User Guide

Type 2BP UWB Module EVK with Raspberry Pi 4/Linux

User Guide - Rev. D

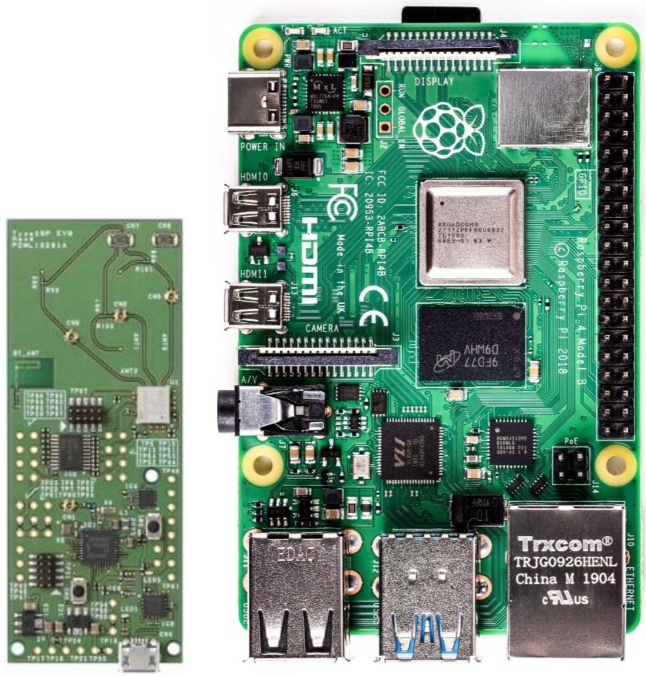


Table of Contents

Figures

Tables

About This Document

This document describes the steps to drive Murata Type 2BP EVK from Raspberry Pi 4 (model B).

Audience & Purpose

This document is intended for the RF engineers and developers who will drive Murata Type 2BP EVK from Raspberry Pi 4 (model B).

Document Conventions

Table 1 describes the document conventions.

Table 1: Document Conventions

Conventions	Description
	Warning Note Indicates very important note. Users are strongly recommended to review.
	Info Note Intended for informational purposes. Users should review.
	Menu Reference Indicates menu navigation instructions. **Example**: Insert→Tables→Quick Tables→Save Selection to Gallery
	External Hyperlink This symbol indicates a hyperlink to an external document or website. **Example**: Type 2BP Product Page Click on the text to open the external link.
	Internal Hyperlink This symbol indicates a hyperlink within the document. **Example**: Overview Click on the text to open the link.
`Console input/output or code snippet`	**Console I/O or Code Snippet** This text ***Style*** denotes console input/output or a code snippet.
`# Console I/O comment` *`// Code snippet comment`*	**Console I/O or Code Snippet Comment** This text ***Style*** denotes a console input/output or code snippet comment. • Console I/O comment (preceded by "#") is for informational purposes only and does not denote actual console input/output. • Code Snippet comment (preceded by "//") may exist in the original code.

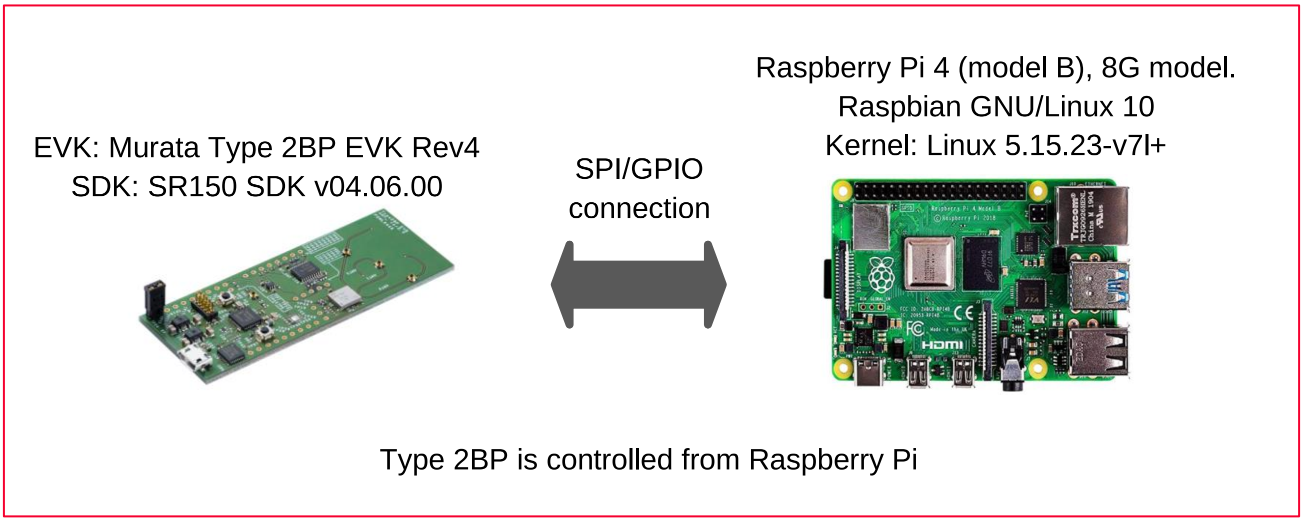
1 Overview

This document details the procedure to drive Murata Type 2BP EVK from Raspberry Pi 4 (Model B). By following this procedure, you can drive Type 2BP from Raspberry Pi / Raspberry Pi OS.

Hardware Components

- **EVK:** Murata Type 2BP EVK Rev4.1
  - **SDK:** SR150 SDK v04.06.00
- **Raspberry Pi 4:** Model B, 8G model.
  - **Operating System:** Raspbian GNU/Linux 10
  - **Kernel:** Linux 5.15.23-v7l+

Figure 1: Hardware Setup



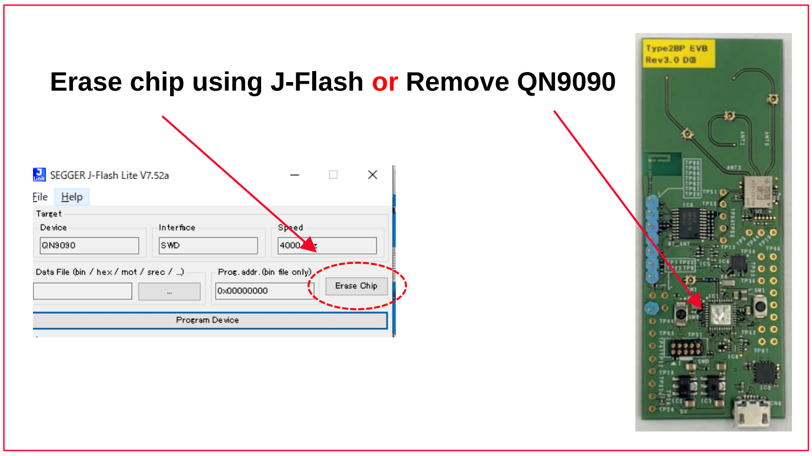
Procedure may be different based on differences of environment. This is just an example of the procedure which worked in Murata environment as shown in **Figure 1**.

2 Step 1: Configuring the Hardware

To control Type 2BP from Raspberry Pi, the following hardware settings need to be changed.

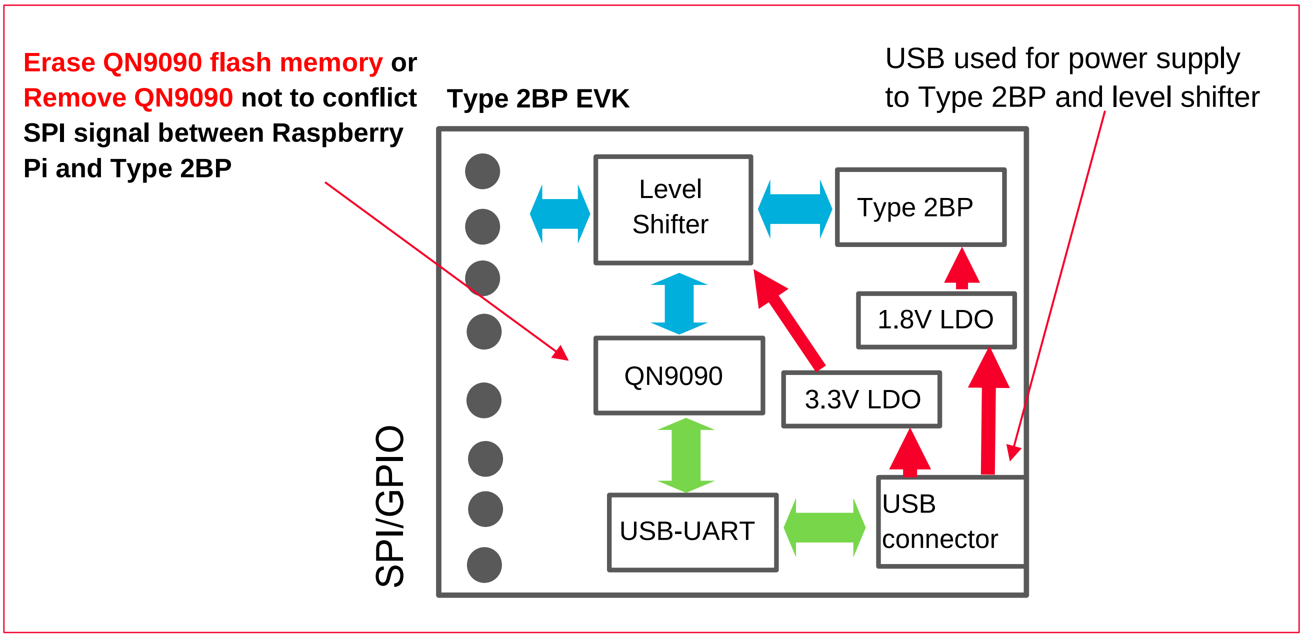
1. Erase QN9090 flash memory or Remove QN9090 on Type 2BP EVK. This is to prevent any conflict of SPI signals between Raspberry Pi and Type 2BP EVK.

Figure 2: Eliminate Possible QN9090 Interference



The flow of configuration is shown in **Figure 3**.

Figure 3: Configuration Flow



2. Attach pin headers to Type 2BP EVK SPI, GPIO and GND pins, and connect with Raspberry Pi 4 with wires, as described in Section 2.1.

2.1 The Pin Mapping

The interconnection between Raspberry Pi 4 and Type 2BP EVK is shown in **Figure 4** (for Type 2BP EVK Rev 3.x) and **Figure 5** (for Type 2BP EVK Rev 4.0 or later).

