UWB Anchor Initiator/Tag Responder Demonstration

Author: Scott Lederer, Ezurio
Date: *November 4, 2024*

NOTE: *This document is included as part of an application archive for use with the Sera NX040 DVK and Canvas Firmware. If you did not receive the archive, contact your Ezurio sales representative.*

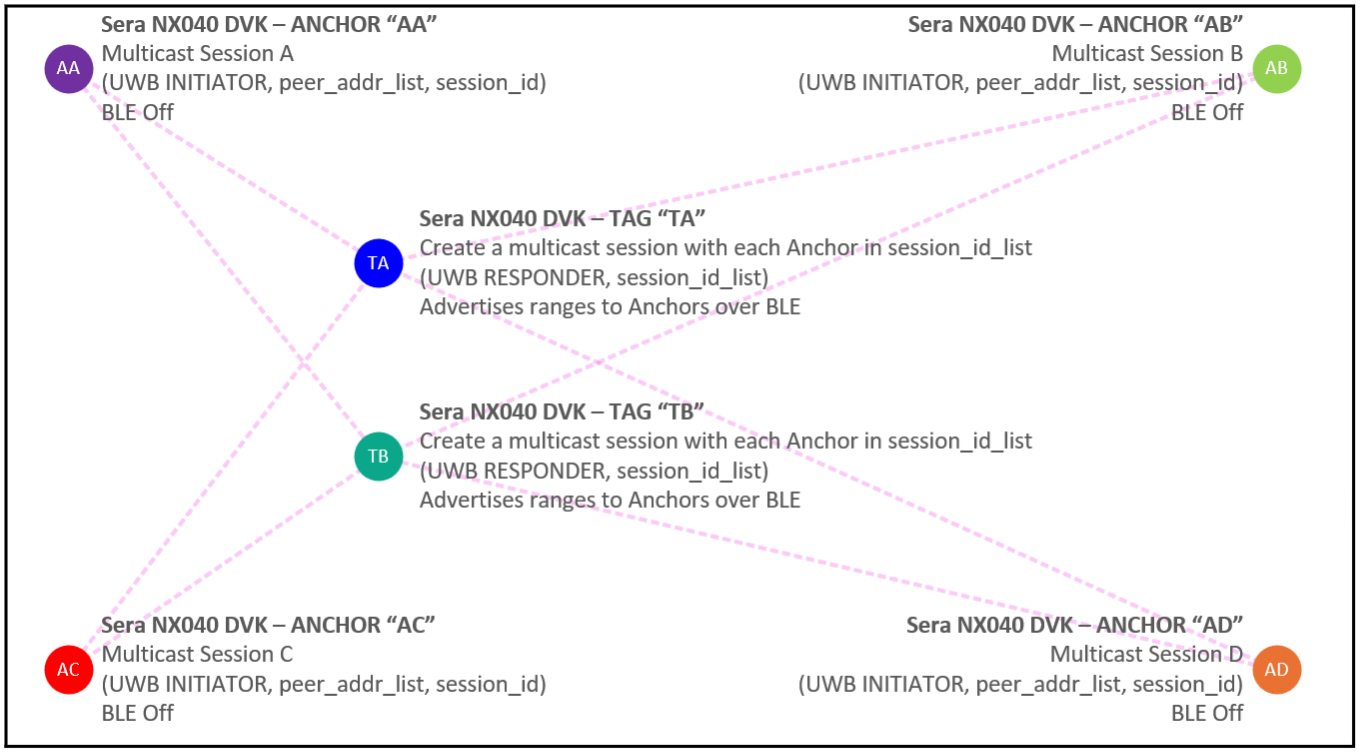
Description

Ezurio has developed a Python script application consisting of several .py script files in support of a UWB ranging demonstration where fixed location **Sera NX040 DVK** boards (a.k.a., "Anchors") are configured as **Initiators** and several mobile **Sera NX040 DVK** boards (a.k.a., "Tags") are configured as **Responders**. All devices require a set of configuration values to be assigned via the REPL console ahead of running the demo to setup ranging parameters between the boards.

Each Anchor (Initiator)has a configuration parameter called **peer_addr_list** containing a list of short addresses (16 bit Integers) for each **Tag** that the Anchor should range with. Anchors have a **session_id** configured, identifying its UWB multicast session that Tags/Responders can start a ranging session with. Each Tag has a configuration parameter called **session_id_list** containing a list of **session_id** values corresponding to each Anchor's ranging session.

System Overview

The following diagram illustrates a typical setup using 6 of the Sera NX040 DVKs in a Tag and Anchor ranging system as intended by this set of Python application scripts.



