Application Project Guide

Renesas PE-HMI1 Synergy S7 with Clarinox SPP Application

Contents

1.	Introduction	3
2.	Prerequisites	3
3.	Requirements	3
4.	Installation and Importing for e2 Studio	3
4.1.	Setting up Hardware	4
4.2.	Importing/Creating the project	5
4.3.	Configuring the project	8
4.4.	Building the Project	. 10
4.5.	Running the application	. 11
5.	Customizing the Application Project	. 19
5.1.	Application Source Files and Purpose	. 19
5.2.	Callback functions	. 20
5.3.	Threads	. 20

1. Introduction

This application demonstrates Bluetooth Serial Port Profile (SPP) of ClarinoxBlue stack with Bluetooth Classic running on the Renesas PE-HMI1 Synergy S7 platform. The application provides the use with the ability pair the Renesas PE-HMI1 with nearby Bluetooth devices and to transfer messages between devices.

A video explaining how to run SPP application on Renesas PE-HMI1 Synergy S7 can be found under following link.

http://www.clarinox.com/videos-clxblue_renesas_s7g2

2. Prerequisites

This document describes building the application project on e2 Studio and running it on Renesas PE-HMI1 Synergy S7 platform. This process requires the following prerequisites.

- Installing Renesas e2 studio and SSP Distribution on PC
- Installing "Bluetooth SPP Pro" app on Android mobile device
- Installing Clarinox Debugger tool on PC (Optional)

Installation instructions and the user guide for Clarinox debugger tool can be found in the document "Clarinox Debugger User Manual".

Also, being familiar with running applications on e2 studio and having PE-HMI1 Synergy S7 platform tested for basic functionality would be useful. Users can run some sample applications on the Synergy platform to check its functionality and to be familiar with the process.

3. Requirements

This application has the following hardware requirements.

- Renesas PE-HMI1 Synergy S7 kit
- Smart Phone or mobile device

Installing, building and running the application require the following tools and software to be pre-installed.

- e2 studio (tested with version 5_4_0_015)
- SSP Distribution (tested with version 1.2.0)
- "Bluetooth SPP Pro" Android Mobile App
- Clarinox Debugger (tested with version 3.2.219)

4. Installation and Importing for e2 Studio

This section includes step-by-step process of importing the project and running the application on Renesas PE-HMI1 Synergy S7.

4.1. Setting up Hardware

The hardware setup for running the application is shown in Figure 01. The PE-HMI1 Synergy S7 board can be powered over ethernet as shown in the picture. Also for debugging via J-Link, user can connect the J-Link connection via J12 which is not shown in the below figure.



Figure 01: Renesas PE-HMI1 Synergy S7 Setup for Programming and Debugging

In order to debug the application, user can use Clarinox Debugger tool which comes with a full detailed protocol analyzer when integrated with Wireshark allowing the users to analyze Bluetooth and Wi-Fi messages.

This application uses JLINK interface available on PE-HMI1 Synergy S7 for debugging.

When the board is powered over ethernet, Clarinox Debugger can be configured for the J-Link debug connection via *Tools -> Configuration -> SEGGER JLINK RTT Interface*. Configure the J-Link interface as shown in below screen capture.

Configuration Manager			×
General Configuration Lua Command Line Pane Message Browser Basic Message Datails Pane Clarinox Symbol File Manager TCP /IP Server UART COM Port UDP Server SEGGER: JLink RTT Interface	Parameter Host Interface JLink Emulator Serial Number JLink Emulator IP Address Target Interface Target Device Name Target Interface Speed RTT Control Block Address Range Start RTT Control Block Address Range End Target RTT Channel	Value USB 0x0 127.0.0.1 SWD R7FS7G27H 64000 0x1ffe0000 0x2007ffff 1	
Factory Reset			OK Cancel

Figure 02: SEGGER J-Link Interface Configuration on Clarinox Debugger

After setting the configurations if the board is powered on, user can start the J-Link debugger connection to the hardware via *Connection -> Start -> SEGGER JLINK RTT Interface*. User can interact with the application via debugger virtual console. More details on using Clarinox Debugger can be found in the document "Clarinox Debugger User Manual"

4.2. Importing/Creating the project

The structure of the project folder is shown in the following figure.



Figure 03: Project Folder Structure

Following table gives the details of the content of these folders.