Figure 4: Pin Mapping of Raspberry Pi 4 to Type 2BP EVK (Rev 3.x)

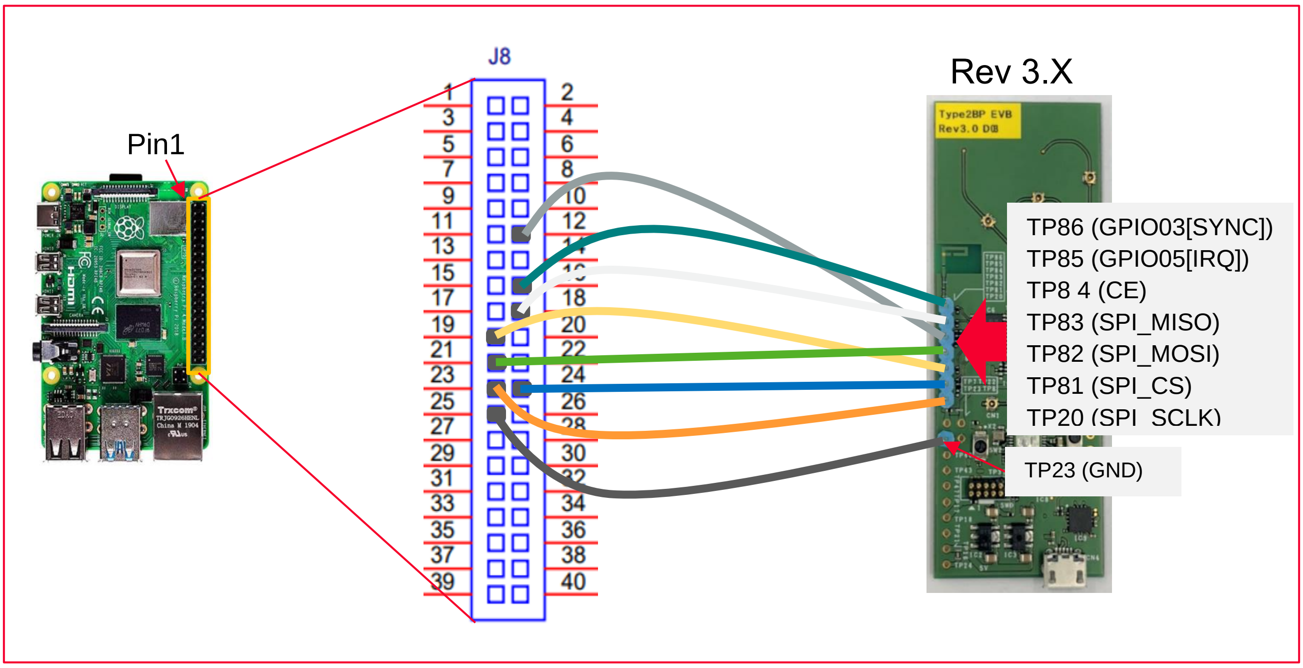


Figure 5: Pin Mapping of Raspberry Pi 4 to Type 2BP EVK (Rev 4.0 or later)

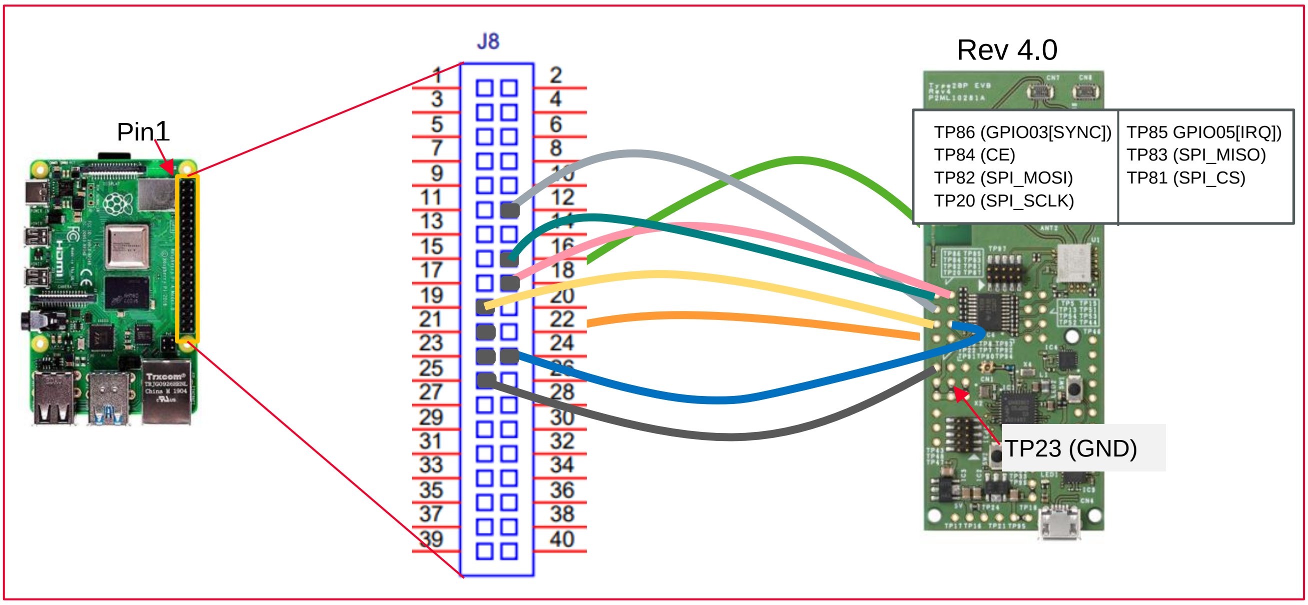


Figure 6 shows the Raspberry Pi pin locations for connection.

Figure 6: Hardware Configuration Details

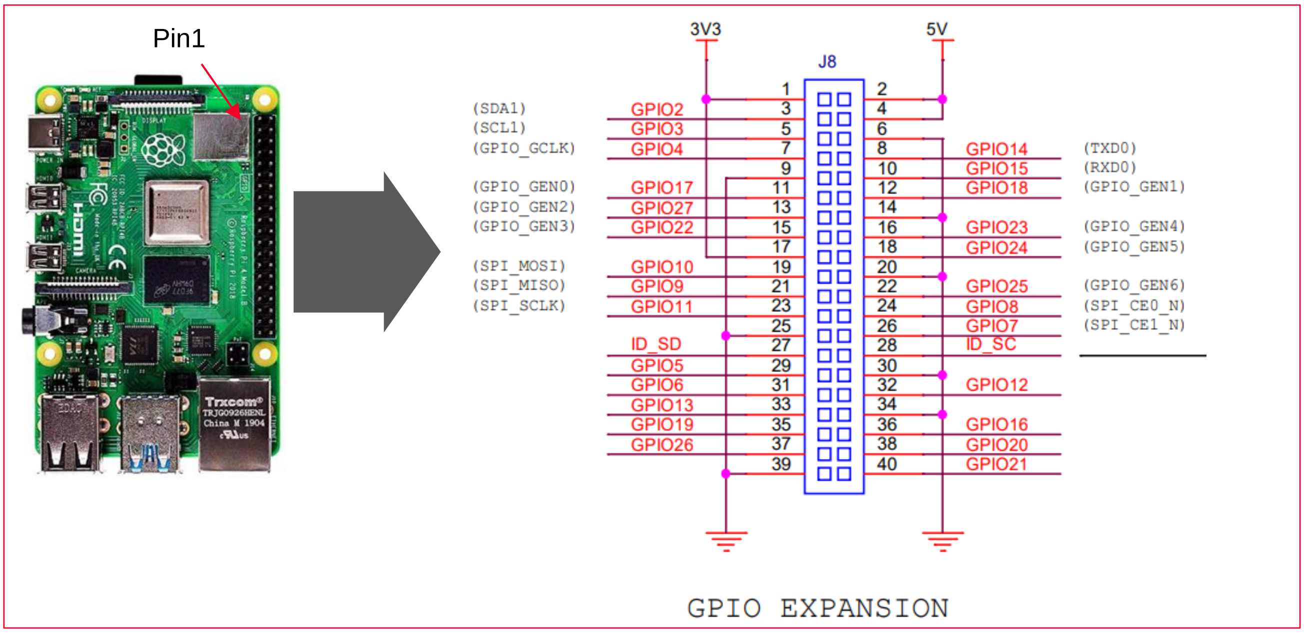


Figure 7 shows the fully connected Raspberry Pi 4 with Type 2BP EVK by wire.

Figure 7: Raspberry Pi 4 Connected to Type 2BP EVK

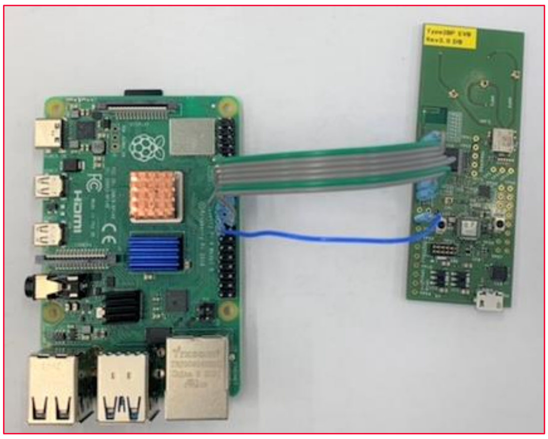


Table 2 describes the pin mapping between Type 2BP EVK and Raspberry Pi 4.

Table 2: Pin Mapping for Raspberry Pi 4 to Type 2BP EVK

Raspberry Pi 4 J8 pin No	Raspberry Pi 4 GPIO No	Type2BP EVK Pin Info
16	23	TP86 (GPIO03[SYNC])
18	24	TP85 (GPIO05[IRQ])
12	18	TP84 (CE)
21	9	TP83 (SPI_MISO)
19	10	TP82 (SPI_MOSI)
24	8	TP81 (SPI_CS)
23	11	TP20 (SPI_SCLK)
25		TP23 (GND)

3 Step 2: Setting Up Raspberry Pi 4 Linux

This procedure is based on reference document: “UWBIoT-MW-Doc.pdf” in the SDK v04.06.00 Linux package, “3.4 Raspberry PI + MK Shield Cmake Project”.

	The Raspberry Pi needs to be connected to the Internet to download the necessary files.

3.1 Prerequisites

The following needs to be done on the Raspberry Pi to set up the development environment.

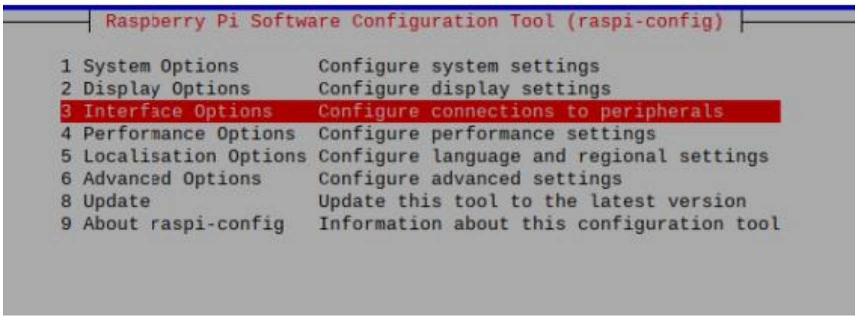
1. Enable SPI and I2C interfaces, by invoking the command below on a terminal.

```
sudo raspi-config
```

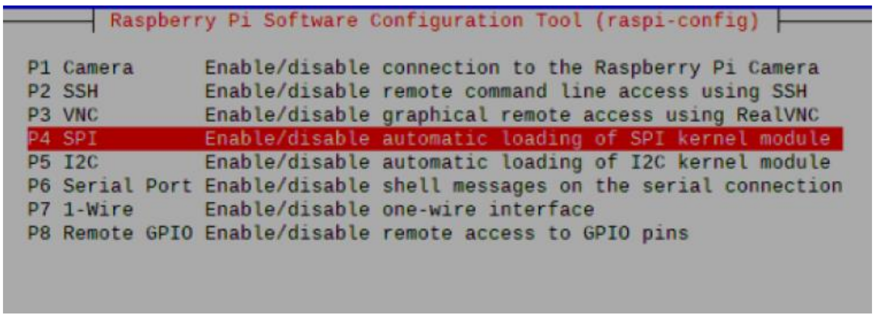
2. This will open the Raspberry Pi Software Configuration Tool. Enable SPI and I2C using the options **Interface Option** → **SPI** and **Interface Option** → **I2C** . As shown in **Figure 8**.

