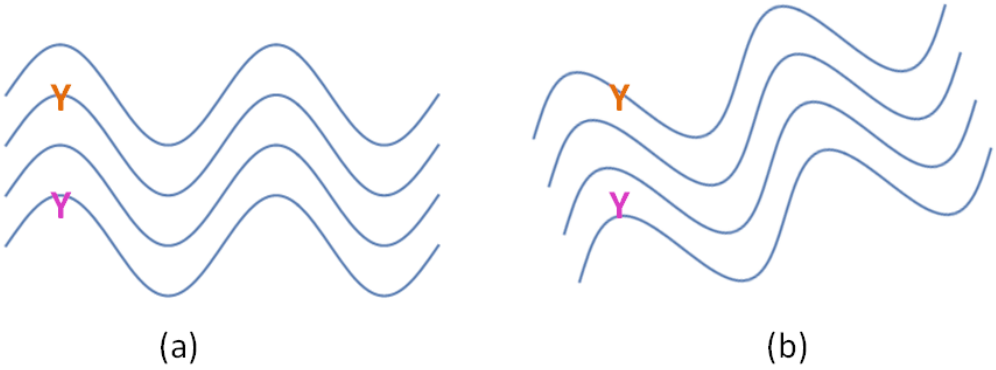

Figure 2.4: Phase Differences Between Antennas

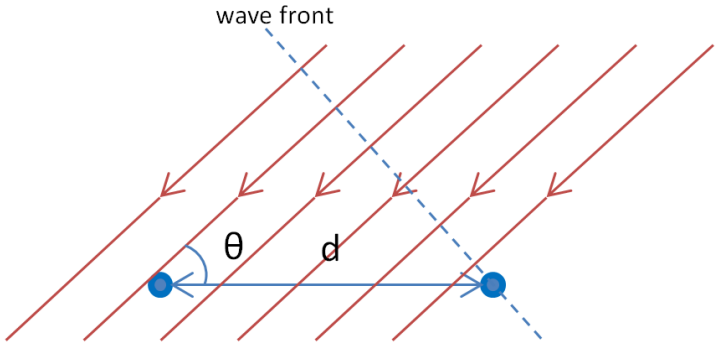

Figure 2.5: Phase Differences Calcuation

$$\Delta t = \frac{d\ cos(\theta)}{c}$$

Where c is the speed of light. The phase difference caused by Δt is:

$$\Delta phase = 2\pi f \Delta t$$

Where f is the frequency of the un-modulated radio signal. From the measurement of phase difference, θ can be deduced:

$$\theta = arccos(\frac{c\ \Delta phase}{2\pi f d}) = arccos(\frac{\lambda\ \Delta phase}{2\pi d})$$

Where λ is the wave length. Note that, $\Delta phase$ is between $-\pi$ and π, which requires that $d < \frac{\lambda}{2}$.

Readers are recommended to read standard textbooks on advanced(array/radar) signal processing for direction estimation algorithms suitable for antenna arrays.

2.4 Location Finding

Without knowing the accurate distance of the asset, and if the asset is moving in full 3D space, one two-dimensional antenna array can provide the direction of the asset, but not its position.

To determine the precise position of the asset in full 3D space, multiple antenna arrays (locators) must be used. By using more than one locator, the position of the asset can then be determined using triangulation. Adding RSSI measurements to the direction estimations may further enhance the position estimation.

2.5 Constant Tone Extension

The un-modulated radio signal is introduced in Bluetooth Core Specification as *Constant Tone Extension* for uncoded PHYs (i.e. 1M & 2M PHYs), starting from Version 5.1.

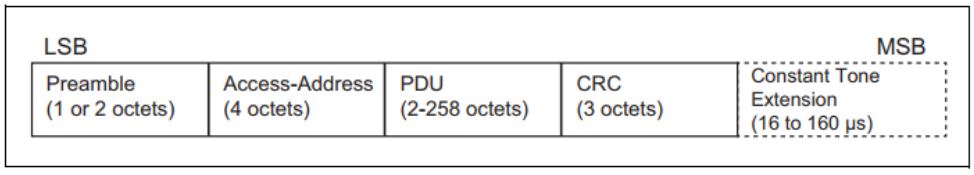

Figure 2.6: Link Layer packet format for the LE Uncoded PHYs

The Constant Tone Extension has a variable length between 16 µs and 160 µs. The contents are a constantly modulated series of 1s, which is an un-modulated wave due to to nature of GFSK, and no whitening is applied to them.

The first 4 µs of the Constant Tone Extension are termed the guard period and the next 8 µs are termed the reference period. After the reference period, the Constant Tone Extension consists of a sequence of alternating switch slots and sample slots, each either 1 µs or 2 µs long as specified by the Host.

The first antenna in the switching pattern is used during the reference period. The second antenna in the pattern shall be used during the first sample slot, the third antenna during the second sample slot, and so on. For example, the switching pattern is $\{5, 6, 8, 7, 10\}$, slot duration is 1 µs, and the antenna selected for each sample is shown in Figure 2.7.

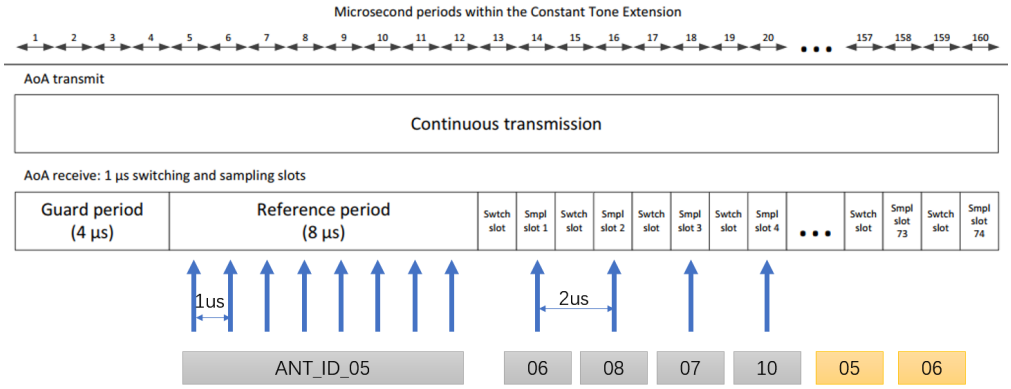

Figure 2.7: Antenna Switching

Chapter 3

Quick-Start Guide

This guide walks through the offered solution to help anyone get started with Direction Finding, especially Direction Finding based on AoA.