Folder	Content
Арі	Clarinox APIs for Bluetooth, WLAN, Common and BSP
Demo	Project files and lib file for SppApp
Framework	Renesas Synergy project S7G2_PE_HMI1 framework
Lib	Clarinox Bluetooth and Softframe libraries
Source	Project source code with BSP (J-link and UART) etc

Table 01: Project Folder Structure

Following steps describe how to import BLE Central and Peripheral application project into e2 Studio workspace.

1. Click on File -> Import -> Existing Projects into Workspace

Import	
elect Create new projects from an archive file or directory.	N
Select an import source:	
type filter text	
 General Archive File CMSIS Pack Existing Projects into Workspace File System HEW Project Preferences Renesas CA78K0R Project Renesas Common Project File C/C++ C/C++ C/C++ C/C++ Executable C/C++ Project Settings Existing Code as Makefile Project Install 	H

Figure 04: Import Existing Project into Workspace

2. Select the root directory of the project and then two projects will appear under "Projects". Select all of them and click Finish. Then two projects named "SppApp" and "S7G2_PE_HMI1" will be loaded into the workspace.

			~
Select a directory to sear	ch for existing Eclipse projects.		
Select root directory:	D:\SppApplication	•	Browse
Select archive file:		•	Browse
Projects:			
S7G2_PE_HMI1 (I	D:\SppApplication\FrameWork)		Select All
SppApp (D:\Spp/	Application\Demo)	[Deselect All
		1	Refresh
2		14	
Options Search for nested pro	ojects rorkspace		
Options Options Search for nested pro Copy projects into w Hide projects that all Working sets	ojects orkspace ready exist in the workspace		
Options Search for nested pro Copy projects into w Hide projects that all Working sets	ojects orkspace ready exist in the workspace ing sets		
Options Search for nested pro Copy projects into w Hide projects that all Working sets Add project to work Working sets:	ojects orkspace ready exist in the workspace ting sets	*	Select
Options Search for nested pro Copy projects into w Hide projects that all Working sets Add project to work Working sets:	ojects orkspace ready exist in the workspace ting sets	*	Select

Figure 05: Locating the Project Root Directory

4.3. Configuring the project In order to build the projects, the path for e2 Studio utilities should be set under project environment variables. Right click on SppApp project and select *properties*. Then edit "PATH" variable under Environment as shown in the following figure. Click on "Edit" button add or modify the existing path.

type filter text	Environment			¢	• 🗘 • •
 Resource Builders C/C++ Build Build Variables Environment Logging 	Configuration: Debug [Active]		▼ Manage Con	figurations
Logging	Environment variables to s	et			Add
Logging Settings Tool Chain Editor Tools Paths C/C++ General Project References Run/Debug Settings	Variable AMS_KEEP_FILE AMS_LICENSE_PATH CWD PATH PWD	Value \${synergyKeepFile} \${synergyLicenseFile} D:\SppApplication\Dem C:\Program Files (x86)\ D:\SppApplication\Dem	Origin USER: PREFS USER: PREFS BUILD SYSTEM USER: CONFIG BUILD SYSTEM		Select Edit Delete Undefine
	 Append variables to na Replace native environ 	tive environment ment with specified one		Restore Defaults	Apply

Figure 06: Edit PATH Variable for SppApp

If the path for utilities folder is already added then check for the correct path, if not add the correct path, eg: "D:\Renesas\e2_studio\Utilities".

Name:		PATH	
Value:		ıs\e2_studio\eclipse;D:\Renesas\e2_studio\Utilities	Variables
	Cancel	1	

Figure 07: Add or Modify PATH Variable

The same should be added under PATH variable for S7G2_PE_HMI1 project as well.

The libraries and preprocessor definitions should already be included under SppApp project's build settings as shown below.



Figure 08: Included Libraries under BleGattApp Build Settings

4.4. Building the Project

The order of building two projects is as follows.

- 1. SppApp
- 2. S7G2_PE_HMI1

In order to build the SppApp project, right click on the project and select "Build project". This should generate SppApp.a library in the Debug folder of the project.

Copy the generated SppApp.a library to the Lib folder in S7G2_PE_HMI1 project which already contains WLAN, Bluetooth, Softframe and WiLink libraries. The Lib folder and the libraries should be included to the project as shown in Figure 09.

Right click on S7G2_PE_HMI1 project and select "Build project" to build the project.

SppApp and S7G2_PE_HMI1should build with few compiler warnings.