Figure 8: Enable Raspberry Pi 4 SPI and I2C Interfaces

raspi-config Tool



Enable SPI and Enable I2C

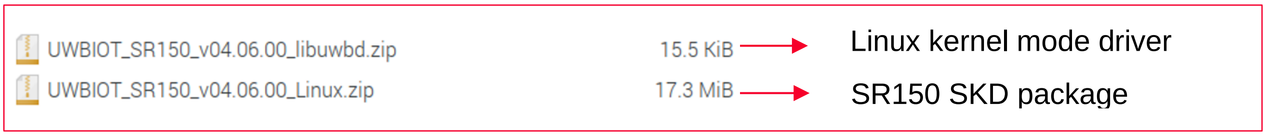


3. Install build tools, by invoking the command below

```
sudo apt update -y
sudo apt install cmake cmake-curses-gui cmake-gui libssl-dev libsystemd-
dev flex bison
```

4. Copy the Linux SDK package to Raspberry Pi. This includes both UWBIOT_SR150_v04.06.00_libuwbd.zip and UWBIOT_SR150_v04.06.00_Linux.zip

Figure 9: Copy Linux SDK Packages to Raspberry Pi 4



3.2 Build UWB Kernel Mode Driver

1. Prepare for kernel mode driver build.

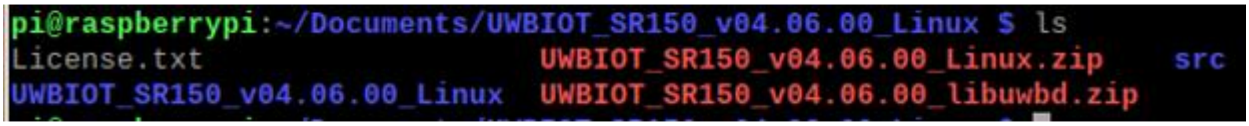
```
sudo apt install flex bison -y
sudo rpi-update

# Reboot Raspberry Pi
sudo reboot

# After reboot
sudo wget https://raw.githubusercontent.com/notro/rpi-source/master/rpi-source -O /usr/bin/rpi-source
sudo chmod +x /usr/bin/rpi-source
/usr/bin/rpi-source -q --tag-update
rpi-source

# Reboot Raspberry Pi
sudo reboot (reboot Raspberry Pi)
```

2. Unzip the UWBIOT_SR150_v04.06.00_libuwbd.zip file.

```
unzip UWBIOT_SR150_v04.06.00_libuwbd.zip
```

After unzipping, UWBIOT_SR150_v04.06.00_libuwbd.zip generates “src” folder as shown in **Figure 10**. The “src” folder is the working folder to build the kernel mode UWB driver.

Figure 10: Unzip UWBIOT_SR150_v04.06.00_libuwbd.zip File



3. Build UWB kernel mode driver.

```
cd src
make clean ; make
```

If the build is successful, sr1xxDriver.ko will be generated under “src” directory, as shown in **Figure 11**.

Figure 11: Build Kernel Mode UWB Driver

```
pi@raspberrypi:~/Documents/UWBIOT_SR150_v04.06.00_Linux $ cd src
pi@raspberrypi:~/Documents/UWBIOT_SR150_v04.06.00_Linux/src $ make clean ; make
make -C /lib/modules/5.15.26-v7l+/build M=/home/pi/Documents/UWBIOT_SR150_v04.06
.00_Linux/src clean
make[1]: ディレクトリ '/home/pi/linux-db4fcc7bd0fc08a9228a81919af21a68d38826b7'
に入ります
make[1]: ディレクトリ '/home/pi/linux-db4fcc7bd0fc08a9228a81919af21a68d38826b7'
から出ます
make -C /lib/modules/5.15.26-v7l+/build M=/home/pi/Documents/UWBIOT_SR150_v04.06
.00_Linux/src modules
make[1]: ディレクトリ '/home/pi/linux-db4fcc7bd0fc08a9228a81919af21a68d38826b7'
に入ります
  CC [M]  /home/pi/Documents/UWBIOT_SR150_v04.06.00_Linux/src/sr1xx.o
  LD [M]  /home/pi/Documents/UWBIOT_SR150_v04.06.00_Linux/src/sr1xxDriver.o
  MODPOST /home/pi/Documents/UWBIOT_SR150_v04.06.00_Linux/src/Module.symvers
  CC [M]  /home/pi/Documents/UWBIOT_SR150_v04.06.00_Linux/src/sr1xxDriver.mod.o
  LD [M]  /home/pi/Documents/UWBIOT_SR150_v04.06.00_Linux/src/sr1xxDriver.ko
make[1]: ディレクトリ '/home/pi/linux-db4fcc7bd0fc08a9228a81919af21a68d38826b7'
から出ます
pi@raspberrypi:~/Documents/UWBIOT_SR150_v04.06.00_Linux/src $ ls
Makefile        modules.order  sr1xxDriver.ko     sr1xxDriver.mod.o
Module.symvers  sr1xx.c        sr1xxDriver.mod    sr1xxDriver.o
Overlay.sh      sr1xx.o        sr1xxDriver.mod.c  uwb_overlay.dts
pi@raspberrypi:~/Documents/UWBIOT_SR150_v04.06.00_Linux/src $
```

4. Disable spidev0 module.

```
dos2unix Overlay.sh
chmod a+x Overlay.sh
./Overlay.sh
Dmesg
```

The command outputs are shown in **Figure 12**.

Figure 12: Disable spidev0 Module

```
pi@raspberrypi:~/Documents/UWBIOT_SR150_v04.06.00_Linux/src $ dos2unix Overlay.sh
dos2unix: ファイル Overlay.sh を Unix 形式へ変換しています。
pi@raspberrypi:~/Documents/UWBIOT_SR150_v04.06.00_Linux/src $ chmod a+x Overlay.sh
pi@raspberrypi:~/Documents/UWBIOT_SR150_v04.06.00_Linux/src $ ./Overlay.sh
pi@raspberrypi:~/Documents/UWBIOT_SR150_v04.06.00_Linux/src $ dmesg
[    0.000000] Booting Linux on physical CPU 0x0

[ 2010.192050] sda: detected capacity change from 0 to 499199
[ 2010.193323]  sda: sda1
[ 3124.719962] OF: overlay: WARNING: memory leak will occur if overlay removed, propert
y: /soc/spi@7e204000/spidev@0/status
[ 3124.719995] OF: overlay: WARNING: memory leak will occur if overlay removed, propert
y: /soc/spi@7e204000/status
pi@raspberrypi:~/Documents/UWBIOT_SR150_v04.06.00_Linux/src $
```

You will see warning regarding spidev0 module in dmesg log because of spidev0 module is disabled.

5. Install kernel mode driver.

```
sudo insmod sr1xxDriver.ko
dmesg
```

Figure 13: Install UWB Kernel Mode Driver

```
pi@raspberrypi:~/Documents/UWBIOT_SR150_v04.06.00_Linux/src $ sudo insmod sr1xxDriver.ko
pi@raspberrypi:~/Documents/UWBIOT_SR150_v04.06.00_Linux/src $
```

You will see [NXP-UWB] driver message as output of “dmseg” command if UWB driver is correctly installed, as shown in **Figure 14**.

Figure 14: UWB Kernel Mode Driver Installation Logs

```
[ 3337.044709] sr1xx_probe chip select : 0 , bus number = 0
[ 3337.044858] sr1xx : irq_gpio = 24, ce_gpio = 18, spi_handshake_gpio = 23
pi@raspberrypi:~/Documents/UWBIOT_SR150_v04.06.00_Linux/src $ lsmod
Module                  Size  Used by
sr1xxDriver            20480  0
sg                     28672  0
```

4 Step 3: Building the Demo Software

The steps below describe the procedure to build the demo software on the Raspberry Pi 4.

4.1 Configure Demo Binary

1. Unzip the UWBIOT_SR150_v04.06.00_Linux.zip file

```
unzip UWBIOT_SR150_v04.06.00_Linux.zip
```

After unzipping, UWBIOT_SR150_v04.06.00_Linux.zip generates “uwbiot-top” folder as shown in **Figure 15**.

Figure 15: Unzip UWBIOT_SR150_v04.06.00_Linux.zip File

```
pi@raspberrypi:~/Documents/UWBIOT_SR150_v04.06.00_Linux/UWBIOT_SR150_v04.06.00_Linux $ ls
EULA.pdf                                SCR.txt                              uwbiot-top
NXP_SR150_UCI_Specification_v2.0.7.pdf  UWBIOT_SR150_v04.06.00_SR150.pdf
NXP_UCI_CCC_Specification_v1.9.pdf      UWBIoT-MW-Doc.pdf
```

2. Applying the patch file

Download the patch file available on the v04.06.00 Linux SDK site and place it in the “UWBIOT_SR150_v04.06.00_Linux” folder.

Apply the patch file following the steps in **Figure 16**.

Figure 16: Applying patch file

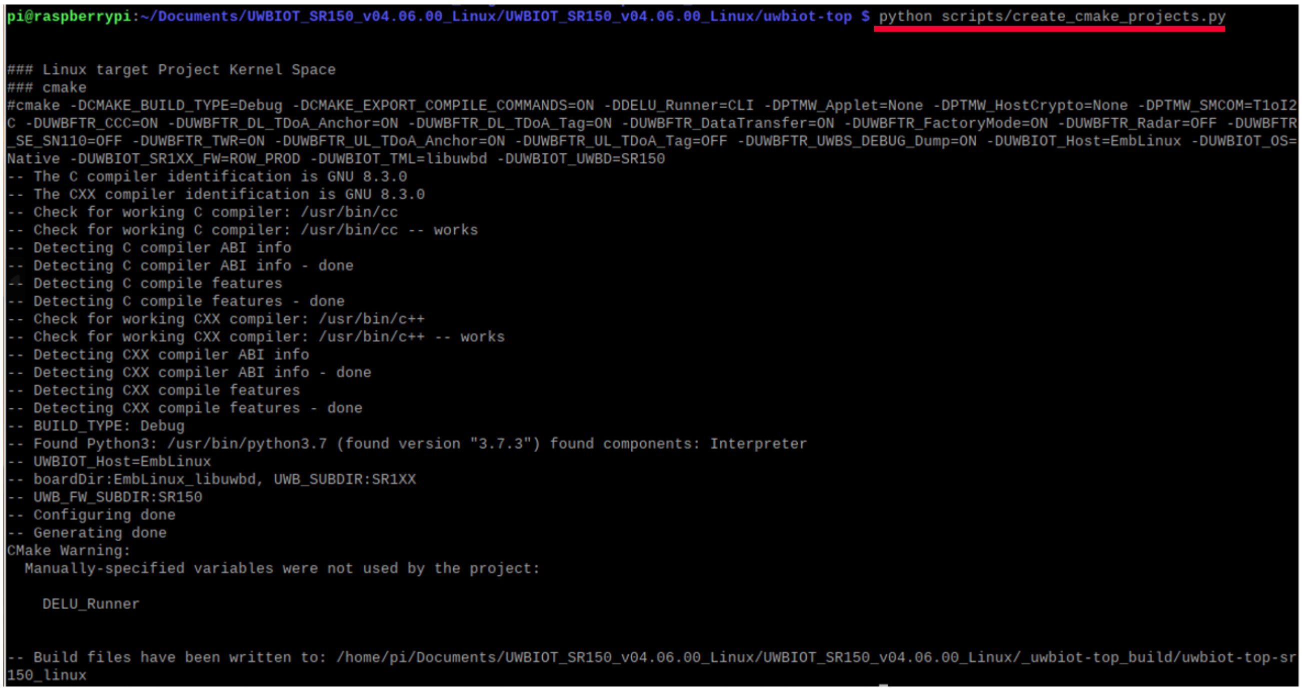
```
pi@raspberrypi:~/UWBIOT_SR150_v04.06.00_Linux $ ls
2bp_v04.06.00_linux.patch  NXP_SR150_UCI_Specification_v2.0.7.pdf  SCR.txt                           UWBIoT-MW-Doc.pdf
EULA.pdf                   NXP_UCI_CCC_Specification_v1.9.pdf      UWBIOT_SR150_v04.06.00_SR150.pdf  uwbiot-top
pi@raspberrypi:~/UWBIOT_SR150_v04.06.00_Linux $ patch -p0 < 2bp_v04.06.00_linux.patch
patching file uwbiot-top/boards/Host/EmbLinux/UWB_DeviceConfig_SR1XX.h
patching file uwbiot-top/boards/Host/EmbLinux/uwb_board.h
patching file uwbiot-top/demos/SR1XX/demo_ranging_controlee/demo_ranging_controlee.c
patching file uwbiot-top/demos/SR1XX/demo_ranging_controller/demo_ranging_controller.c
patching file uwbiot-top/demos/common/Demo_Common_Config.c
patching file uwbiot-top/libs/uwb-iot/uwb_api/PrintUtility/PrintUtility.c
patching file uwbiot-top/libs/uwb-iot/uwb_api/PrintUtility/SR1XX/PrintUtility_Proprietary.c
pi@raspberrypi:~/UWBIOT_SR150_v04.06.00_Linux $
```

3. Move to “uwbiot-top” directory and then run python script to make cmake project.

```
cd uwbiot-top
python scripts/create_cmake_projects.py
```

If the demo software build is successful, you will see "_uwbiot-top_build" directory at the same directory of "uwbiot-top". The build work area will be created at "_uwbiot-top_build/uwbiot-top-sr150_linux", as shown in **Figure 17**.