3.1 Setup

The whole demo includes 1 or more tags, 1 antenna array board and 1 PC running Windows.

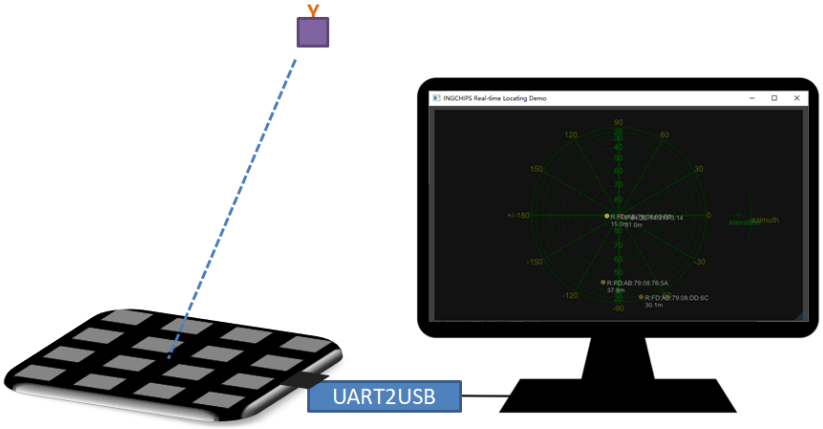

Figure 3.1: Hardware Setup

- Tags - CTE Transmitters

  All ING9188xx based boards/products can be used as tags. Just download *Peripheral LED & CTE* example in SDK.

- 4x4 Antenna Array Board (Rectangle) - CTE Receiver

  App for the 4x4 antenna array board should already have been downloaded, which is exactly the *Central CTE* example in SDK with `PRO_MODE` defined.

The top side of the board (shown in the left of Figure 3.2) should face tags, but not the opposite side.

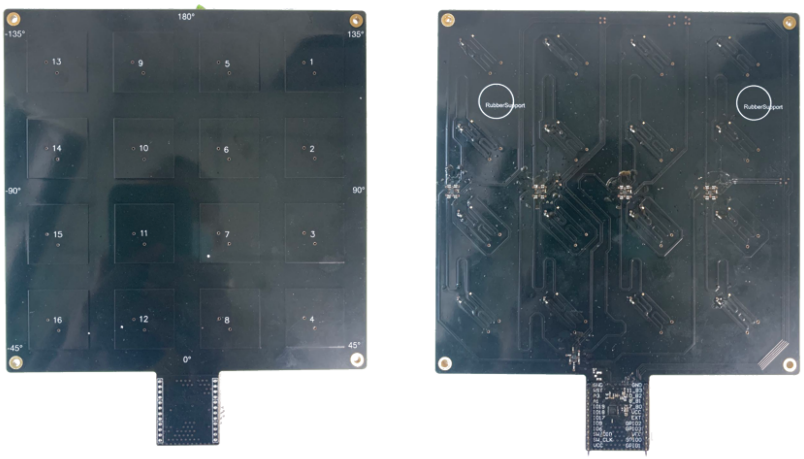

Figure 3.2: 4x4 Antenna Array Board (Rectangle)

Or,

- 3x3 Antenna Array Board (Round) - CTE Receiver

  App for the 3x3 antenna array board should already have been downloaded, which is exactly the *Central CTE* example in SDK with `PRO_MODE` defined, and `CURRENT_ARRAY = ANTENNA_ARRAY_3X3`.

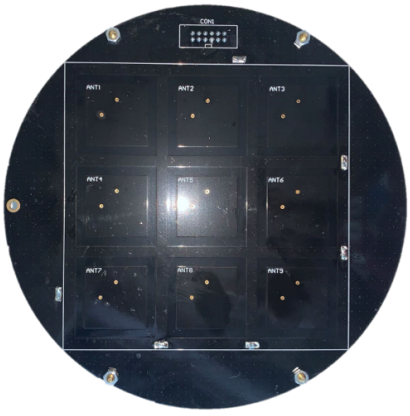

Figure 3.3: 3x3 Antenna Array Board (Round)

- PC

  1 PC running x64 Windows 10 or newer with a mainstream desktop CPU is recommended.

From the main menu of `Wizard`, select “Tools” -> “More” -> “Realtime Locating Demo (DX12)”. If the program could not be started, install Microsoft Visual C++ Redistributable packages[1] for Visual Studio 2022, and try again.

If the PC is running elder Windows version than 10, then select “Tools” -> “More” -> “Real-time Locating Demo (OpenGL)”. If the program could not be started, install Microsoft Visual C++ Redistributable packages for Visual Studio 2022, and try again.

Python[2] 3 is another required piece of software. It’s recommended to use a packge manager for Windows - Anaconda[3] , Chocolatey[4], etc. - to install Python. Use below commands to install additioal required Python libraries:

```
pip install pyserial
```

- UART2USB

  A UART2USB kit is required to connect the antenna array board to PC. Hardware connection (as shown in Figure 3.4):

  – GPIO2 is used for UART transmission
  – Antenna array board is powered through VCC & GND
  – 3.3V is recommended for VCC

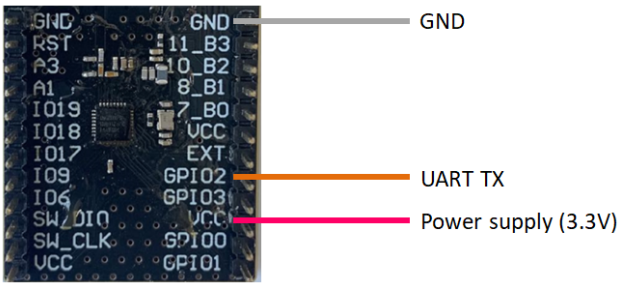

Figure 3.4: Connect 4x4 Antenna Array Board (Rectangle) to UART2USB (PC)

3.2 Run the Demo

After all hardware components are ready, the demo can be started.

From the main menu of `Wizard`, select “Tools” -> “More” to start *Real-time Locating Demo*. There are two variants, one built against DirectX 12, and the other OpenGL. Both share the same functionalities, but have subtle differences on GUI appearance and system dependencies as summarized in Table 3.1. If an array board other than the default 4x4 Antenna Array Board (Rectangle) is used, check out Table 5.3 for the corresponding parameters when launching the tool.