Figure 09: Included Libraries for S7G2_PE_HMI1 Project

4.5. Running the application

In order to run the application, the PE-HMI1 Synergy S7 board should be assembled as shown in Figure 01. Power on the board via DEBUG_USB by connecting the micro-USB to PC.

To debug the application right click on S7G2_PE_HMI1 Synergy project and select *Debug* As -> 2 Renesas GDB Hardware Debugging. User can also click on the debug icon on e2 studio to debug the application.

If this project is run for the first time then it will ask for the debug hardware. Click on the J-Link ARM. Then select from the given list of devices as shown below. Then it will start downloading the application on the PE-HMI1 Synergy S7.

R7FS12476	
R7FS12477	_
R7FS12878	=
R7FS3A37A	1
R7FS3A7	
R7FS3A77C	
R7FS5D97C	
R7FS5D97E	
R7FS7G2	
R7FS7G27G	
R7FS7G27H	
R7S721000	
R7S721000_DualSPI	
R75721001	-

Figure 10: Select the Device for PE-HMI1 Synergy S7

If Clarinox Debugger is used for debugging the application make sure to start the J-Link debugging connection just after downloading the program on PE-HMI1 Synergy S7. Start Clarinox Debugger connection via *Start -> Connection -> SEGGER JLink RTT Interface*.

When the SPP application starts running on the Renesas PE-HMI1 Synergy S7, a menu will be displayed on the debugger console as shown in the following picture. The selections will provide the user with options to use the SPP application for initialization and termination Bluetooth stack, searching for Bluetooth devices, connecting to Bluetooth devices and using the provided profiles.

Figure 11: Main Menu for SPP Application

Below steps show running of an example scenario with SPP application.

The first step is to initialize the Bluetooth stack. The stack initiates connection to the Controller via UART (or USB or any other method specified) at this point. Below shows the console output when executing this option.



Figure 12: Initializing Bluetooth Stack

Then the user can make the device (PE-HMI1 Synergy S7) discoverable and connectable with the option 4 in the menu. This will make the device appear in Bluetooth scan results on mobile devices.

User can tap on the name appeared on mobile device as "ClxSppTest" to pair with PE-HMI1 Synergy S7 and this will ask to confirm a passkey as shown in Figure 13. The same confirmation procedure will happen on Synergy SK-S7G2 as well.

Once this pairing process is happened ClxSppTest can connect to any paired device at any time by entering the menu option 3.



Figure 13: Synergy SK-S7G2 Pairing Process on Mobile Device

The following figure shows the result on debugger console when the pairing process completed.



Figure 14: Synergy SK-S7G2 with Mobile Device Pairing Process on Debugger Console This example uses "Bluetooth Spp Pro" mobile app to communicate data between two devices. Please note that first of all, Synergy SK-S7G2 Bluetooth device should be made discoverable by entering option 6 in the menu. This will allow mobile app to detect the device.

On the mobile app, select the device (Synergy SK-S7G2 as ClxSppTest) once it appears on the screen and tap on the connect button. Then a sub menu to communicate with the connected device will appear on the debugger console. Tap on the "Byte Stream Mode" button on the mobile app to initiate the communication by opening a message console as shown in the following figure.