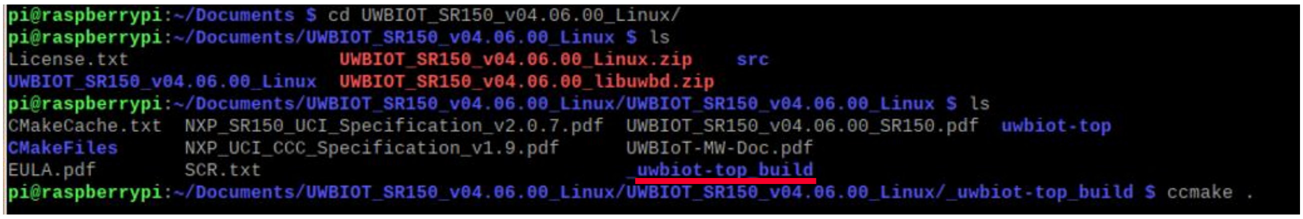
Figure 17: Build Demo Project



4. Move to the "_uwbiot-top_build/uwbiot-top-sr150_linux" directory and start the cmake configuration using "ccmake ." command, as shown in **Figure 18**.

```
cd ../_uwbiot-top_build/uwbiot-top-sr150_linux
ccmake.
```

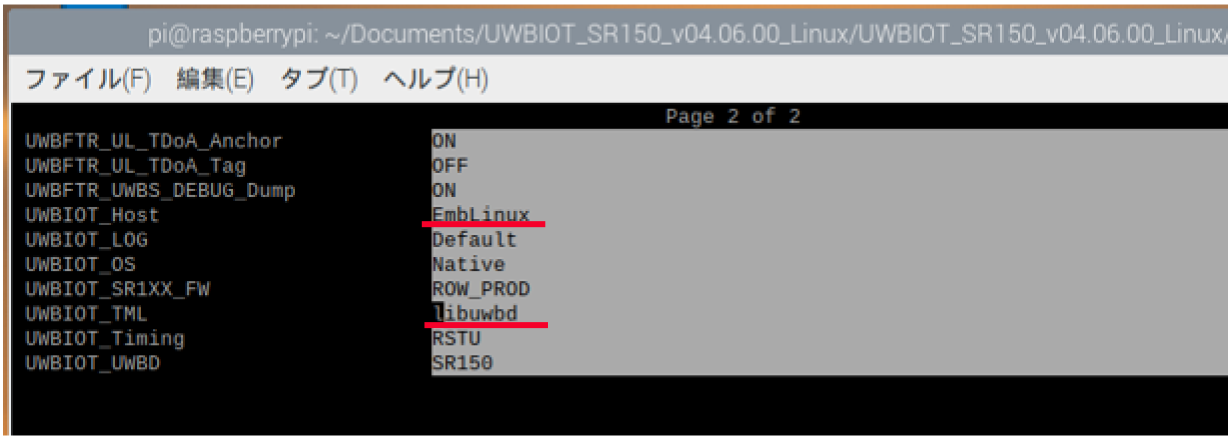
Figure 18: Start cmake Configuration



5. Set the following settings. The example setting page used in Murata environment is shown in **Figure 19**.
   - UWBIOT_HOST: EmbLinux
   - UWBIOT_TML: libuwbd

   Press [c] and [g] to make the ConfigFile.

Figure 19: cmake Settings in Murata Environment



4.2 Build Demo Binary

To build demo binary, run the appropriate make command. For example, to build the demo_ranging_controlee application, use the command below.

```
make demo_ranging_controlee
```

We recommend applying calibration data for Type 2BP EVK before building the demo application. Please refer to 4.1.2 Applying the patch file.

If the build is successful, binary file will be created at /bin directory, as shown in **Figure 20**.

Figure 20: Build Demo Binary

```
pi@raspberrypi:~/Documents/UWBIOT_SR150_v04.06.00_Linux/UWBIOT_SR150_v04.06.00_Linux/_uwbiot-top_build/uwbiot-top-sr150_linux $ make demo_ranging
_controlee
Scanning dependencies of target board_EmbLinux_libuwbd
[  1%] Building C object boards/EmbLinux_libuwbd/CMakeFiles/board_EmbLinux_libuwbd.dir/__/Host/EmbLinux/UWB_GPIOExtender.c.o
[  3%] Building C object boards/EmbLinux_libuwbd/CMakeFiles/board_EmbLinux_libuwbd.dir/__/Host/EmbLinux/boards.c.o
[  5%] Building C object boards/EmbLinux_libuwbd/CMakeFiles/board_EmbLinux_libuwbd.dir/hardware_init.c.o
[  7%] Building C object boards/EmbLinux_libuwbd/CMakeFiles/board_EmbLinux_libuwbd.dir/peripherals.c.o
[  9%] Building C object boards/EmbLinux_libuwbd/CMakeFiles/board_EmbLinux_libuwbd.dir/uwb_bus_interface.c.o
[ 10%] Building C object boards/EmbLinux_libuwbd/CMakeFiles/board_EmbLinux_libuwbd.dir/uwb_bus_io.c.o
[ 12%] Linking C static library libboard_EmbLinux_libuwbd.a
[ 12%] Built target board_EmbLinux_libuwbd
Scanning dependencies of target mwlog
[ 14%] Building C object libs/halimpl/log/CMakeFiles/mwlog.dir/nxLog.c.o
[ 16%] Linking C static library libmwlog.a
[ 16%] Built target mwlog
Scanning dependencies of target halimpl
[ 18%] Building C object libs/halimpl/CMakeFiles/halimpl.dir/fwd/SR1XX/phNxpUciHal_fwd.c.o
[ 20%] Building C object libs/halimpl/CMakeFiles/halimpl.dir/hal/phNxpUciHal.c.o
[ 21%] Building C object libs/halimpl/CMakeFiles/halimpl.dir/hal/phNxpUciHal_ext.c.o
[ 23%] Building C object libs/halimpl/CMakeFiles/halimpl.dir/log/nxLog.c.o
[ 25%] Building C object libs/halimpl/CMakeFiles/halimpl.dir/tml/phTmlUwb.c.o
[ 27%] Building C object libs/halimpl/CMakeFiles/halimpl.dir/tml/phTmlUwb_transport.c.o
[ 29%] Building C object libs/halimpl/CMakeFiles/halimpl.dir/utils/phNxpUciHal_utils.c.o
[ 30%] Building C object libs/halimpl/CMakeFiles/halimpl.dir/utils/phNxpUwbConfig.c.o
[ 32%] Building C object libs/halimpl/CMakeFiles/halimpl.dir/utils/utlv.c.o
[ 34%] Building C object libs/halimpl/CMakeFiles/halimpl.dir/transport/libuwbd/SR1XX/uwb_uwbs_tml_interface.c.o
[ 36%] Building C object libs/halimpl/CMakeFiles/halimpl.dir/osal/phOsalUwb_Queue_posix.c.o
[ 38%] Building C object libs/halimpl/CMakeFiles/halimpl.dir/osal/phOsalUwb_Thread_posix.c.o
[ 38%] Building C object libs/halimpl/CMakeFiles/halimpl.dir/osal/phOsalUwb_Timer_posix.c.o
[ 40%] Building C object libs/halimpl/CMakeFiles/halimpl.dir/osal/phOsalUwb_posix.c.o
[ 41%] Building C object libs/halimpl/CMakeFiles/halimpl.dir/fwdl_provider/uwb_fwdl_ext.c.o
[ 43%] Building C object libs/halimpl/CMakeFiles/halimpl.dir/fwdl_provider/uwb_fwdl_ram.c.o
[ 45%] Linking C static library libhalimpl.a
[ 45%] Built target halimpl
Scanning dependencies of target uwb_core
```

```
[ 47%] Building C object libs/uwb-iot/uwb_core/CMakeFiles/uwb_core.dir/adaptation/UwbAdaptation.c.o
[ 49%] Linking C static library libuwb_core.a
[ 49%] Built target uwb_core
Scanning dependencies of target uci_core
[ 50%] Building C object libs/uci-core/CMakeFiles/uci_core.dir/src/uci_hmsgs.c.o
[ 52%] Building C object libs/uci-core/CMakeFiles/uci_core.dir/src/uwa_dm_act.c.o
[ 54%] Building C object libs/uci-core/CMakeFiles/uci_core.dir/src/uwa_dm_api.c.o
[ 56%] Building C object libs/uci-core/CMakeFiles/uci_core.dir/src/uwa_dm_main.c.o
[ 58%] Building C object libs/uci-core/CMakeFiles/uci_core.dir/src/uwa_sys_main.c.o
[ 60%] Building C object libs/uci-core/CMakeFiles/uci_core.dir/src/uwb_main.c.o
[ 60%] Building C object libs/uci-core/CMakeFiles/uci_core.dir/src/uwb_task.c.o
[ 61%] Building C object libs/uci-core/CMakeFiles/uci_core.dir/src/uwb_ucif.c.o
[ 63%] Linking C static library libuci_core.a
[ 63%] Built target uci_core
Scanning dependencies of target uwb_SR150
[ 65%] Building C object libs/uwb-iot/uwb_api/CMakeFiles/uwb_SR150.dir/Api/SR1XX/AppConfigParams.c.o
[ 67%] Building C object libs/uwb-iot/uwb_api/CMakeFiles/uwb_SR150.dir/Api/SR1XX/UwbApi_Proprietary.c.o
[ 69%] Building C object libs/uwb-iot/uwb_api/CMakeFiles/uwb_SR150.dir/Api/SR1XX/UwbApi_Proprietary_Fm.c.o
[ 70%] Building C object libs/uwb-iot/uwb_api/CMakeFiles/uwb_SR150.dir/Api/SR1XX/UwbApi_Proprietary_Internal.c.o
[ 72%] Building C object libs/uwb-iot/uwb_api/CMakeFiles/uwb_SR150.dir/Api/SR1XX/UwbApi_RfTest.c.o
[ 74%] Building C object libs/uwb-iot/uwb_api/CMakeFiles/uwb_SR150.dir/Api/UwbApi.c.o
[ 76%] Building C object libs/uwb-iot/uwb_api/CMakeFiles/uwb_SR150.dir/Api/UwbApi_Internal.c.o
[ 78%] Building C object libs/uwb-iot/uwb_api/CMakeFiles/uwb_SR150.dir/Api/UwbApi_Utility.c.o
[ 80%] Building C object libs/uwb-iot/uwb_api/CMakeFiles/uwb_SR150.dir/PrintUtility/PrintUtility.c.o
[ 81%] Building C object libs/uwb-iot/uwb_api/CMakeFiles/uwb_SR150.dir/PrintUtility/SR1XX/PrintUtility_Proprietary.c.o
[ 83%] Building C object libs/uwb-iot/uwb_api/CMakeFiles/uwb_SR150.dir/PrintUtility/PrintUtility_RfTest.c.o
[ 85%] Linking C shared library libuwb_SR150.so
[ 85%] Built target uwb_SR150
Scanning dependencies of target app_common
[ 85%] Building C object demos/common/CMakeFiles/app_common.dir/Standalone_Main_EmbLinux.c.o
[ 87%] Building C object demos/common/CMakeFiles/app_common.dir/Demo_Common_Config.c.o
[ 89%] Building C object demos/common/CMakeFiles/app_common.dir/AppCallbacks.c.o
[ 90%] Building C object demos/common/CMakeFiles/app_common.dir/AppRecovery.c.o
[ 92%] Building C object demos/common/CMakeFiles/app_common.dir/AppStateManagement.c.o
[ 94%] Building C object demos/common/CMakeFiles/app_common.dir/Utilities.c.o
[ 96%] Linking C static library libapp_common.a
[ 96%] Built target app_common
Scanning dependencies of target demo_ranging_controlee
[ 98%] Building C object demos/SR1XX/demo_ranging_controlee/CMakeFiles/demo_ranging_controlee.dir/demo_ranging_controlee.c.o
[100%] Linking C executable ../../../bin/demo_ranging_controlee
[100%] Built target demo_ranging_controlee
```

5 Step 3: Running Demo Application

To run the demo application, enter the following command, as shown in **Figure 21**.