Example configuration with 4 Anchors and 2 Tags where Anchors are configured in corners of a room.

NOTE: In this sample application, Anchors are not configured to range with one another and therefore do not report distance between Anchors measured over UWB. If distances are needed for triangulation of the Tags, take physical measurements between Anchors using a measuring tape or similar. This is a tradeoff made to allow a maximum number of Tag devices to participate in the system.

Configuration

The python script application gets its configuration from an instance of the **Config** class defined in the config.py file. To set a parameter from the REPL, use a command like the following:

```
>>> config.config['ranging_interval_ms'] = 200
```

This command sets the **ranging_interval_ms** parameter to a value of 200 milliseconds. Use similar syntax to assign values to all other parameters.

To save the configuration to non-volatile storage (i.e., a configuration file) that gets loaded on startup, type the following at the REPL:

```
>>> config.save()
```

This saves the configuration to a file in the Flash filesystem. To apply the new configuration, reboot the device by typing **Ctrl+D** at the REPL or pressing the RESET button on the DVK board.

To get started, program a recent version of Canvas firmware onto the Sera NX040 DVKs and load the .py script files from the "Anchor_Initiator-Tag_Responder.zip" archive onto the DVK boards using Xbit VS Code. Once programmed, the script/application needs to be configured by connecting a serial terminal to the REPL console of each board. This can also be done from Xbit VS Code using the **TERMINAL** window. The sections below describe how to setup the Sera NX040 DVK boards as either **Anchors** or **Tags**.

Parameter Table:

Anchors and Tags use the following parameters to control their ranging operations:

Data Type	Parameter	Description	Default Value
uint32_t	range_led	RGB value for the LED when a ranging event occurs.	0x003F00
uint32_t	error_led	RGB value for the LED when a ranging error occurs.	0x000000
uint32_t	base_led	RGB value for the LED when not ranging.	0x000F00
uint16_t	local_addr	Local short address for this device	Last 2 bytes of device unique ID
Array(uint16_t)	peer_addr_list	**ANCHOR ONLY**: Array of uint16 values matching the short address of each Tag this anchor will range with.	[ ] (empty array)
uint32_t	ranging_interval_ms	Determines the period at which ranging measurements are attempted (milliseconds).	1000
uint8_t	anchor_mode	Mode of operation for this device 0: Tag mode 1: Anchor mode	0 : Tag Mode
bytes	dev_id	Unique identifier (8 bytes)	Defaults to hardware unique ID
String	ble_name	BLE complete name included in the BLE advertisement if enabled.	'UWB Simple'
uint32_t	session_id	**ANCHOR ONLY**: Unique session ID for the multicast ranging session this anchor will offer.	0x1234xxxx where the xxxx bytes match the short address of this device
Array(uint32_t)	session_id_list	**TAG ONLY**: List of session_id values for each Anchor this Tag should start a ranging session with.	[ ] (empty array)
uint32_t	advertising_interval_ms	**TAG ONLY**: BLE advertising interval (milliseconds)	100

Anchors

To configure the Sera NX040 DVK as an “Anchor”, the following parameters (at a minimum) must be configured with valid values:

- **base_led**: Set to a 32 bit value indicating 0xRRGGBB value for the LED. This is useful for identifying this Anchor among other Anchor boards.
- **range_led**: Set to a 32 bit value indicating 0xRRGGBB value for the LED. This is the color of the LED that will blink briefly when each ranging event occurs. It is recommended to set this to a slightly brighter value of the same color used for **base_led**.
- **peer_addr_list**: Set this to an array (bracket notation) of Integers corresponding to the short address for each **Tag** in the system. Capture the short address of each Tag by inspecting the startup message at the REPL where both the device id and short address are displayed as hexadecimal values.
- **ranging_interval_ms**: The number of milliseconds between each ranging event. Valid values for this must be multiples of 50ms if default slot duration and number of ranging rounds is used. This value must match the value programmed into the **Tags** in the system.
- **anchor_mode**: Set this to **1** to indicate the device is acting as an Anchor role in the system.
- **ble_name**: Set a unique BLE name to identify the device. Note by default Anchors do not implement any BLE functionality but do still print their ble_name value to the REPL on startup.
- **session_id**: Can be left at the default value which is based on the device’s short address or a custom value can be used here. This value must be included in the **session_id_list** of the Tags in the system.

Once these configuration values are set, remember to type **`config.save()<ENTER>`** to save the values you’ve assigned to be applied on the next reset.