⊉ 🖳 🖬 🗱 🕷 🖗	6:59 AM	Ý 🗈 🔹 🗱	🗱 🛜 🖌 📔 7:00 AM	Ý 🖳 🖬	🗚 🕅 🕄 🕯 🗐 🖍 🕅 🕅 🕯
🔊 Scan Device	SCAN CLOSE	Bluetooth spp pro	RESCAN	👋 Bluetooth spp p	ro rescan
ClxSppTest	RSSI	Connect the device:		Connect the device	د
MAC: 00:1B:DC:06:5D:E5 CoD: ff2420	-50 Rondod	Device name: ClxSppTest		Device name: ClxSppTest	
Device Type: BR/EDR Bluetooth	Bonded	Mac addr: 00:1B:DC:06:5D:E5		Mac addr: 00:1B:DC:06:5D:	5
Simon's Mac Pro	-85	Signal: -50		Class of device:ff2420 Signal: -50	
MAC: 18:AF:61:B9:23:D3 CoD: 1100 Device Type: BB/EDB Bluetooth	Nothing	Type: BR/EDR Bluetooth		Type: BR/EDR Bluetooth	
TI CN130-93B752	RSSI	billu state. bollueu		Bind state: Bonded	
MAC: A0:E6:F8:93:B7:52 CoD: 1f00	-78	Service's UUID:		Service's UUID :	
Device Type: BR/EDR Bluetooth	Nothing	00001101-0000-1000-8000-008)5f9b34fb	00001101-0000-1000-8000	-00805f9b34fb
MI Band 2	RSSI				
MAC: E1:D2:98:8C:89:E5 CoD: 1f00	-98 Nothing				
Device Type: BR/EDR Bluetooth	Notifing	Connect		Select commu	nication mode
CIxSppTest	-68]	Duto stro	am mode
MAC: 00:1B:DC:06:77:98 CoD: ff2420 Device Type: BB/EDB Bluetooth	Nothing			byte strea	ammode
ClySpp Windows	RSSI			Walter I and	0110
MAC: 00:1B:DC:06:5C:27 CoD: ff2420	-64			Keyboard mode	CMD line mode
Device Type: BR/EDR Bluetooth	Nothing				
Simon's Mac Pro	RSSI				
MAC: 41:DD:22:DC:13:35 CoD: 1f00	-85				
Device Type: BR/EDR Bluetooth	Nothing				
Select [110] : 4 Waiting for incomin	ng connec	tion request			
Please confirm the	e passkey	"435863" to conn	ect to "CLAR]	INOX G S4" ? [y]	(nN] : y
Authentication com	pleted				
SPP connected to B	luetooth 1	Device.			
Enter your selection	on:				=
1. Connect to	SPP				
2. Send ASCII	Text				
3. Send Prebuil	lt Test D	ata			
4. Disconnect	from SPP				
5. Return to p	revious m	enu			
Select [15] :					-

Figure 15: Connecting to ClxSppTest via Bluetooth SPP Pro and the Debugger Console Output Then the user can send some text from ClxSppTest to the mobile device by entering the 2nd option "Send ASCII Text" in the sub menu. User can enter a message with up to 48 characters on the debugger console and this message will be received by the mobile app as shown in the following figures.



Figure 16: Sending a Text Message from ClxSppTest (Synergy SK-S7G2)

User can also send a block of prebuilt test data to find the speed of the connection by entering the 3rd option on ClxSppTest sub menu. This will take few seconds to transmit the data and when it is finished the data will appear on the mobile device and the speed of the connection will be displayed on the debugger console as shown below.

<pre>Inter your selection: 1. Connect to SPP 2. Send ASCII Text 3. Send Prebuilt Test Data 4. Disconnect from SPP 5. Return to previous menu Select [15] : 3 peed = 131000 Bytes/Sec Inter your selection: 1. Connect to SPP 2. Send ASCII Text 3. Send Prebuilt Test Data 4. Disconnect from SPP 5. Return to previous menu Select [15] :</pre>				
 Connect to SPP Send ASCII Text Send Prebuilt Test Disconnect from SI 	; Data			
5. Return to previous	menu			
Select [15] : 3	10 J			
Speed = 131000 Bytes/Sec				_
Enter your selection:				
1. Connect to SPP				
2. Send ASCII Text				
3 Send Prebuilt Test	Data			
4 Disconnect from SI	D			
5. Return to previous	menu			
o. Revain oo pictica				
Select [15] :				-
	Ý == 🔜 🔺 🕅	🛠 🕄 📶 🖸 7:07 AM		•
	< 🧩 Byte stream mode	CLEAR		
	Txd: 0B Rxd: 43333B	Running: 94s		
	Waiting to receive…			
	Hi This is a message fro	om Synergy S7G2-		
	5K!"#\$%&`()*+,-			
	VZ[\]^ `abcdefghiik]mnopgr	stuvwxvz{ }~Пôôôô		
	000000000000000000000000000000000000000	000000000000000000000000000000000000000		
	000000000000000000000000000000000000000	000000000000000000000000000000000000000		



Figure 17: Sending a Block of Prebuilt Test Data from ClxSppTest (Synergy SK-S7G2)

The reverse scenario of sending messages from the mobile device to ClxSppTest can also be achieved if the user types the message on the mobile device and tap on send icon which will then be received by the other side. This scenario is shown in the following figure.