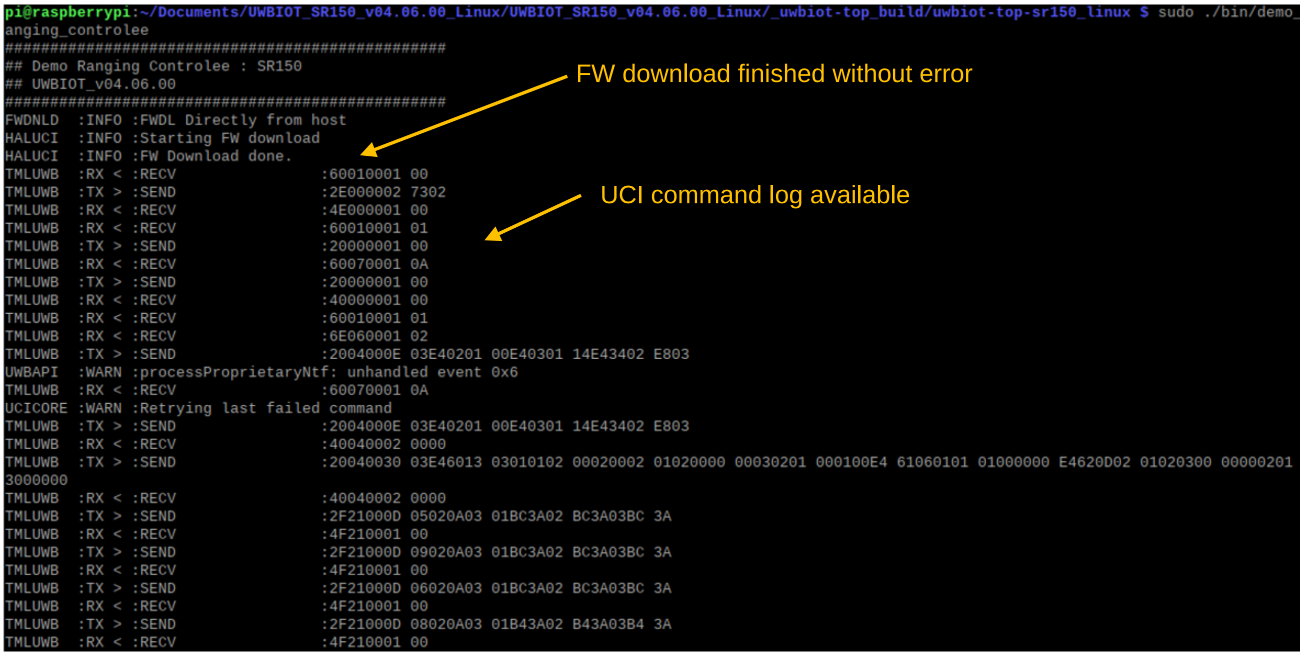
```
sudo ./bin/demo_ranging_controlee
```

Figure 21: Run Demo Application

```
pi@raspberrypi:~/Documents/UWBIOT_SR150_v04.06.00_Linux/UWBIOT_SR150_v04.06.00_Linux/_uwbiot-top_build/uwbiot-top-sr150_linux $ ls
CMakeCache.txt  Makefile          _cmake_createnew.sh  bin     cmake_install.cmake    demos  uwb_iot_ftr.h
CMakeFiles      _cmake_build.sh   _cmake_editcache.sh  boards  compile_commands.json  libs
pi@raspberrypi:~/Documents/UWBIOT_SR150_v04.06.00_Linux/UWBIOT_SR150_v04.06.00_Linux/_uwbiot-top_build/uwbiot-top-sr150_linux $ sudo ./bin/demo_r
anging_controlee
################################################
## Demo Ranging Controlee : SR150
## UWBIOT_v04.06.00
################################################
FWDNLD  :INFO :FWDL Directly from host
HALUCI  :INFO :Starting FW download
HALUCI  :INFO :FW Download done.
```

On successful run, UCI command logs will be seen in the console, as shown in **Figure 22**.

Figure 22: Demo Application Run Success




```
pi@raspberrypi:~/Documents/UWBIOT_SR150_v04.06.00_Linux/UWBIOT_SR150_v04.06.00_Linux/_uwbiot-top_build/uwbiot-top-sr150_linux $ sudo ./bin/demo_
anging_controlee
################################################
## Demo Ranging Controlee : SR150
## UWBIOT_v04.06.00
################################################
FWDNLD  :INFO :FWDL Directly from host
HALUCI  :INFO :Starting FW download
HALUCI  :INFO :FW Download done.
TMLUWB  :RX < :RECV                :60010001 00
TMLUWB  :TX > :SEND                :2E000002 7302
TMLUWB  :RX < :RECV                :4E000001 00
TMLUWB  :RX < :RECV                :60010001 01
TMLUWB  :TX > :SEND                :20000001 00
TMLUWB  :RX < :RECV                :60070001 0A
TMLUWB  :TX > :SEND                :20000001 00
TMLUWB  :RX < :RECV                :40000001 00
TMLUWB  :RX < :RECV                :60010001 01
TMLUWB  :RX < :RECV                :6E060001 02
TMLUWB  :TX > :SEND                :2004000E 03E40201 00E40301 14E43402 E803
UWBAPI  :WARN :processProprietaryNtf: unhandled event 0x6
TMLUWB  :RX < :RECV                :60070001 0A
UCICORE :WARN :Retrying last failed command
TMLUWB  :TX > :SEND                :2004000E 03E40201 00E40301 14E43402 E803
TMLUWB  :RX < :RECV                :40040002 0000
TMLUWB  :TX > :SEND                :20040030 03E46013 03010102 00020002 01020000 00030201 000100E4 61060101 01000000 E4620D02 01020300 00000201
3000000
TMLUWB  :RX < :RECV                :40040002 0000
TMLUWB  :TX > :SEND                :2F21000D 05020A03 01BC3A02 BC3A03BC 3A
TMLUWB  :RX < :RECV                :4F210001 00
TMLUWB  :TX > :SEND                :2F21000D 09020A03 01BC3A02 BC3A03BC 3A
TMLUWB  :RX < :RECV                :4F210001 00
TMLUWB  :TX > :SEND                :2F21000D 06020A03 01BC3A02 BC3A03BC 3A
TMLUWB  :RX < :RECV                :4F210001 00
TMLUWB  :TX > :SEND                :2F21000D 08020A03 01B43A02 B43A03B4 3A
TMLUWB  :RX < :RECV                :4F210001 00
```

If you see failure at FW download (as shown in **Figure 23**), please check if USB connector is plugged to Type 2BP EVK. SR150 needs power supply from USB. If you see failure even with power supplied correctly, please also check that the SPI H/W connection is correct. Also, try to update “wiring pi” version to 2.52 or later.

Figure 23: Demo Application Run Failure

```
pi@raspberrypi:~/Documents/UWBIOT_SR150_v04.06.00_Linux/UWBIOT_SR150_v04.06.00_Linux/_uwbiot-top_build/uwbiot-top-sr150_linux $ sudo ./bin/demo_r
anging_controlee
#################################################
## Demo Ranging Controlee : SR150
## UWBIOT_v04.06.00
#################################################
FWDNLD  :INFO :FWDL Directly from host
HALUCI  :INFO :Starting FW download
TMLUWB  :WARN :uwb_bus_data_rx failed
TMLUWB  :ERROR:uwb_bus_data_rx failed
TMLUWB  :ERROR:uwb_uwbs_tml_data_trx : uwb_uwbs_tml_data_rx failed
FWDNLD  :ERROR:Failure!
HALUCI  :ERROR:FW download is failed: status= 1
UWBAPI  :ERROR:uwbInit failed
APP     :ERROR:UwbApi_Init_New Failed
APP     :ERROR:UwbApi_ShutDown Failed
APP     :ERROR:
Finished /home/pi/Documents/UWBIOT_SR150_v04.06.00_Linux/UWBIOT_SR150_v04.06.00_Linux/uwbiot-top/demos/SR1XX/demo_ranging_controlee/demo_ranging_
controlee.c : Failed!
```

FW download is failed

5.1 Connect with “Controller”

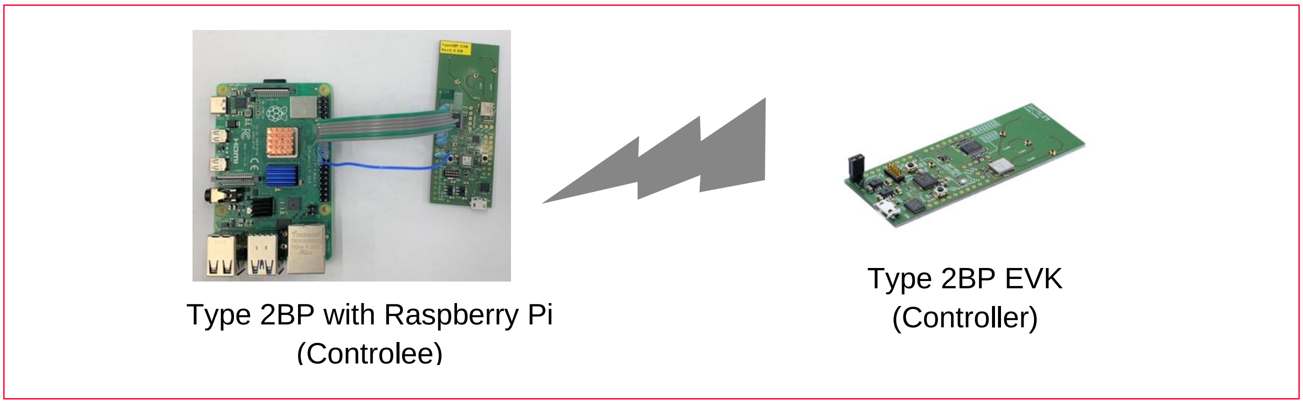
During Type 2BP working on Raspberry pi as “Controlee”, if you turn on “Controller (standalone binary with v04.06.00)”, UWB connection will be established automatically. And you will see the logs shown in **Figure 24** if the UWB connection is successful.