[1] `https://aka.ms/vs/17/release/vc_redist.x64.exe`
[2] `https://www.python.org`
[3] `https://www.anaconda.com`
[4] `https://chocolatey.org`

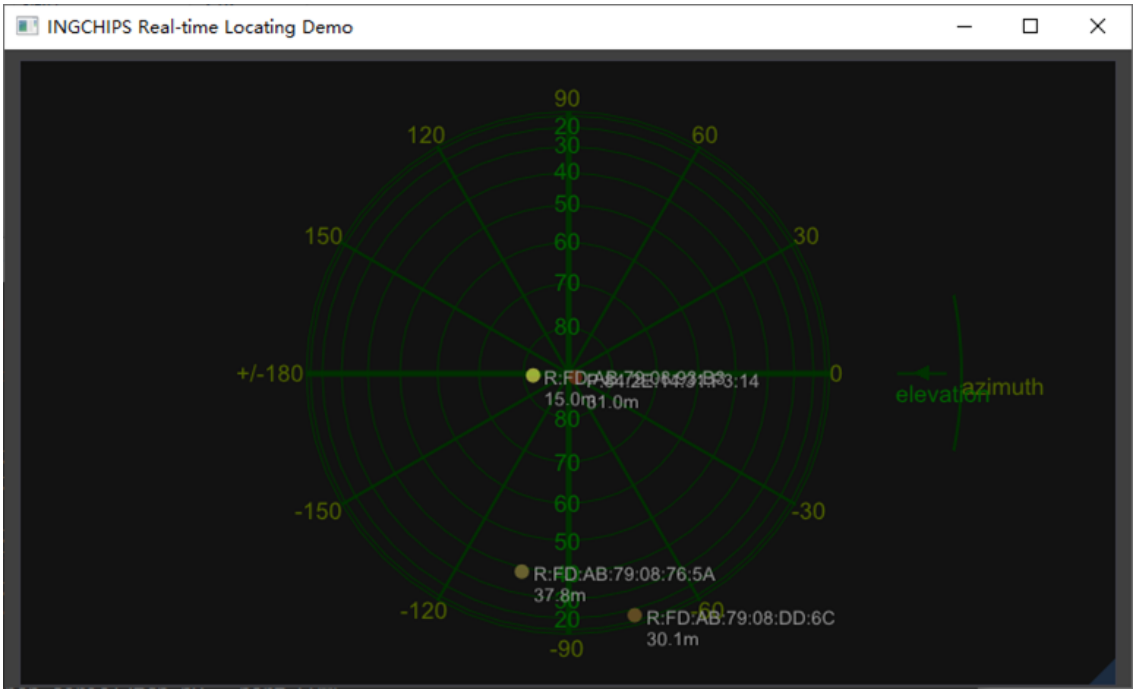

Figure 3.5: Main Window of Real-time Locating Demo

The *real-time* used here means that data fed to *Real-time Locating Demo* is assumed to be in real-time, and the wall clock when each CTE report is received is utilized in post processing of AoA estimations.

Table 3.1: Comparison of DX and OpenGL Demo variants

Variant	Windows	Support of High DPI display
DirectX 12	Windows x64 (>= 10)	Excellent
OpenGL	Windows x64	Not good

Find the COM port of the UART2USB kit in Windows Device Manager. Take "COM7" as an example.

Open a console (prompt, or terminal) window, make sure that the Python executable is in the search paths, use `cd` command to change directory to "RLT" which is in SDK's installing directory, and run following command to start feeding data to *Real-time Locating Demo*:

```
python serial2tcp.py --port COM7
```

Now, *Real-time Locating Demo* will start showing AoA results on its main window (Figure 3.5). Azimuth (θ) and elevation (ϕ) results are shown on a sky plot. The center of the plot corresponds to the zenith seen from the board, i.e., it's right above the center of the board.

Real-time Locating Demo supports detection of multiple tags. To simplify the demo, *Peripheral LED & CTE* has been designed to randomly generate a new address each time it got powered on, therefore, just reset the tag, and a **new** tag will appear on *Real-time Locating Demo*. If CTE from a tag has not been received for a period of time, *Real-time Locating Demo* will forget the tag, and remove it from the plot.

3.3 AoA Accuracy

As direction finding solutions will be deployed in different scenarios and SIG has not defined **standard** scenarios for accuracy assessment, it's impossible to give accuracy results that can be used for evaluation/comparison of different AoA solutions.

To put it another way, we developed a low cost test platform and performed real environment testing with the following results.

3.3.1 Test Platform

Figure 3.6 is the overview of the test platform. The antenna array board is turned by a servo which is controlled by Controller. An OLED display is used for displaying information during the test. Data output from the array board is saved to an SD card. ING918xx development board is used to implement the Controller.

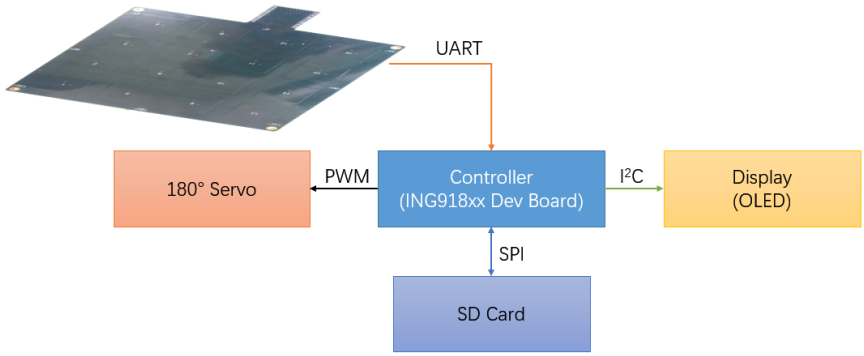

Figure 3.6: AoA Test Platform

3.3.2 Test Results with 4x4 Antenna Array Board (Rectangle)

The following devices were used for all antenna array accuracy measurements in this section:

- 4x4 Antenna Array Board (Rectangle) (CTE Receiver)
- ING918xx development boards (Tags - CTE Transmitters)

3.3.2.1 Environment 1 (1 locator, 1 tag)

- Location: Indoor, office corner, lots of objects inside measurement area
- Locator height from floor: $\approx 0.7m$
- Locator is placed horizontally on the test platform
- Tag height from Locator: $\approx 1.5m$
- Tag is in different poses

Setup of this test is shown in Figure 3.7. Locator rotates between $0 \sim 360^{\circ}$. This test is carried with the tag in three different poses. Figure 3.8 is a closeup of the tag with different poses.