<pre> Pyte stream mode Tyd.08 Rxd.433333B Running.94s Hithis is a message from the mobile device ! his this thus 1 2 3 4 5 6 7 8 9 0 Q w e r t y u i o p a s d f g h j k I z x C v b n m sym registriUSS registriUSS registriUSS xter your selection: 1. Connect to SPP Send ASCII Text Send Probuilt Test Data Disconnect from SPP Return to previous menu </pre>		Ý == 🖳 🛋 🔧	3 🐼 🛜 📶 🖸 7:08 AM		
Dvd: 0B Rxd: 43333B Running. 94s Hithis is a message from the mobile device ! Image: Compare the mobile device ! his this thus 1 2 4 5 7 9 0 Q W r tyuiop 1 2 4 5 7 9 0 Q W r tyuiop yuiop Image: Compare the second		(📣 Byte stream mode	CLEAR		
Hi. this is a message from the mobile device ! his this thus 1 2 3 4 5 6 7 8 9 0 q w e r t y u i o p a s d f g h j k l i z x c v b n m sym English(US) . end ter your selection: 1. Connect to SPP 2. Send ASCII Text 3. Send Prebuilt Test Data 3. Send Prebuilt Test Data 4. Disconnect from SPP 5. Return to previous menu		Txd: 0B Rxd: 43333B	Running: 94s		
Hithis is a message from the mobile device ! his this thus 1 2 3 4 5 6 7 8 9 0 Q w e r t y u i o p a s d f g h j k l 2 x c v b n m sym English(US) extra selection: 1. Connect to SPP 2. Send ASCII Text 3. Send Prebuilt Test Data 3. Send Prebuilt Test Data 3. Send Prebuilt Test Data 3. Send Prebuilt Test Data 4. Disconnect from SPP 5. Return to previous menu					
Hithis is a message from the mobile device ! his this thus > his this thus > 1 2 3 4 5 6 7 8 9 0 > > > Q w e r t y u i o p a s d f g h j k l > > > a s d f g h j k l > z x c v b n m > > > Sym English(US) ? + > <td></td> <td></td> <td></td> <td></td> <td></td>					
Hi. this is a message from the mobile device ! his this thus 1 1 2 3 4 5 6 7 8 9 0 q w e r t y u i o p a s d f g h j k I r z x c v b n m sym <u>e registry</u> <u>r</u> c ter your selection: 1. Connect to SPP 2. Send ASCII Text 3. Send Prebuilt Test Data 4. Send Prebuilt Test Data 5. Return to previous menu					
Hi. this is a message from the mobile device ! his this thus > 1234567890 0 0 0 ertyuiop 1234567890 0 0 0 ertyuiop a d f g 0 0 q ertyuiop a g f g 0 q ertyuiop a g f g 0 q ertyuiop a g f g					
Hi this is a message from the his this this <t< td=""><td></td><td></td><td></td><td></td><td></td></t<>					
Hi. this is a message from the his this					
Hi. this is a message from the mobile device ! his this 1 2 3 4 5 6 7 8 9 0 Q W e r t y u i o p a s d f g h j k l r z x c v b n m <					
mobile device ! his this 1234567890 qwertyuiop 1234567890 qwertyuiop asdfghjkl zxcvbnm≪ Sym English(US) i. Connect to SPP . Send ASCII Text . Send Prebuilt Test Data . Disconnect from SPP 5. Return to previous menu		Hi., this is a message fro	m the		
his this thus 1 2 3 4 5 6 7 8 9 0 q wertyuiop a s d f g h j k l 1 z x c v b n m Sym English(US) . Connect to SPP 2. Send ASCII Text 3. Send Prebuilt Test Data 4. Disconnect from SPP 5. Return to previous menu		mobile device !	\triangleright		
<pre>nis nus unus v 1 2 3 4 5 6 7 8 9 0 q w e r t y u i o p a s d f g h j k l r z x c v b n m sym English(US) ?? ter your selection: 1. Connect to SPP 2. Send ASCII Text 3. Send Prebuilt Test Data 4. Disconnect from SPP 5. Return to previous menu Next fi 5 k Wi this is a measure form the mehile device of the second secon</pre>		hte date			
<pre>1 2 3 4 5 6 7 8 9 0 q w e r t y u i o p a s d f g h j k l f z x c v b n m * Sym * English(US) ?? * *********************************</pre>		nis tnis	thus 🗸		
q wertyuiop asdfghjkl Tzxcvbnm Sym English(US) .ter your selection: 1. Connect to SPP 2. Send ASCII Text 3. Send Prebuilt Test Data 4. Disconnect from SPP 5. Return to previous menu		1 2 3 4 5 6	7 8 9 0		
q w e r t y u i o p a s d f g h j k l ↑ z x c v b n m × Sym ☆ English(US) . ? • • • • • • • • • • • • • • • • • • •					
a s d f g h j k l T z x c v b n m Sym Sym English(US) 1. Connect to SPP 2. Send ASCII Text 3. Send Prebuilt Test Data 4. Disconnect from SPP 5. Return to previous menu		qwerty	uıop		
a S d T g n j k i ↑ Z X C V b n m Sym English(US) 					
<pre>x c v b n m x ym x English(US)</pre>		asarg	пјкі		
Sym & English(US) ? 4 Sym & English(US) ? 4 I. Connect to SPP 2. Send ASCII Text 3. Send Prebuilt Test Data 4. Disconnect from SPP 5. Return to previous menu					
Sym is English(US) Let your selection: 1. Connect to SPP 2. Send ASCII Text 3. Send Prebuilt Test Data 4. Disconnect from SPP 5. Return to previous menu					
<pre>ter your selection: 1. Connect to SPP 2. Send ASCII Text 3. Send Prebuilt Test Data 4. Disconnect from SPP 5. Return to previous menu loct [1 5] : Wi this is a presence from the public device 1</pre>		Sym 🚆 English(US			
ter your selection: 1. Connect to SPP 2. Send ASCII Text 3. Send Prebuilt Test Data 4. Disconnect from SPP 5. Return to previous menu					
 Connect to SPP Send ASCII Text Send Prebuilt Test Data Disconnect from SPP Return to previous menu 	ter your selection:				
1. Connect to SPP 2. Send ASCII Text 3. Send Prebuilt Test Data 4. Disconnect from SPP 5. Return to previous menu					
3. Send Prebuilt Test Data 4. Disconnect from SPP 5. Return to previous menu	1. Connect to SPP				
4. Disconnect from SPP 5. Return to previous menu	 Send Aschr lext Send Prebuilt Te 	st Data			
5. Return to previous menu	4. Disconnect from	SPP			
lost [1 E] . His this is a message from the mobile device	Return to previo	is menu			
	elect [1 5] · Hi th	is is a message from	the mobile device	1	