Figure 24: UWB Connection Success

```
APP      :INFO :TWR[0].aoa_azimuth: -52.52
APP      :INFO :TWR[0].aoa_elevation       : -60.0
TMLUWB   :RX < :RECV                :6200005C 47000000 01000000 00C80000 00010000 00000000 00000000 01111100 011D0020
E66400E2 64000000 00000005 00000000 00000000 00000000 22000000 01020102 00020203 20E6943C 3F0300E2 92D03F03 222218D0
00D12020 24D040D0
APP      :INFO :TWR[0].nLos             : 1
APP      :INFO :TWR[0].distance         : 29
APP      :INFO :TWR[0].aoa_azimuth: -52.32
APP      :INFO :TWR[0].aoa_elevation       : -60.0
TMLUWB   :RX < :RECV                :6200005C 48000000 01000000 00C80000 00010000 00000000 00000000 01111100 002200AC
E66400E2 64000000 00000005 00000000 00000000 00000000 22000000 01020102 00020203 ACE61A3B 410300E2 22CD4103 222230D0
40D02121 3AD040D0
APP      :INFO :TWR[0].nLos             : 0
APP      :INFO :TWR[0].distance         : 34
APP      :INFO :TWR[0].aoa_azimuth: -51.44
APP      :INFO :TWR[0].aoa_elevation       : -60.0
TMLUWB   :RX < :RECV                :6200005C 49000000 01000000 00C80000 00010000 00000000 00000000 01111100 00250000
E26421EC 64000000 00000005 00000000 00000000 00000000 22000000 01020102 00020203 00E26E3C 410321EC E8E44103 2222F5CF
00D02020 FDCF00D0
APP      :INFO :TWR[0].nLos             : 0
APP      :INFO :TWR[0].distance         : 37
APP      :INFO :TWR[0].aoa_azimuth: -60.0
APP      :INFO :TWR[0].aoa_elevation       : -40.33
```

The connection setup of Type 2BP EVK (Controller) and Raspberry Pi with Type 2BP EVK (Controlee) is shown in **Figure 25**.

Figure 25: Connect with Controller Type 2BP with Raspberry Pi with Type 2BP EVK



6 Run Demo Application After Raspberry Pi Reboot

After the Raspberry pi reboots, you need to repeat the steps from “disable pcidev0” (Step 4 of Section 3.2) to run the demo application again, as shown below.

```
cd src
./Overlay.sh
sudo insmod sr1xxDriver.ko
cd ../_uwbiot-top_build/uwbiot-top-sr150_linux
sudo ./bin/demo_ranging_controlee
```

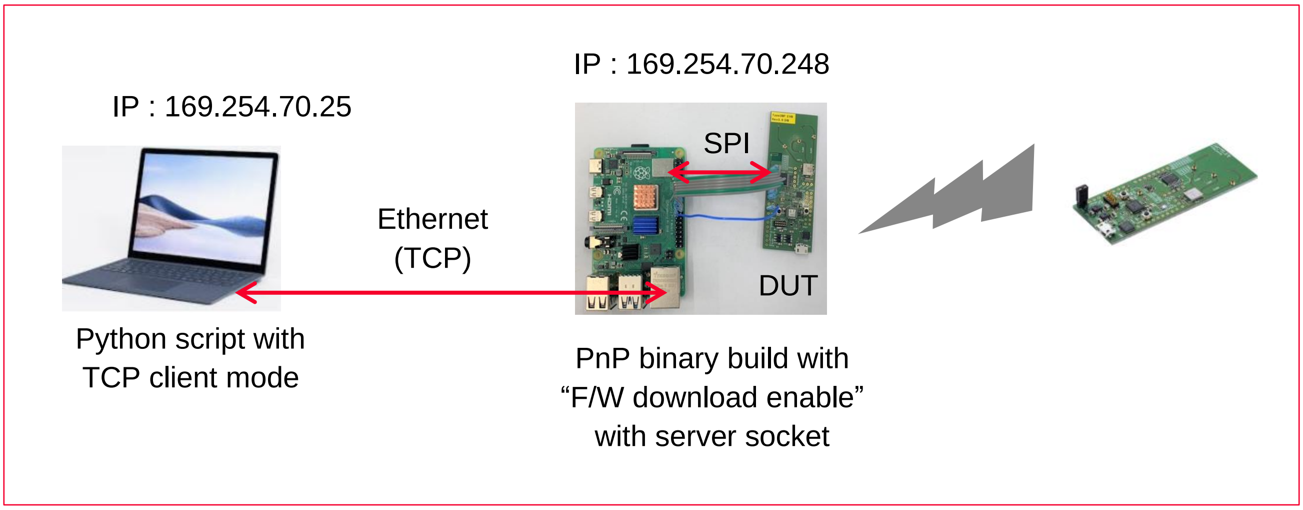
7 Appendix: Running PnP Through Ethernet

This appendix describes the process of running PnP script on a Windows PC and controlling the Type 2BP EVK (connected with the Raspberry Pi 4) via TCP over Ethernet.

7.1 Controlling DUT Using Raspberry Pi Through PC

The demo setup is shown in **Figure 26**. To run the PnP python script in TCP client mode on the Windows PC, there is demo program to send Python command (UCI command) through TCP.

Figure 26: Setup for PnP Through Ethernet Example



7.2 Build PnP Binary

Please follow the instructions in Section 3 to set up the Raspberry Pi, if not already done. This covers the kernel mode driver build (UWBIOT_SR150_v04.06.00_libuwbd).

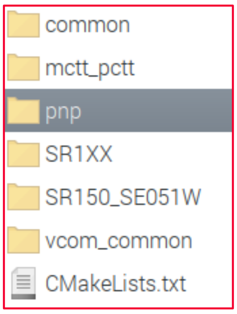
The PnP demo application is included in the UWBIOT_SR150_v04.06.00_Linux.zip file.

1. Unzip the UWBIOT_SR150_v04.06.00_Linux.zip file

```
unzip UWBIOT_SR150_v04.06.00_Linux.zip
```

2. After unzipping, UWBIOT_SR150_v04.06.00_Linux.zip generates “uwbiot-top” folder.
3. Move to “uwbiot-top/demos” directory. This will contain the pnp application that needs to be built, as shown in **Figure 27**.

Figure 27: PnP Application Directory



4. With “pnp” application default build setting, FW of SR150 is not downloaded automatically. (The default build setting is for the system which downloads F/W from external system, i.e Litepoint tester).

   To control DUT from Windows PC/Python script through TCP connection, please change the build configuration setting in UWBIOT_SR150_v04.06.00_Linux/uwbiot-top/libs/halimpl/inc/phUwb_BuildConfig.h file, as shown in **Figure 28**, to enable FW download.

Figure 28: Change Build Config Setting

```
#define INTERNAL_FIRMWARE_DOWNLOAD DISABLED
/* doc:end:enable-int-fw-download */
```

```
#define INTERNAL_FIRMWARE_DOWNLOAD ENABLED
/* doc:end:enable-int-fw-download */
```

5. Build the demo binary.

```
make pnp_EmbLinux_SR150
```

6. If the build is successful, pnp_Emblinux_SR150.bin will be created. Sample console output is shown in **Figure 29**.

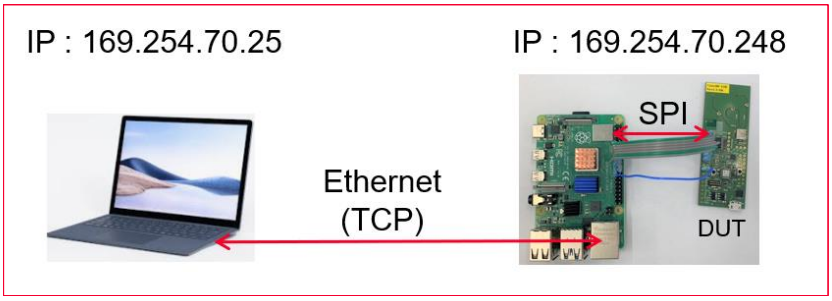
Figure 29: Build PnP Demo Binary

```
pi@raspberrypi:~/Documents/UWBIOT_SR150_v04.06.00_Linux/UWBIOT_SR150_v04.06.00_Linux/_uwbiot-top_build/uwbiot-top-sr150_linux $ make clean
pi@raspberrypi:~/Documents/UWBIOT_SR150_v04.06.00_Linux/UWBIOT_SR150_v04.06.00_Linux/_uwbiot-top_build/uwbiot-top-sr150_linux $ make pnp_EmbLinux_SR150
[  1%] Building C object boards/EmbLinux_libuwbd/CMakeFiles/board_EmbLinux_libuwbd.dir/__/Host/EmbLinux/UWB_GPIOExtender.c.o
[  3%] Building C object boards/EmbLinux_libuwbd/CMakeFiles/board_EmbLinux_libuwbd.dir/__/Host/EmbLinux/boards.c.o
[  4%] Building C object boards/EmbLinux_libuwbd/CMakeFiles/board_EmbLinux_libuwbd.dir/hardware_init.c.o
[  6%] Building C object boards/EmbLinux_libuwbd/CMakeFiles/board_EmbLinux_libuwbd.dir/peripherals.c.o
[  8%] Building C object boards/EmbLinux_libuwbd/CMakeFiles/board_EmbLinux_libuwbd.dir/uwb_bus_interface.c.o
[  9%] Building C object boards/EmbLinux_libuwbd/CMakeFiles/board_EmbLinux_libuwbd.dir/uwb_bus_io.c.o
[ 11%] Linking C static library libboard_EmbLinux_libuwbd.a
[ 11%] Built target board_EmbLinux_libuwbd
[ 13%] Building C object libs/halimpl/log/CMakeFiles/mwlog.dir/nxLog.c.o
[ 14%] Linking C static library libmwlog.a
[ 14%] Built target mwlog

[ 90%] Building C object demos/pnp/CMakeFiles/pnp_EmbLinux_SR150.dir/Socket/Uwb_Srv_Pnp.c.o
[ 91%] Building C object demos/pnp/CMakeFiles/pnp_EmbLinux_SR150.dir/UWBPnP_Common.c.o
[ 93%] Building C object demos/pnp/CMakeFiles/pnp_EmbLinux_SR150.dir/UWB_Hbci.c.o
[ 95%] Building C object demos/pnp/CMakeFiles/pnp_EmbLinux_SR150.dir/UWB_Spi_Driver_Interface_Sr1xx_Sr2xx.c.o
[ 96%] Building C object demos/pnp/CMakeFiles/pnp_EmbLinux_SR150.dir/Uwb_Hif_Handler_posix.c.o
[ 98%] Building C object demos/pnp/CMakeFiles/pnp_EmbLinux_SR150.dir/Uwb_Read_task_posix.c.o
[100%] Linking C executable ../../bin/pnp_EmbLinux_SR150
[100%] Built target pnp_EmbLinux_SR150
pi@raspberrypi:~/Documents/UWBIOT_SR150_v04.06.00_Linux/UWBIOT_SR150_v04.06.00_Linux/_uwbiot-top_build/uwbiot-top-sr150_linux $
```

7.3 Prepare PnP Script on PC Side

The PnP python script needs to be modified to send command over TCP. The demo network environment is shown in **Figure 30**.

Figure 30: Example of Network Environment from Client Site



In order to communicate over TCP, the network address of PC and Raspberry Pi board must match.

1. Confirm the IP address of Raspberry Pi board, as shown in **Figure 31**.

Figure 31: Confirm the IP address of Raspberry Pi Board

```
pi@raspberrypi:~/Documents/UWBIOT_SR150_v04.06.00_Linux/UWBIOT_SR150_v04.06.00_Linux/_uwbiot-top_build/uwbiot-top-sr150_linux $ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 169.254.70.248  netmask 255.255.0.0  broadcast 169.254.255.255
        inet6 fe80::23bd:f29:67dd:f38f  prefixlen 64  scopeid 0x20<link>
        ether dc:a6:32:f4:f2:7e  txqueuelen 1000  (イーサネット)
        RX packets 61  bytes 6048 (5.9 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 35  bytes 5584 (5.4 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (ローカルループバック)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

pi@raspberrypi:~/Documents/UWBIOT_SR150_v04.06.00_Linux/UWBIOT_SR150_v04.06.00_Linux/_uwbiot-top_build/uwbiot-top-sr150_linux $ ping 169.254.70.25
PING 169.254.70.25 (169.254.70.25) 56(84) bytes of data.
64 bytes from 169.254.70.25: icmp_seq=1 ttl=128 time=0.836 ms
```

2. Set up the correct IP address of Raspberry pi in "MTD-SCP-045-A_Type2BP_EVK_with_RaspberryPi4_serial2socket.py" script, as shown in **Figure 32**.