Tags

To configure the Sera NX040 DVK as a "Tag", the following parameters (at a minimum) must be configured with valid values:

- **base_led**: Set to a 32 bit value indicating 0xRRGGBB value for the LED. This is useful for identifying this Tag among other Tag boards.
- **range_led**: Set to a 32 bit value indicating 0xRRGGBB value for the LED. This is the color of the LED that will blink briefly when each ranging event occurs. It is recommended to set this to a slightly brighter value of the same color used for **base_led**.
- **session_id_list**: Set this to an array (bracket notation) of Integers corresponding to the short address for each **Anchor** in the system. Capture the short address of each Anchor by inspecting the startup message at the REPL where both the device id and short address are displayed as hexadecimal values.
- **ranging_interval_ms**: The number of milliseconds between each ranging event. Valid values for this must be multiples of 50ms if default slot duration and number of ranging rounds is used. This value must match the value programmed into the **Anchors** in the system.
- **anchor_mode**: Set this to **0** to indicate the device is acting as an Tag role in the system.
- **ble_name**: Set a unique BLE name to identify the device. Note by default Tags advertise over BLE and can be identified by the ble_name specified by this parameter.

Once these configuration values are set, remember to type **`config.save()<ENTER>`** to save the values you've assigned to be applied on the next reset.

Viewing Range Data

REPL UART via Serial Port

The Sera NX040 DVK boards configured as "Tags" will display range to each Anchor via the REPL serial port using a simple JSON formatted string.

Here is sample output from a Tag showing range to 4 anchors:

```
{"local":"b015","ebcd":113,"6f95":153,"562a":140,"3367":144}
{"local":"b015","ebcd":122,"6f95":153,"562a":140,"3367":139}
{"local":"b015","ebcd":121,"6f95":157,"562a":140,"3367":142}
{"local":"b015","ebcd":118,"6f95":157,"562a":140,"3367":142}
{"local":"b015","ebcd":119,"6f95":156,"562a":139,"3367":144}
{"local":"b015","ebcd":113,"6f95":156,"562a":139,"3367":144}
{"local":"b015","ebcd":120,"6f95":155,"562a":139,"3367":140}
{"local":"b015","ebcd":118,"6f95":155,"562a":139,"3367":141}
{"local":"b015","ebcd":120,"6f95":155,"562a":138,"3367":138}
{"local":"b015","ebcd":121,"6f95":155,"562a":138,"3367":138}
{"local":"b015","ebcd":120,"6f95":158,"562a":138,"3367":140}
{"local":"b015","ebcd":117,"6f95":158,"562a":138,"3367":140}
{"local":"b015","ebcd":117,"6f95":155,"562a":141,"3367":142}
{"local":"b015","ebcd":123,"6f95":155,"562a":141,"3367":141}
```

Bluetooth Advertisements

The Sera NX040 DVK boards configured as “Tags” will advertise over BLE. The advertisement contains the BLE name configured for the device along with a manufacturer-specific data element containing the short address of and range to each Anchor it is configured to start a ranging session with.

Advertising Format

BLE peripherals will advertise using the following format. This includes a fixed header section using manufacturer-specific data. This fixed section is followed by one or more TLV entries depending on device state and configuration.

Header:

Byte Offset	Field	Size (bytes)	Value
0	companyId = Laird	2	0x0077
2	protocolId = BLE UWB Tag	2	0x000C
4	networkId	2	
6	flags	2	
8	deviceId	8	
16	timestamp	4	

Flags:

Flag	Description
0x0001	Set if configured, clear if not configured
0x0002	Set if acting as an anchor (fixed position), clear if acting as a tag (mobile)
0x0004	Set if capable of BLE scanning, clear if advertising only
0x0008	Set if capable of UWB responder role, clear if initiator only
0x0010	Set if one or more ranging sessions are active, clear if no active ranging sessions

TLVs (one per ranging peer):

Byte Offset	Field	Size (bytes)	Value
0	type = Range to Device	1	0x00
1	length = 4	1	0x04
2	shortAddr	2	XXXX
4	range	2	XXXX cm

Tips and Tricks

Because of the intense nature of UWB ranging performed by this sample application, the REPL console may become unresponsive to tools like Xbit VS Code for purposes of transferring script files, etc. while the application script is running.

To assist with this, a feature has been included in the scripts to **skip directly to the REPL prompt upon RESET when the user button on the Sera NX040 DVK is held down**. This allows manual interaction with the board including editing of scripts using Xbit VS Code without having to contend with the UWB ranging sessions taking up CPU time.

If you'd like to inspect or modify the configuration of a board when in this mode (i.e. when the board was started while holding down the USER button to enter the REPL and not begin UWB ranging), you can first instantiate the **Config** object and then modify the configuration values like you normally would when the full application is running.

```
>>> config = Config()
>>> config.config['ranging_interval_ms'] = 200
```