Figure 18: Sending a Text Message from Mobile Device

Following screen capture shows an example of Bluetooth protocol messages captured in Clarinox Debugger Bluetooth Monitor showing lower level message details when the data is transferred with SPP Application.

😂 Clarinox Debugger						– ø ×
Connection View Browse Tools Help						
🕨 🖸 🛍 🖬 🛊 🖬 🛔 🔍 🛦 🕲 🖪 📕 📜						
Message Details	• • X Message Browser1	ClarinoxBlue Monitor1 🗙 📕 Clarin	iox SDIO Monitor1 🔳Clarinox Tasking Monitor1 🔳Memo	ry Monitor1 Memory Poolset1		-
# Rivetooth MCI MI	Time D	Upper Lie Nei Trap	Frotocol Message	Trace Buffer	Trace Buffor (hox)	^
E Bluetooth HCLACL Parket	00100107.945 0	((BCI) ACL_DERA (LECAP_START) (LECA) LECAP_SED (SDP)	R	48 20 18 00 14 00 42 00 16 00 00 00 ++ ++ ++ ++	
0000 0100 1000 = Connection Handler 0x00/4				5	35 05 0a 00	
- 00 = 00 Elas: Eist Non-automatically Elysh	00108107.945 0		(SDP) SDP_SERVICESEARCHATTR_REQ		06 00 00 00 02 35 03 19 12 00 02 90 35 05 0m 00	
00 = BC Flag: Point To Point (0)	00100107.945 0	[mas]	(SDP) SDP SERVICESEARCHAFTE RSP	y.v6.#6.p	07 00 00 00 79 00 76 36 00 73 36 00 70 09 00 00	
Data Total Length: 120					08 00 01 00 00 09 00 01 36 00 03 19 12 00 09 00	
Cananat in forman 40					04 35 08 35 03 19 01 00 33 03 19 00 01 09 00 05 35 03 19 10 02 09 00 06 35 09 09 65 68 09 00 68	
Connect In name: 40		i i i i			09 01 00 09 00 08 08 07 09 00 09 35 08 35 06 19	
Source bb_ADDic Vencer_00.5cces (00.1b:0c.0b:5cce	(9)				12 00 09 01 03 09 01 00 25 00 09 02 00 09 01 03	
Source Device Name:	and the second sec				04 10 09 02 04 29 01 09 02 05 09 00 01 00	
Source Role: Master (1)	00100107.945 0	1 [see5] [(LDC) LICAD_BDU (SDF)	×.Jy.v6.a6.	76 00 4a 00 67 00 00 00 79 00 76 36 00 73 36 00	
Destination BD_ADDic Samsungs_e1:50:4F (64:11:9e			LIDOTS BOX DETA UNVERSION	Prix.	70 09 00 00	
Destination Device Name:	001001071946 0		(HCI EVT) NBR OF COMPLETED PACKETS		13 05 01 48 00 01 00	
Destination Kole: Slave (2)	00:00:07.947 1	<	(HCI) ACL_DATA (L2CAP_START)	E	49 20 08 00 04 00 41 00 35 73 01 cd	
Last Role Change in Frame: 38	00:00:07.947 0	Canal I	(LIC) LICAP_SEC (RECORD)		04 00 41 00 35 73 01 cd 35 73 01 cd	
Current Mode: Active Mode (0)	00:00:07.947 0	>	(RPC_ULI) NDC		03 ef 09 e3 05 3b 0d 70	
Last Mode Change in Frame: 40	00100107.947 0	1 1>1 1	(LTC) LICAN BLO (RECOMM)	**0******P	08 00 4£ 00 03 e£ 09 e3 05 3b 0d 70	
Bluetooth L2CAP Protocol	00100107.048		(BCI RVF) NBR OF COMPLETED PACKETS	8	13 05 01 48 00 01 00 13 05 01 48 00 01 00	
Length: 126	00:00:07.951	C+++	(HCI) ACL DATA (LICAP START)	НВ.J.	48 20 0m 00 08 00 01 00 06 07 04 00 42 00 4a 00	
-CID: Dynamically Allocated Channel (0x004a)	00:00:07.951 0	1 (C+++)	(LIC) L2CAP_DISCONNECTION_SEQUEST		08 00 01 00 06 07 04 00 42 00 44 00	