Figure 32: Update Script to Use Correct IP Address of Raspberry Pi

```python
import socket

HOST = '169.254.70.248' #'192.168.1.42'
PORT = 3001
```

3. Modify the "MTD_SCP_102_A_DS_TWR_SR150_UART_interface_v040600.py" python script to use transfer serial port commands through TCP, as shown in **Figure 33**.

Figure 33: Update Script to Transfer Serial Port Communication Through TCP

```
import numpy as np
import matplotlib.pyplot as plt
import numpy as np
import os
import queue
import serial
from serial2socket import serial
import signal
import sys
```

Python scripts for v04.06.00 are available on MyMurata.

UWBIOT SR150 v04.06.00 MCUx site

- MTD-SCP-102-A_DS-TWR_SR150_Unicast_v040600.py
- MTD_SCP_102_A_DS_TWR_SR150_UART_interface_v040600.py

Type2BP Document Site (Software Guide)

- MTD-SCP-045-A_Type2BP_EVK_with_RaspberryPi4_serial2socket.py

Once the above steps are completed, all the commands for serial port are transferred through TCP.

7.4 Execute the Program

Execute PnP socket application on Raspberry Pi, using the commands listed below.

```
cd src
./Overlay.sh
sudo insmod sr1xxDriver.ko
cd ../_uwbiot-top_build/uwbiot-top-sr150_linux
sudo ./bin/pnp_EmbLinux_SR150
```

After the boot up, the PnP binary is downloaded to SR150 and will start waiting for serial command over TCP (port 3001).

The log of the PnP socket application, “pnp_Emblinux_SR150.bin” is shown below in **Figure 34**.

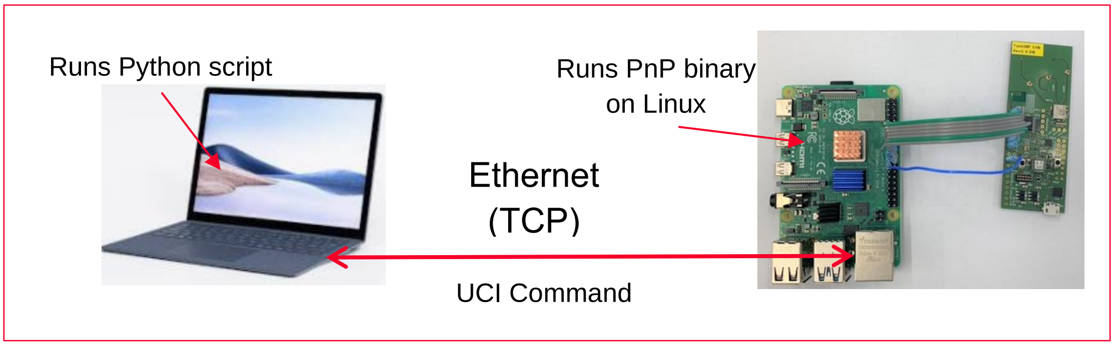
Figure 34: PnP Socket Application Output

```
pi@raspberrypi:~/Documents/UWBIOT_SR150_v04.06.00_Linux/UWBIOT_SR150_v04.06.00_Linux/_uwbiot-top_build/uwbiot-top-sr1
50_linux $ sudo ./bin/pnp_EmbLinux_SR150
#################################################
## Demo PNP Socket Server : SR150
## UWBIOT_v04.06.00
#################################################
APP     :INFO :UWB PNP Major Version: 0x4
APP     :INFO :UWB PNP Minor Version: 0x6
APP     :INFO :main(): Helios initialized

HELIOS FW download completed
```

You will see “HELIOS FW downloaded completed” with “FW Download enable” build setting in this log.

The demo setup with respect to the software, for the PC to Raspberry Pi connection, is shown in **Figure 35** below.

Figure 35: Demo Hardware Setup for Communication Over Ethernet (TCP)



On the PC side, now execute the python script with a BAT file. The log message generated on the console are shown in **Figure 36**. Then you will see the same log of you execute Python script of connection with serial port. Please refer to the PnP Test Guide for instructions on how to operate in PnP mode.

Figure 36: Execute Python Script on PC Side

```
C:¥NXP¥UWBIOT_SR150_v04.06.00_Linux¥PnP_RaspberryPi>py -m MTD-SCP-102-A_DS-TWR_SR150_Unicast_v040600
##########################################################################
File Name: MTD-SCP-102-A_DS-TWR_SR150_Unicast_v040600.py
Date Time: 2024-05-23 13:21:22.480695
##########################################################################
Role:Initiator   Port:COM14   Nb Meas:0   Timestamp:False   Range Plot:False
Configure serial port...
Serial port configured
Start adding commands to the queue...
adding commands to the queue completed
Start processing...
Read from serial port started
Write to serial port started
NXPUCIX => 2E 00 00 02 73 04
NXPUCIR <= 60 07 00 01 0A
NXPUCIX => 2E 00 00 02 73 04
NXPUCIR <= 4E 00 00 01 00
```

The “serial2socket.py” must be in the same folder of script.

On the Raspberry Pi side, the log message will now show “TCP/IP client Connected!” message and start showing Rx/Tx messages, as shown in **Figure 37**.

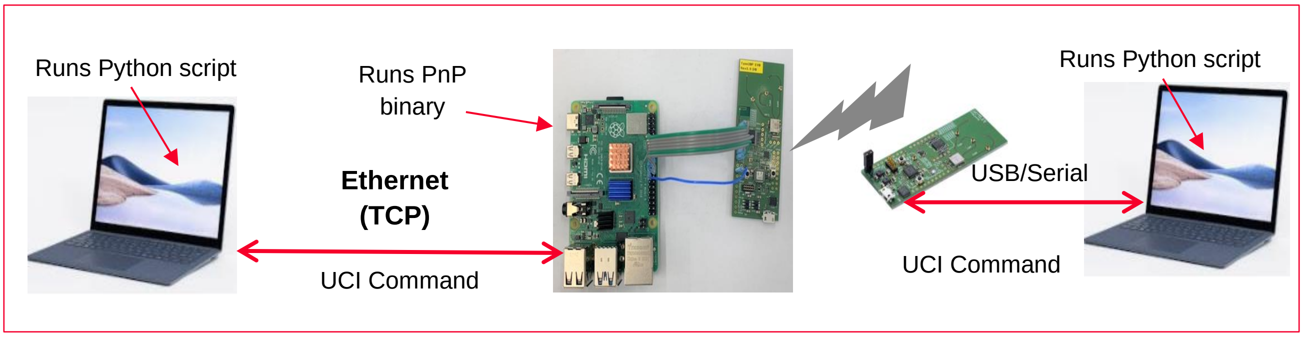
Figure 37: Raspberry Pi Log Message

```
####################################################
## Demo PNP Socket Server : SR150
## UWBIOT_v04.06.00
####################################################
APP     :INFO :UWB PNP Major Version: 0x4
APP     :INFO :UWB PNP Minor Version: 0x6
APP     :INFO :main(): Helios initialized

HELIOS FW download completed
APP     :INFO :received uci rsp/ntf: 0x60 0x1 0x0 0x1
APP     :INFO :TCP/IP client Connected!

APP     :INFO :received uci rsp/ntf: 0x4E 0x0 0x0 0x1
APP     :INFO :received uci rsp/ntf: 0x60 0x1 0x0 0x1
APP     :INFO :received uci rsp/ntf: 0x60 0x7 0x0 0x1
APP     :INFO :received uci rsp/ntf: 0x40 0x0 0x0 0x1
APP     :INFO :received uci rsp/ntf: 0x60 0x1 0x0 0x1
APP     :INFO :received uci rsp/ntf: 0x6E 0x6 0x0 0x1
APP     :INFO :received uci rsp/ntf: 0x60 0x7 0x0 0x1
APP     :INFO :received uci rsp/ntf: 0x40 0x4 0x0 0x2
APP     :INFO :received uci rsp/ntf: 0x4E 0x2 0x0 0xE8
APP     :INFO :received uci rsp/ntf: 0x40 0x2 0x0 0x5C
```

Please follow the steps in the PnP Test Guide to start the operation of the opposite device's Type2BP-EVK.

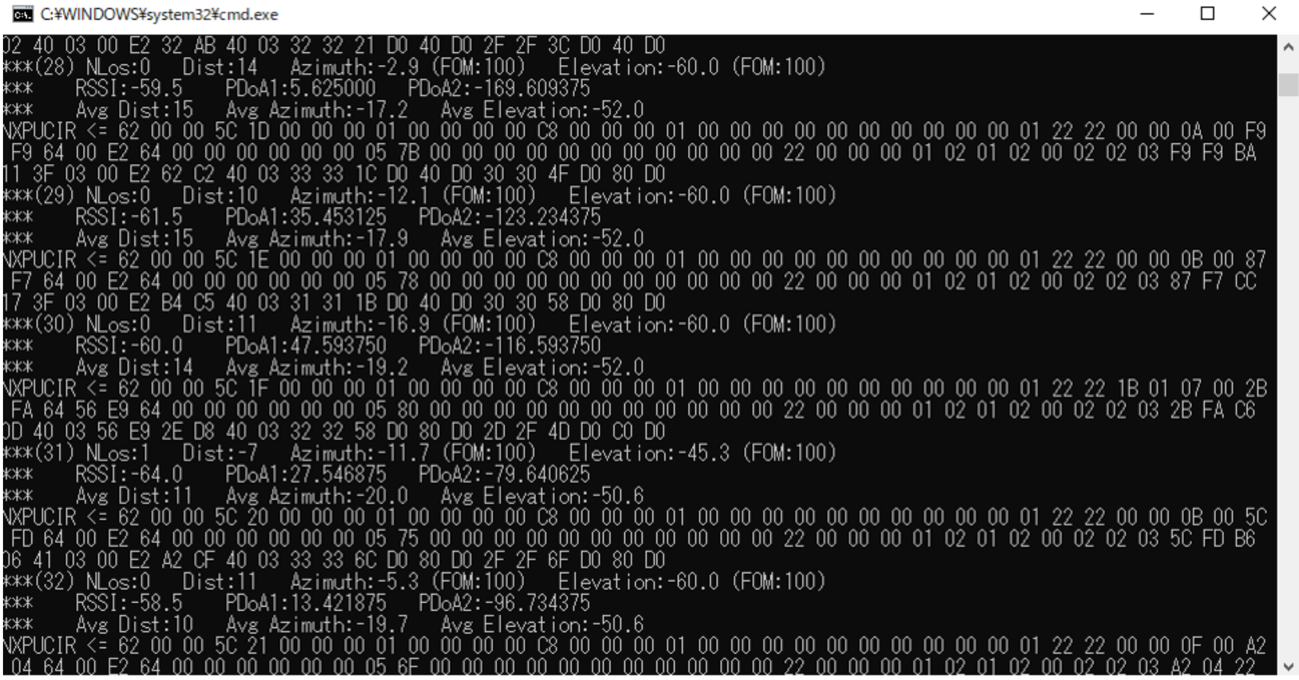
The complete demo setup with respect to the software, to run the ranging test, is shown in **Figure 38** below.
Executing the script on the opposite device will initiate ranging with the Type2BP on the Raspberry Pi side.