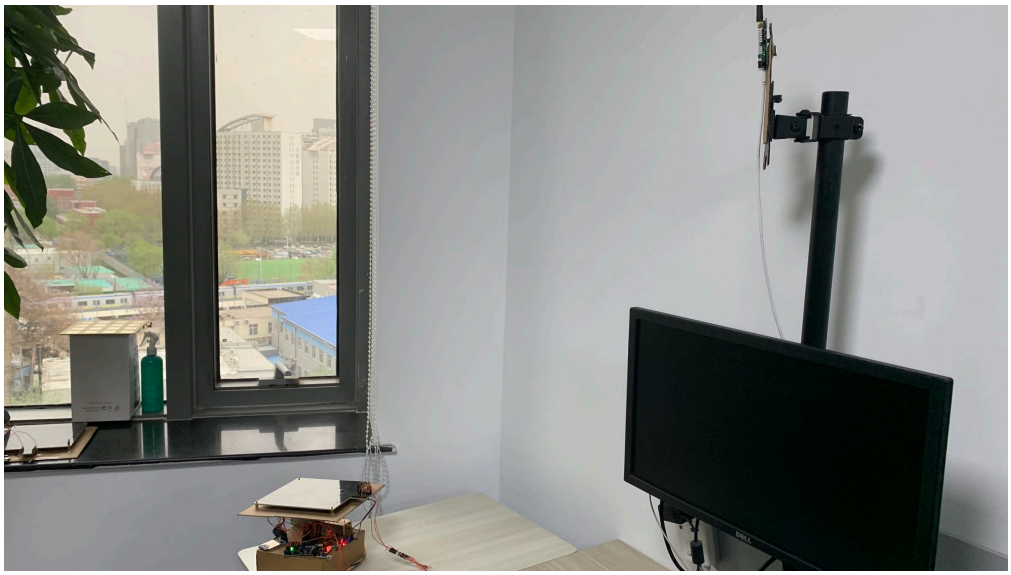

Figure 3.7: Test Setup (Environment 1)

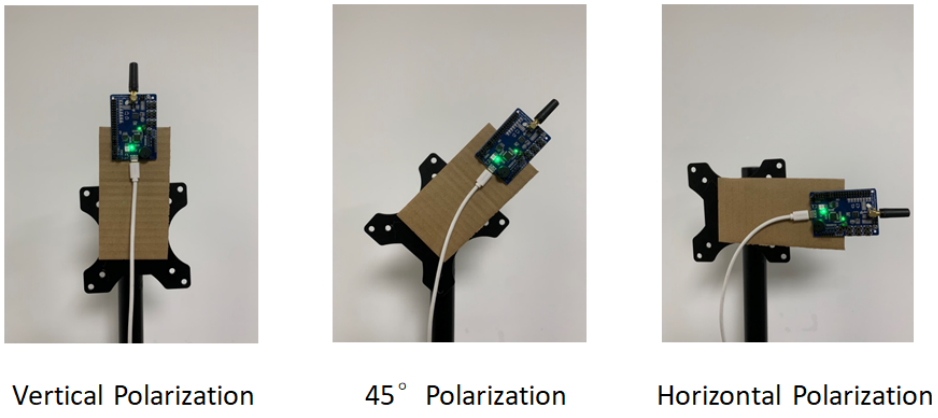

Figure 3.8: Tag Pose Closeup

Azimuth error vs. azimuth angle when tag is in vertical polarization is shown in Figure 3.9. Azimuth error vs. azimuth angle when tag is in 45° polarization is shown in Figure 3.10. Azimuth error vs. azimuth angle when tag is in horizontal polarization is shown in Figure 3.11.

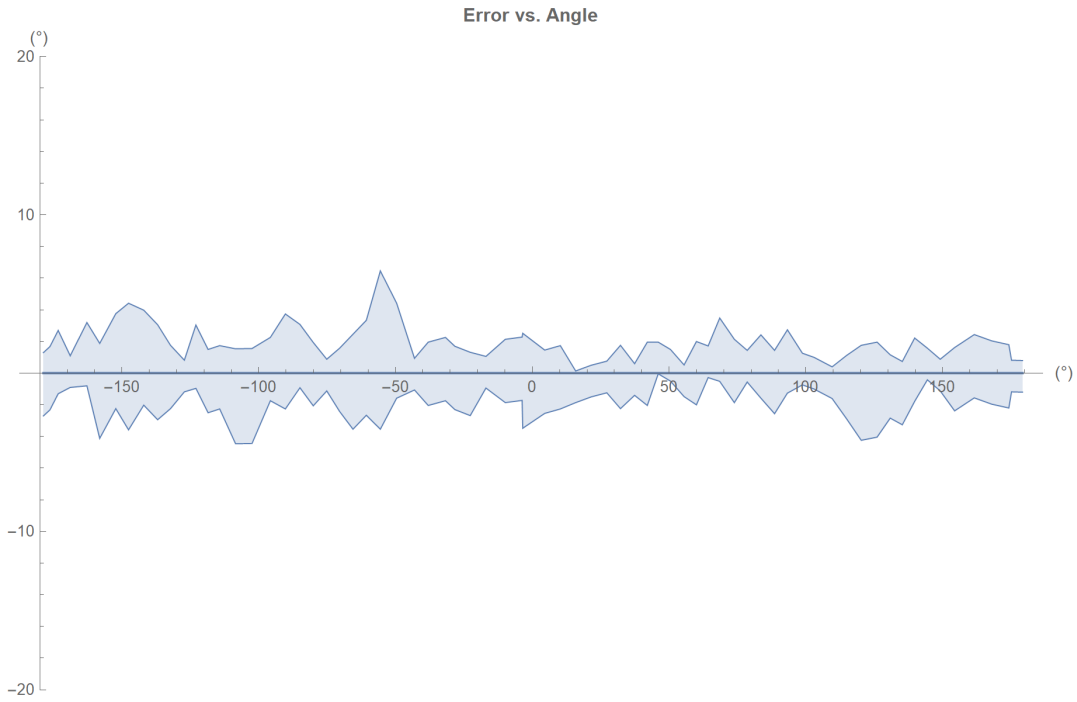

Figure 3.9: Azimuth Error vs. Azimuth Angle (Environment 1, Vertical Polarization)