Connect in frame: 90	00:00:07.952 1		(RCI) ACL DATA (UNKNOWN)	8B.J.	48 00 00 00 00 00 01 00 07 07 04 00 42 00 4a 00	
PSM: SDP (0x0001)	00100107.952 0	I I ICanal	(NCI_HVE) NER_OF_COMPLETED_DACKETS		13 05 01 40 00 01 00	
⊟ Bluetooth SDP Protocol	00100107.954 2	(Cana)	(HOI) ACL DATA (LICAP START)	Rh	48 20 0c 00 08 00 41 00 01 cf 09 c1 05 35 36 4a	
PDU: Service Search Attribute Response (0x07)	00100107.955 0	((RFC_UIE) NSC		01 ef 09 el 05 30 0d an	
Transaction Id: 0x0000	00:00:07.955 1	<	(HCI) ACL_DATA (L2CAP_START)	R	49 20 0c 00 00 00 41 00 01 ef 09 e3 05 3b 9d as	
Parameter Length: 121	00:00:07.955 0	(case)	(EPC (CH) NAC		08 00 41 00 01 et 09 e3 05 35 6d as	
Attribute List Byte Count: 118	00:00:07.955 0	i i içenen i	(RPC_ULH) NBC		03 mE 09 ml 05 3b 0d 70	
Attribute Lists (count = 1)	00100107.955 0	S	(LTC) LTCAP BOD (RECOMM)	0 <i>i</i> .p	00 00 4F 00 03 eF 09 el 05 3h 04 70	
E Data Element: Sequence uint16 115 bytes	00100107.953 0	Contraction of Contraction	(BCI) ACL DRTA (LOCAP START)	K	48 20 00 00 08 00 01 00 02 08 04 00 01 00 46 00	
0011 0 = Data Element Type: Sequence (6)	00:00:07.568 0	I I Caral I	(LTC) LICAP CONNECTION REQUEST	K.	08 00 01 00 02 08 04 00 01 00 45 00	
110 - Data Flament Size uint16 (6)	00:00:07.961 1		(SCI_SVT) COMMAND COMPLETE		0e 04 01 1a 5g 00	
Data Element Var Size: 115	00100107.961 0	>	(HCI HC) WRITE SCAN ENABLE		10 00 01 00	
iei Date Value	00100107.966 5	(***)	(HCI_EVE) COMMAND_COMPLETE		0c 01 01 1a Cc 00	
Continuation States on (00)	00100107.567 1	L Level 1	(LIC) LICAP CONNECTION RESPONDE	C.R		
Contendaden State: No (00)	00100107.967 0	>	(L2C) L2CAP CONFIGURE REQUEST	·····	00 00 01 00 04 07 04 00 45 00 00 00	
	00:00:07.967 0	· · · · · · · · · · · · · · · · · · ·	(HCI) ACL_DATA (UNKNOWN)	EC.X.	40 00 10 00 00 00 01 00 03 00 08 00 ++ ++ ++ ++	
1.	00100101.968 1	1 1	(BCI) SCLORIS (UNKNOWN)			~
Virtual Console			▼ # X Lua Comm	and Line		▼ ‡ X
Discovering CON ports on the remote device			* creater	ypeRatchingSule		~
Enter your selection:			octVors spplytt	ion phlightRule		
1 000000 00 000			creates	tringMatchingRule		
2. Send ASCII Text			help	congen		
3. Send Prebuilt Test Data			createl	uaMatchingRule		
5. Return to previous menu			Call co	.help("function name") in order to get	a description of each function.	
Select [15] : 3			The com	mos library 'od_common.lus' Loaded.		
Speed = 135000 Bytes/Sec			The life	rary 'cd memory.lus' Loaded :		
Enter your selection:			fur fur	ection : nigniightrool@aste ection : calculatePool@aste		
2. Send ASCII Text 3. Send Probuilt Text Data			cd_	init.lus END		
4. Disconnect from SPP S. Return to previous meau			> od.ge 3,2,215	EVeraion()		
Select [15] -			× >			~
			a possibilitaria de la companya de la compa			
«No Active Comm Interface»			Session 1	00:	05:06 idle 758 KB Recei	bed