Figure 38: Demo Hardware Setup for Ranging with Communication Over Ethernet (TCP)



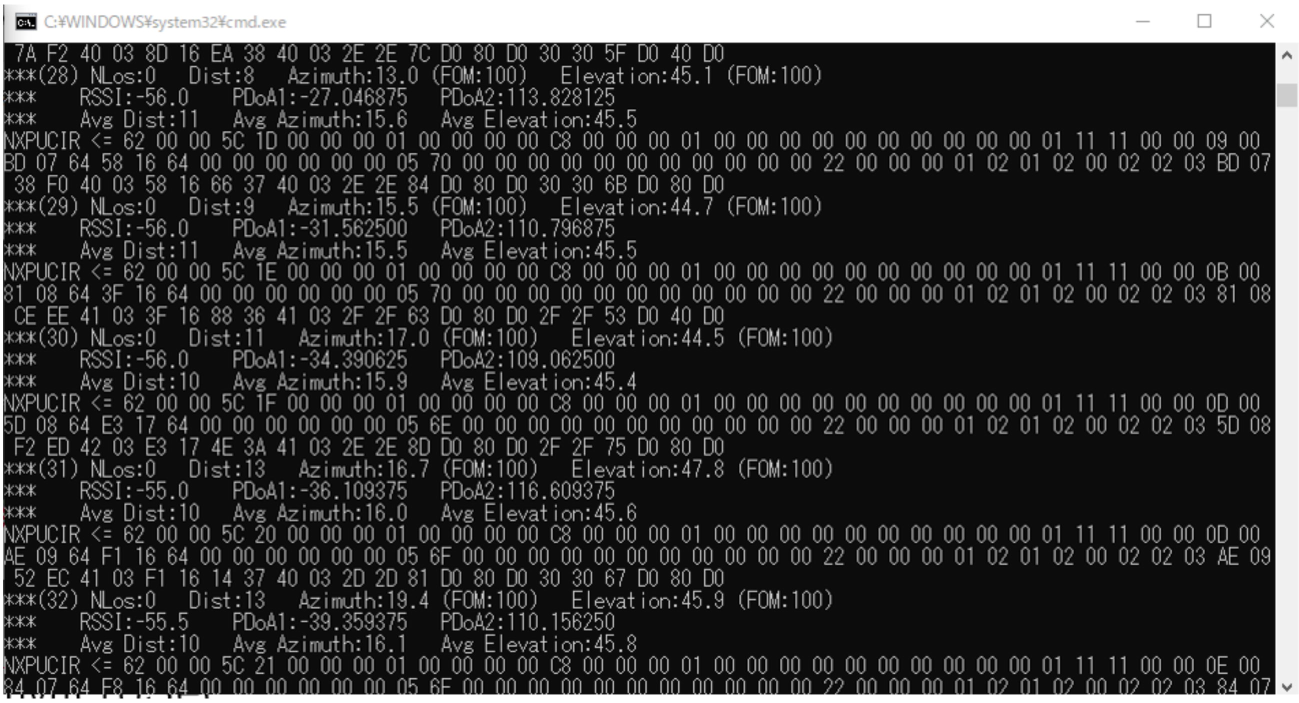
The console output for the PC controlling the Raspberry Pi with Type 2BP EVK (Controllee) is shown in **Figure 39**. The console output for the PC controlling the Type 2BP EVK (controller) is shown in **Figure 40**.

Figure 39: Controller Console Output for PnP Application



```
C:¥WINDOWS¥system32¥cmd.exe
02 40 03 00 E2 32 AB 40 03 32 32 21 D0 40 D0 2F 2F 3C D0 40 D0
***(28) NLos:0   Dist:14   Azimuth:-2.9 (FOM:100)   Elevation:-60.0 (FOM:100)
***    RSSI:-59.5    PDoA1:5.625000   PDoA2:-169.609375
***    Avg Dist:15   Avg Azimuth:-17.2   Avg Elevation:-52.0
NXPUCIR <= 62 00 00 5C 1D 00 00 00 01 00 00 00 00 C8 00 00 00 01 00 00 00 00 00 00 00 00 00 00 01 22 22 00 00 0A 00 F9
 F9 64 00 E2 64 00 00 00 00 00 00 05 7B 00 00 00 00 00 00 00 00 00 00 00 22 00 00 00 01 02 01 02 00 02 02 03 F9 F9 BA
11 3F 03 00 E2 62 C2 40 03 33 33 1C D0 40 D0 30 30 4F D0 80 D0
***(29) NLos:0   Dist:10   Azimuth:-12.1 (FOM:100)   Elevation:-60.0 (FOM:100)
***    RSSI:-61.5    PDoA1:35.453125   PDoA2:-123.234375
***    Avg Dist:15   Avg Azimuth:-17.9   Avg Elevation:-52.0
NXPUCIR <= 62 00 00 5C 1E 00 00 00 01 00 00 00 00 C8 00 00 00 01 00 00 00 00 00 00 00 00 00 00 01 22 22 00 00 0B 00 87
 F7 64 00 E2 64 00 00 00 00 00 00 05 78 00 00 00 00 00 00 00 00 00 00 00 22 00 00 00 01 02 01 02 00 02 02 03 87 F7 CC
17 3F 03 00 E2 B4 C5 40 03 31 31 1B D0 40 D0 30 30 58 D0 80 D0
***(30) NLos:0   Dist:11   Azimuth:-16.9 (FOM:100)   Elevation:-60.0 (FOM:100)
***    RSSI:-60.0    PDoA1:47.593750   PDoA2:-116.593750
***    Avg Dist:14   Avg Azimuth:-19.2   Avg Elevation:-52.0
NXPUCIR <= 62 00 00 5C 1F 00 00 00 01 00 00 00 00 C8 00 00 00 01 00 00 00 00 00 00 00 00 00 00 01 22 22 1B 01 07 00 2B
 FA 64 56 E9 64 00 00 00 00 00 00 05 80 00 00 00 00 00 00 00 00 00 00 00 22 00 00 00 01 02 01 02 00 02 02 03 2B FA C6
0D 40 03 56 E9 2E D8 40 03 32 32 58 D0 80 D0 2D 2F 4D D0 C0 D0
***(31) NLos:1   Dist:-7   Azimuth:-11.7 (FOM:100)   Elevation:-45.3 (FOM:100)
***    RSSI:-64.0    PDoA1:27.546875   PDoA2:-79.640625
***    Avg Dist:11   Avg Azimuth:-20.0   Avg Elevation:-50.6
NXPUCIR <= 62 00 00 5C 20 00 00 00 01 00 00 00 00 C8 00 00 00 01 00 00 00 00 00 00 00 00 00 00 01 22 22 00 00 0B 00 5C
 FD 64 00 E2 64 00 00 00 00 00 00 05 75 00 00 00 00 00 00 00 00 00 00 00 22 00 00 00 01 02 01 02 00 02 02 03 5C FD B6
06 41 03 00 E2 A2 CF 40 03 33 33 6C D0 80 D0 2F 2F 6F D0 80 D0
***(32) NLos:0   Dist:11   Azimuth:-5.3 (FOM:100)   Elevation:-60.0 (FOM:100)
***    RSSI:-58.5    PDoA1:13.421875   PDoA2:-96.734375
***    Avg Dist:10   Avg Azimuth:-19.7   Avg Elevation:-50.6
NXPUCIR <= 62 00 00 5C 21 00 00 00 01 00 00 00 00 C8 00 00 00 01 00 00 00 00 00 00 00 00 00 00 01 22 22 00 00 0F 00 A2
 04 64 00 E2 64 00 00 00 00 00 00 05 6F 00 00 00 00 00 00 00 00 00 00 00 22 00 00 00 01 02 01 02 00 02 02 03 A2 04 22
```

Figure 40: Controllee Console Output for PnP Application



```
C:¥WINDOWS¥system32¥cmd.exe
 7A F2 40 03 8D 16 EA 38 40 03 2E 2E 7C D0 80 D0 30 30 5F D0 40 D0
***(28) NLos:0   Dist:8   Azimuth:13.0 (FOM:100)   Elevation:45.1 (FOM:100)
***    RSSI:-56.0    PDoA1:-27.046875   PDoA2:113.828125
***    Avg Dist:11   Avg Azimuth:15.6   Avg Elevation:45.5
NXPUCIR <= 62 00 00 5C 1D 00 00 00 01 00 00 00 00 C8 00 00 00 01 00 00 00 00 00 00 00 00 00 00 01 11 11 00 00 09 00
BD 07 64 58 16 64 00 00 00 00 00 00 05 70 00 00 00 00 00 00 00 00 00 00 00 22 00 00 00 01 02 01 02 00 02 02 03 BD 07
 38 F0 40 03 58 16 66 37 40 03 2E 2E 84 D0 80 D0 30 30 6B D0 80 D0
***(29) NLos:0   Dist:9   Azimuth:15.5 (FOM:100)   Elevation:44.7 (FOM:100)
***    RSSI:-56.0    PDoA1:-31.562500   PDoA2:110.796875
***    Avg Dist:11   Avg Azimuth:15.5   Avg Elevation:45.5
NXPUCIR <= 62 00 00 5C 1E 00 00 00 01 00 00 00 00 C8 00 00 00 01 00 00 00 00 00 00 00 00 00 00 01 11 11 00 00 0B 00
81 08 64 3F 16 64 00 00 00 00 00 00 05 70 00 00 00 00 00 00 00 00 00 00 00 22 00 00 00 01 02 01 02 00 02 02 03 81 08
 CE EE 41 03 3F 16 88 36 41 03 2F 2F 63 D0 80 D0 2F 2F 53 D0 40 D0
***(30) NLos:0   Dist:11   Azimuth:17.0 (FOM:100)   Elevation:44.5 (FOM:100)
***    RSSI:-56.0    PDoA1:-34.390625   PDoA2:109.062500
***    Avg Dist:10   Avg Azimuth:15.9   Avg Elevation:45.4
NXPUCIR <= 62 00 00 5C 1F 00 00 00 01 00 00 00 00 C8 00 00 00 01 00 00 00 00 00 00 00 00 00 00 01 11 11 00 00 0D 00
5D 08 64 E3 17 64 00 00 00 00 00 00 05 6E 00 00 00 00 00 00 00 00 00 00 00 22 00 00 00 01 02 01 02 00 02 02 03 5D 08
 F2 ED 42 03 E3 17 4E 3A 41 03 2E 2E 8D D0 80 D0 2F 2F 75 D0 80 D0
***(31) NLos:0   Dist:13   Azimuth:16.7 (FOM:100)   Elevation:47.8 (FOM:100)
***    RSSI:-55.0    PDoA1:-36.109375   PDoA2:116.609375
***    Avg Dist:10   Avg Azimuth:16.0   Avg Elevation:45.6
NXPUCIR <= 62 00 00 5C 20 00 00 00 01 00 00 00 00 C8 00 00 00 01 00 00 00 00 00 00 00 00 00 00 01 11 11 00 00 0D 00
AE 09 64 F1 16 64 00 00 00 00 00 00 05 6F 00 00 00 00 00 00 00 00 00 00 00 22 00 00 00 01 02 01 02 00 02 02 03 AE 09
 52 EC 41 03 F1 16 14 37 40 03 2D 2D 81 D0 80 D0 30 30 67 D0 80 D0
***(32) NLos:0   Dist:13   Azimuth:19.4 (FOM:100)   Elevation:45.9 (FOM:100)
***    RSSI:-55.5    PDoA1:-39.359375   PDoA2:110.156250
***    Avg Dist:10   Avg Azimuth:16.1   Avg Elevation:45.8
NXPUCIR <= 62 00 00 5C 21 00 00 00 01 00 00 00 00 C8 00 00 00 01 00 00 00 00 00 00 00 00 00 00 01 11 11 00 00 0E 00
84 07 64 F8 16 64 00 00 00 00 00 00 05 6F 00 00 00 00 00 00 00 00 00 00 00 22 00 00 00 01 02 01 02 00 02 02 03 84 07
```

Revision History

Revision	Date	Author	Change Description
A	Mar. 4, 2022		Initial Release
B	Sep. 29, 2022		• Update for SDK 3.15.11 • Add PnP mode
C	Dec. 23, 2022		Update for SDK 4.02.01
D	May. 24, 2024		Update for SDK 04.06.00