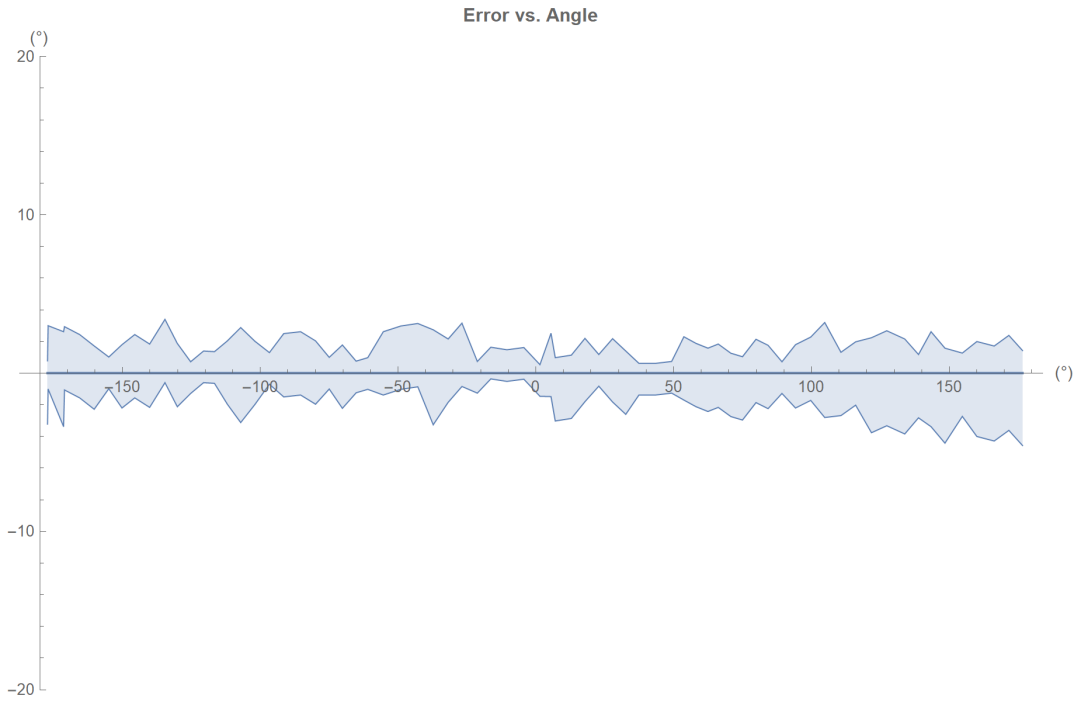

Figure 3.10: Azimuth Error vs. Azimuth Angle (Environment 1, 45 Degree Polarization)

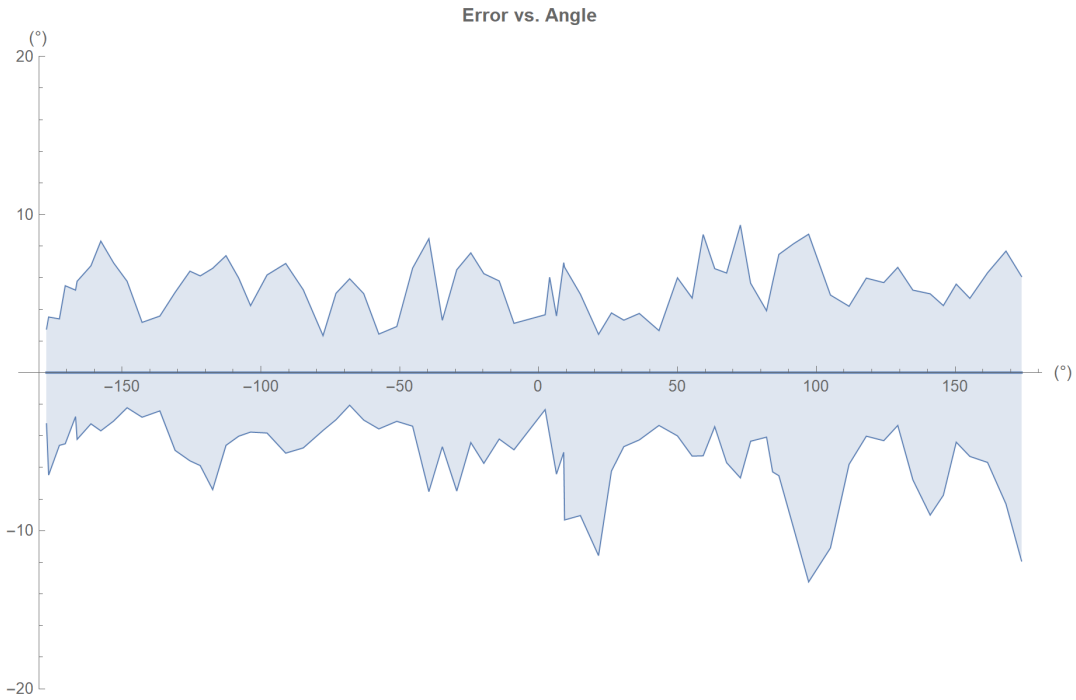

Figure 3.11: Azimuth Error vs. Azimuth Angle (Environment 1, Horizontal Polarization)

Azimuth measurements summary:

- Average azimuth errors: $\pm 3.5^{\circ}$
- Worst-case azimuth errors: $\pm 14^{\circ}$

3.3.2.2 Environment 2 (1 locator, 1 tag)

- Location: Indoor, office corner, lots of objects inside measurement area
- Locator height from floor: $\approx 1.2m$
- Locator is placed vertically on the test platform
- Tag height from floor: $\approx 1.2m$
- Tag is in different poses

Setup of this test is shown in Figure 3.12. Locator rotates between $0 \sim 180^{\circ}$. So, the elevation goes from 0° to 90°, i.e. the zenith seen from the board, and then goes back to 0° again. This test is carried with the tag in three different poses. Figure 3.8 is a closeup of the tag with different poses.

Elevation error vs. elevation ideal angle when tag is in vertical polarization is shown in Figure 3.13. Elevation error vs. elevation ideal angle when tag is in 45° polarization is shown in Figure 3.14. Elevation error vs. elevation ideal angle when tag is in horizontal polarization is shown in Figure 3.15.