Figure 19: ClarinoxBlue Protocol Monitor on Clarinox Debugger

Please note that a user manual explaining all the functionality of SPP application is available in the project folder.

5. Customizing the Application Project

This section guides the user to exploit the complete functionality of the system.

5.1. Application Source Files and Purpose

Clarinox IoT Application source files in "source" folder are provided under three categories;

- 1. Clarinox wireless application source files
- 2. Renesas Synergy platform support files
- 3. Third party interface files

5.1.1. Clarinox wireless application source files

In the first category, there are Bluetooth and Bluetooth low energy application files provided.

Main.cpp file which provides the Bluetooth stack configuration and initialization functionality. In addition, a console based simple menu allows basic inquiry, inquiry scan and connection functionalities.

SPP.cpp file provides the creation, deletion of the SPP profile in addition to a simple SPP menu. The menu provides connect/disconnect and send/receive functions. In the same file, a callback function is provided to handle events delivered by Clarinox middleware.

5.1.2. Renesas Synergy platform support files

BspOs.cpp file provides the Renesas platform ThreadX RTOS interface functionality.

Bsp.cpp file provides memory pool setup for Clarinox wireless components. In addition, terminal input and console print functionalities can be configured in this file. Platform specific setup of the Bluetooth configuration parameters are also set as part of the Board Support Package (BSP) initialization. Other Renesas Synergy Starter Kit or HMI board specific hardware settings can be found in this file.

5.1.3. Third party interface files

Segger JLink/RTT files are used to provide a back channel for connecting the Renesas Synergy platforms to PC based Clarinox debugger. An alternative mechanism is to use of UART interface.

5.2. Callback functions

In the Spp.cpp file, "stackMessageHandler" callback function is provided to handle Bluetooth related events delivered by Clarinox middleware. Any events raised by GAP and SPP profiles cause this call-back function executed with the associated event and parameters.

Users can customize this callback function to perform a task based on the type of indication. For an example, in SPP Application when there is an Indication of a SPP connection, user can turn off discoverability and connectability features to save power.

5.3. Threads

Threads are dynamically created as required, e.g. if the Bluetooth stack is started, then a thread is created and the associated scheduler is run on this thread. Thread priorities and Thread stack sizes are provided as part of the board support package (in Bsp.cpp file).

Configurable items are set by using "clxConfigInitIntegerParam" function call, an example is shown as follows;

```
clxConfigInitIntegerParam(&linkRequestTimeout, "LinkRequestTimeout", 16000,
configList); /* Link request timeout is set to 16 * 1.25 seconds */
```