Elevation measurements summary:

- Worst-case elevation errors in full scale: $\pm 35^{\circ}$
- Worst-case elevation errors in a right cone which has an opening angle of 120° : $\pm 5^{\circ}$

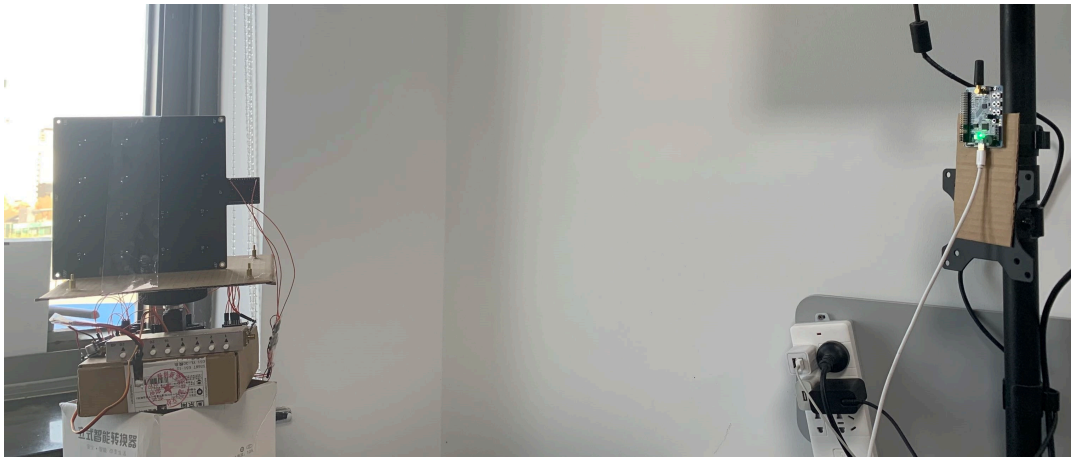

Figure 3.12: Test Setup (Environment 2)

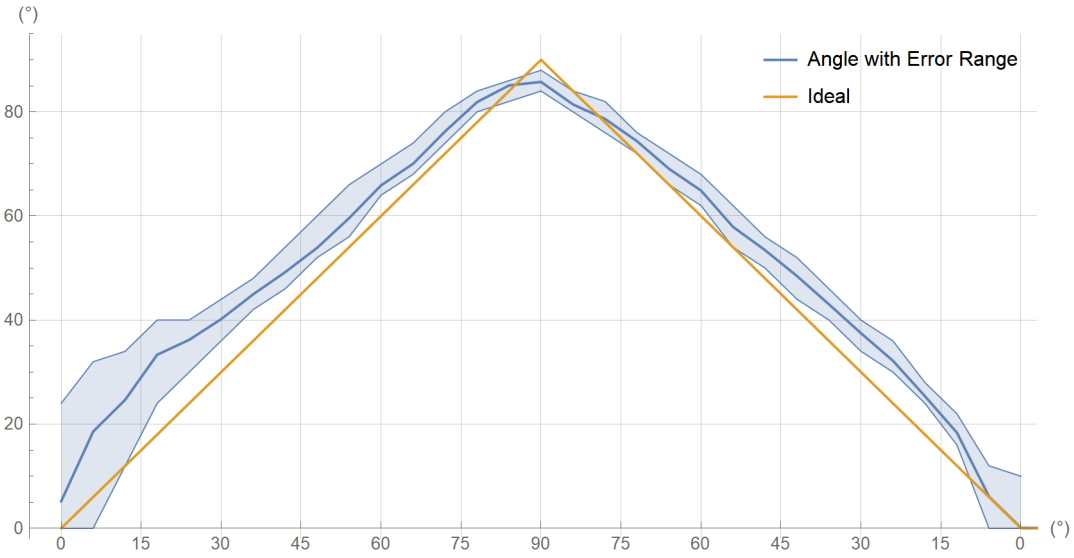

Figure 3.13: Elevation Error vs. Elevation Ideal Angle (Environment 2, Vertical Polarization)

3.3.3 Test Results with 3x3 Antenna Array Board (Round)

The following devices were used for all antenna array accuracy measurements in this section:

- 3x3 Antenna Array Board (Round) (CTE Receiver)
- ING918xx development boards (Tags - CTE Transmitters)

3.3.3.1 Environment 1 (1 locator, 1 tag)

- Location: Indoor, office corner, lots of objects inside measurement area
- Locator height from floor: $\approx 0.7m$
- Locator is placed horizontally with four different poses
- Tag height from floor: $\approx 1.2m$

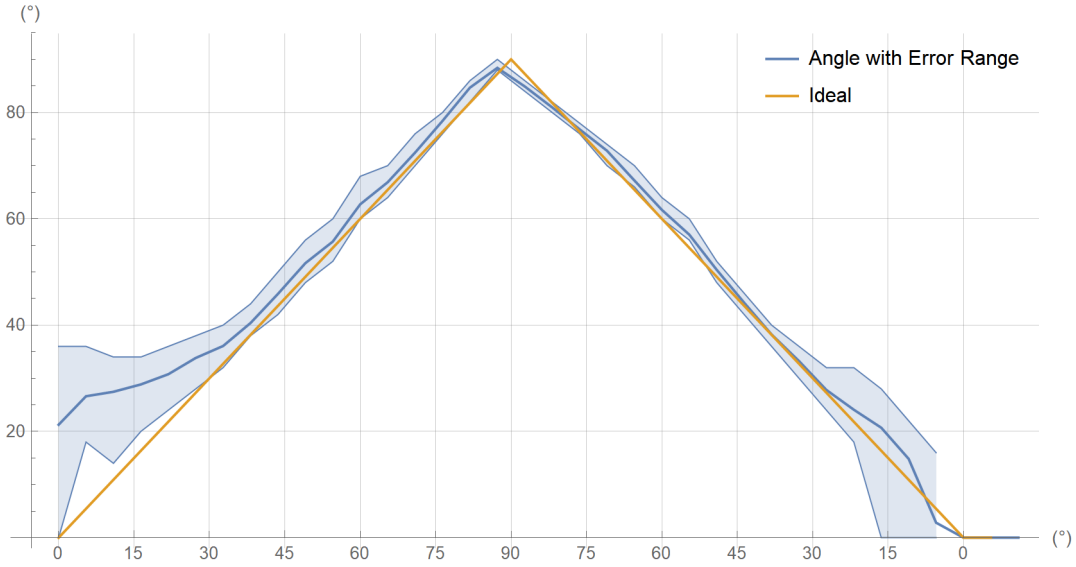

Figure 3.14: Elevation Error vs. Elevation Ideal Angle (Environment 2, 45 Degree Polarization)

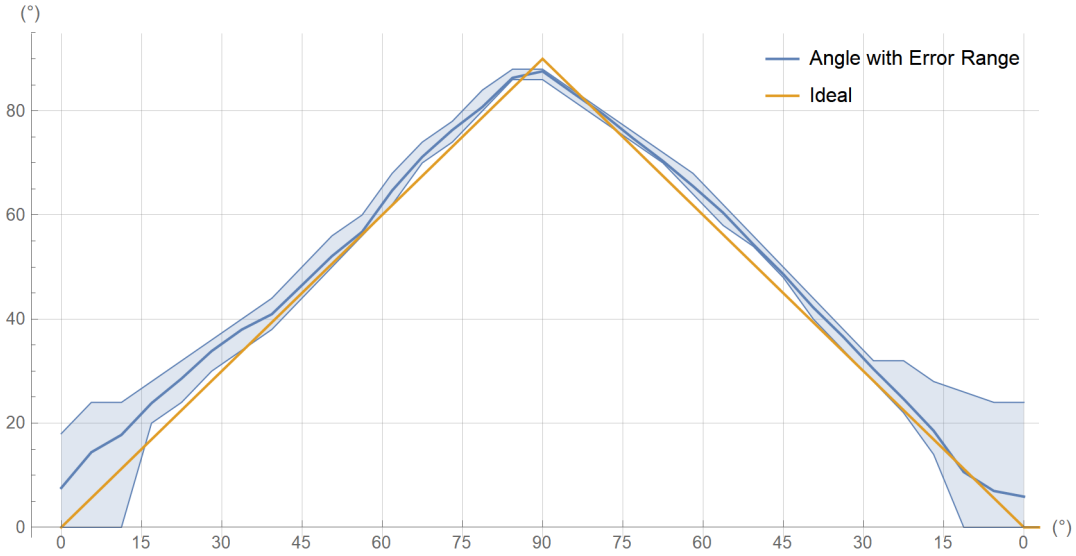

Figure 3.15: Elevation Error vs. Elevation Ideal Angle (Environment 2, Horizontal Polarization)

- Tag is in horizontal polarization

Azimuth & elevation are measured at four poses and summarized in Table 3.2 and Table 3.3. Summary of the result:

- Worst-case azimuth errors: $\pm 7.0^{\circ}$
- Worst-case elevation errors: $\pm 8.2^{\circ}$
- Average 3σ azimuth error: 6.3°
- Average 3σ elevation error: 6.9°

Table 3.2: Azimuth Results with 3x3 Antenna Array Board (Round)

Pose #	Mean	Standard Deviation	(Min, Max)
1	0.9°	1.7°	$[-4^\circ, 6^\circ]$
2	88.5°	1.5°	$[84^\circ, 92^\circ]$
3	175.0°	2.4°	$[168^\circ, 178^\circ]$
4	-88.1°	2.7°	$[-94^\circ, -82^\circ]$

Table 3.3: Elevation Results with 3x3 Antenna Array Board (Round)

Pose #	Mean	Standard Deviation	(Min, Max)
1	45.8°	3.2°	$[40^\circ, 54^\circ]$
2	44.8°	3.0°	$[38^\circ, 54^\circ]$
3	51.5°	1.5°	$[50^\circ, 56^\circ]$
4	54.6°	1.6°	$[50^\circ, 58^\circ]$

3.4 Trouble Shooting

- *Real-time Locating Demo* shows nothing

  Use other COM tools to check if there are lots of data received from the array board

  - If Yes, maybe there are compatibility issues between `pyserial` and the UART2USB kit.

    Try another UART2USB kit. And check tips below for a better test environment.
  - If No, check if the tag is working properly

    The chip series must be support CTE, such as ING9188xx, ING9186xx, etc, while ING9187xx won't work.

- *Real-time Locating Demo* shows wrong results

  - Rotate & move the tags slowly to see if there are *good* positions where *Real-time Locating Demo* gives good results
  - Make sure that the array board is not upside down
  - Re-start *Real-time Locating Demo* with command line parameter `-plot`, then it will show SNR on each antenna. If SNRs are all lower than 10dB, follow below tips for a better test environment:
    * Test in a large & open environment. Desktop environment where there are lots of things reflecting or blocking radio waves will worsen AoA results significantly.
    * Use highest transmission power in tags. The highest Tx power is already used in the SDK example. The antennas in tags should be checked.

    If SNRs are all excellent, contact us for further technical support.

Chapter 4

SDK Support of Direction Finding

SDK supports four types of CTE solutions, two are defined by Bluetooth SIG in Bluetooth Core Specification, and the other two are INGCHIPS proprietaries.

4.1 Types of Solutions

4.1.1 Connection Based Solution

Connection based solution is defined by Bluetooth SIG. A connection must be established between two partners, and then CTE signals are sent upon the requests from a central device. This is demonstrated in *Peripheral LED & CTE* and *Central CTE*.

In *Peripheral LED & CTE*'s advertising data, the availability of CTE service is broadcasted (Figure 4.1).

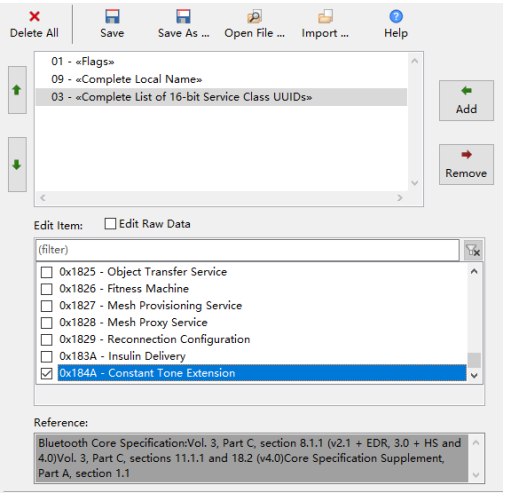

Figure 4.1: Advertising of CTE Service

There is an *Enable* characteristic in CTE service. A GATT client should set it to `true` before requesting the peripheral to send CTE.

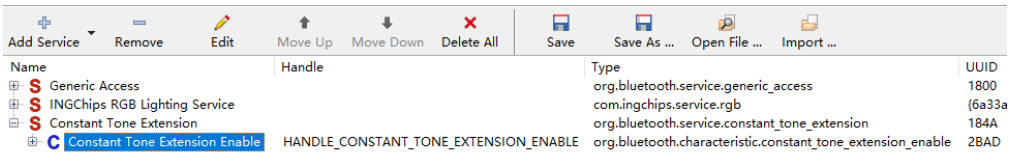

Figure 4.2: Service Definition of CTE

4.1.2 Connectionless Solution

Connectionless solution is also defined by Bluetooth SIG. The CTE transmitter sends CTE in a periodic advertising set. The receiver has to establish synchronization with the periodic advertising set firstly, and then CTE can be received. This is demonstrated in *Periodic Advertiser* and *Periodic Scanner*.

4.1.3 Proprietary Solution #1

This is based INGCHIPS BLE extension APIs, and provides lots of flexibilities, such as:

- More bits in IQ samples (all bits from RF can be utilized)
- More samples within a single sample slot
- Full control over access address, CRC initialization value for privacy
- Full control over channel selection

Read *BLE Extensions Reference* for more information.

4.1.4 Proprietary Solution #2

This type tries to provide compatibility of Silicon Labs proprietary AoA[1], which sends CTE in extended advertisements. This is also demonstrated in *Peripheral LED & CTE* and *Central CTE*.

Compatibility status:

- A tag sending CTE with proprietary direction finding #2 can be received by Silicon Labs AoA locator
- A tag sending CTE with Silicon Labs proprietary AoA can be received by INGCHIPS AoA locator **if** the least significant byte of the tag's device address (no matter if it is a public or random address) is **0x14** or some other proper values.

[1] `https://www.silabs.com/documents/public/quick-start-guides/qsg175-direction-finding-solution-quick-start-guide.pdf`

4.2 Recommendation

It is **strongly** recommended to use proprietary direction finding solution, because of several shortcomings in the two Bluetooth SIG defined solutions:

- Scalability

  Both of the two Bluetooth SIG defined solutions do not scale very well: devices can't support lots of connections or periodic advertising synchronizations simultaneously.

- Latency

  Both of the two Bluetooth SIG defined solutions have pre conditions before the reception of CTE, which is time consuming, especially in the case of connectionless solution.

- Robustness

  In the case of connection less solution, it's rather difficult & tricky to keep synchronization with several periodic advertising sets.

Chapter 5

Further Information

5.1 Algorithm Integeration

A command line tool called `alg` which performs the same algorithm as *Real-time Locating Demo* is provided in the *RTL* sub-directory of SDK. It uses ASCII strings over standard input/output as external interface, and can be integrated into locating systems easily.

`alg` reads a whole line as input, and output a line containing the result in JSON format, as shown in Table 5.1. The input is a prefix "EXT0:" followed by the Base64 encoding of below packed data structure `le_meta_pro_connless_iq_report`:

```
typedef struct le_iq_sample
{
    int8_t i;
    int8_t q;
} le_iq_sample_t;

typedef struct le_meta_pro_connless_iq_report
{
    uint8_t addr_type;
    uint8_t addr[6];
    phy_type_t rx_phy;
    uint8_t  channel_index;
    int16_t  rssi;
    uint8_t  rssi_ant_id;
    uint8_t  cte_type;
    uint8_t  slot_durations;
    uint8_t  packet_status;
    uint8_t  sample_count;
    le_iq_sample_t samples[0];
} le_meta_pro_connless_iq_report_t;
```

Table 5.1: Fields in the JSON Output

Field	Note
status	A string describing if error occurs in the processing of the input.
id	A string representing the tag's address. Only available when status is "ok".
azimuth	Estimated azimuth of the tag in degree. Only available when status is "ok".
elevation	Estimated elevation of the tag in degree. Only available when status is "ok".

An example session of `alg`:

```
EXT0:ABTzMRQuhAERPv4AAAEAUt7PziEgMjTh4s3LHRw1N+gdE5o/Ka0+1ykTnyI
E3jv6NjPB+NjYId40J8MH8LQ59g0fhwBKyUr3HCSc6xXlNxgVRszb8cks8xo8xuo
WtzcQ/SKWx1v3RRsHLbG/HvImLfNJ5MgLyC4I/UHY0TfKJyrtHLydVCUtOvIr2qE
iAgk70DgCwCXVJB7cNfXDS+0NOeEO+Ig7Rg1J3x4MmhwT7Ti7GyHK
{"status": "ok", "id": "P:84:2E:14:31:F3:14", "azimuth": 174.0,
"elevation": 48.0 }
```

Note that, the input must be in a single line.

5.2 Evaluation of Different Antenna Array Types

To assist the evaluation of different antenna array types, both `alg` and *Real-time Locating Demo* might accept an additional command option to specify a sub portion of the whole antenna array board by "`alg -array {TYPE}`" or "`RTLing_xx -array {TYPE}`". When using this option, the switching pattern used in the firmware (*Central CTE* example) should be updated accordingly.

Table 5.2: Supported Array Types for 4x4 Antenna Array Board (Rectangle)

{TYPE}	Switching Pattern	Note
4x4	$0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15$	Whole array is used (default)
3x3	$0, 1, 2, 4, 5, 6, 8, 9, 10$	3×3 URA sub-array
2x2	$0, 1, 4, 5$	2×2 URA sub-array
1x4	$0, 1, 2, 3$	1×4 ULA sub-array
1x3	$0, 1, 2$	1×3 ULA sub-array
1100	$0, 1$	1×2 ULA sub-array
1010	$0, 2$	1×2 ULA sub-array
1001	$0, 3$	1×2 ULA sub-array

Table 5.3: Supported Array Types for 3x3 Antenna Array Board (Round)

{TYPE}	Switching Pattern	Note
o3x3	$0, 1, 2, 3, 4, 5, 6, 7, 8$	Whole array is used

For example, if we want to evaluate the performance of 1x4 ULA sub-array on 4x4 Antenna Array Board (Rectangle), then

1. Change the switching pattern used in *Central CTE* to $0, 1, 2, 3$,
2. Build *Central CTE* and download it to the Antenna Array Board,
3. Start `alg` by `alg -array 1x4`, or,
4. Start *Real-time Locating Demo* by "`RTLing_dx -array 1x4`" or "`RTLing_gl -array 1x4`".

Chapter 6

Revision History

Version	Notes	Date
1.0	Draft	2021-12-22
1.1	Minor updates	2022-01-10
1.2	Add “Evaluation of Different Antenna Array Types”	2022-02-18
1.3	Add “AoA Accuracy”	2022-04-12
1.4	Add “3x3 Array Board (Round)”	2022-08-22
1.4.1	Update data sample	2022-11-07