NXP Semiconductors Application Note Document Number: AN12273 Rev. 1, 11/2018

Using MCUXpresso SDK CAN and LIN Drivers to Create a Bluetooth LE-CAN and Bluetooth LE-LIN Bridges on KW36/KW35

1. Introduction

Bluetooth Low Energy usage in automotive and industrial applications is growing. These applications often require integration with other standard communication protocols like CAN and LIN.

NXP offers KW35/KW36 devices that enable Bluetooth Low Energy, CAN and LIN communication for automotive and industrial embedded systems.

NXP provides a complete Bluetooth LE stack solution to which CAN and LIN drivers can be added to create a bridge application between CAN or LIN and Bluetooth LE.

2. Abstract

This application note describes the process to create a new Bluetooth Low Energy project for the FRDM-KW36 development board and MCUXpresso IDE, and how to add CAN and LIN drivers into the new project to create BLE-CAN and BLE-LIN bridges.

This application note includes source and header files to download for a Bluetooth LE to CAN and LIN bridge demo example.

Contents

1.	Introc	luction	1
2.	Abstr	act	1
3.	Creat	ing a new Bluetooth LE project	2
4.	Addir	g FlexCAN and LIN drivers	6
5.	Addir	g FlexCAN and LIN demo examples into a B	luetooth
	LE pr	oject	9
	5.1.	FlexCAN	9
	5.2.	LIN	12
6.	Addir	ng AES 128 security to FlexCAN data frames	15
7.	Bluet	ooth LE-CAN-LIN Bridge Demo	16
	7.1.	Overview	16
	7.2.	Prerequisites	19
	7.3.	Set Up	19
	7.4.	Software Modifications	26
8.	Revis	ion history	28
		-	



© 2018 NXP B.V.

3. Creating a new Bluetooth LE project

Follow below steps to create a new FRDM-KW36 Bluetooth Low Energy project for MCUXpresso IDE:

- 1. Download the latest FRDM-KW36 SDK for MCUXpresso IDE from MCUXpresso SDK Builder webpage:
 - a. Go to https://mcuxpresso.nxp.com.
 - b. On welcome page, click on "Select Development Board". See Figure 1



Figure 1. MCUXpresso SDK Builder

. .

c. Search for "FRDM-KW36" and click on "Build MCUXpresso SDK". See Figure 2

Select Development Board Search for your board or kit to get started.		
Search by Name	Hardware Details	
kw36	Board	FRDM-KW36
	Device	MKW36Z4
Select a Device Board or Kit	Core Type / Max Freq	Cortex-M0P / 48MHz
▼ Boards	Memory Size	256 KB Flash 64 KB RAM
FRDM-KW36		_
▼ Kits	Actions	
 Processors 	→ Build MCUXpres	sso SDK
MKW36A512xxx4		
MKW36Z512xxx4	Explore selectio	n with Clocks tool
Name your SDK	Explore selectio	n with Pins tool
SDK_2.2.0_FRDM-KW36		

Figure 2. Select development board window

d. Select "MCUXpresso IDE" and your Host OS, then click on "Download SDK" and agree with software terms and conditions to start your FRDM-KW36 SDK download. See Figure 3

SDK Builder Generate a downloadable SDK archive for use with desktop MCUXpresso Tools.						
Developer Environment Settings Selections here will impact files and examples proj	ects included in the SDK and Generated Projects					
Host OS Toolchu Windows - MCI	in / IDE JXpresso IDE •					
Select Optional Middleware Add middleware, operating systems, and software libraries to your SDK.						
Add software component						
This MCUXpresso SDK configuration is	available for direct download					
Download SDK	Archive Name					
Dominad ODA	Don't use: <>					

Figure 3. SDK Builder window

- 2. Download and install the latest version for MCUXpresso IDE:
 - a. Go to webpage: <u>https://www.nxp.com/support/developer-resources/software-</u> <u>development-tools/mcuxpresso-software-and-tools/mcuxpresso-integrated-development-</u> <u>environment-ide:MCUXpresso-IDE</u>
 - b. Download and install the MCUXpresso IDE executable located in "Downloads" tab.
- 3. Import your FRDM-KW36 SDK into MCUXpresso IDE:
 - a. Open MCUXpresso IDE.
 - b. Go to "Installed SDKs" tab, then drag and drop your FRDM-KW36 SDK.



Figure 4. . Importing SDK into MCUXpresso IDE

- 4. Create a new Bluetooth LE project from your FRDM-KW36 SDK:
 - a. Click on "Import SDK example(s)" from the Quickstart Panel.

UQuicks	🕬= Global	(x)= Variab	⁰o Break	🗄 Outline	0
	UXpress	o IDE - Q	uickstar	t Panel	
No p	project selec	ted			
- Create o	or import a j	project			
	New project. mport SDK e mport projec	xample(s) ct(s) from file	e system	_	

Figure 5. Import SDK example(s).

b. Search and select "frdmkw36" in the "board and/or device selection page", then click "next" See *Figure* 6

X SDK Import W	izard							_	_ □	×
 Importing proje 	ect(s) for device:	MKW36Z512xx	ox4 using board:	FRDM-KW36				N	P	G
X Board ar	nd/or Device	e selection	page							^
* SDK MCUs		Available b	oards						Jªz ↑ªz	a a a a a a a a a a a a a a a a a a a
MCUs from insta	lled SDKs	Please selec	ct an available b	oard for your project.						
KW362	2512004		frdmkw36	5DK						
Selected Device	MVW267512	www.using.hon		c	SDKs for selected MC					
Terret Corre	. WIKW 302312X		IU. FROMI-RWO	0	Name	SDK Verci	Manifect	Location		
Description:	cortex-moplus	5			# SDK 2.x FRDM-KW	36 2.2.0	3.1.0	Cocation	ocation>\SDk	\$ 22
2020 priori	KW36Z: Kinetis connectivity W	s KW36Z - 2.4 (/ireless Radio N	GHz BLE and Ger Aicrocontroller (I	neric FSK MCU) based on						
						1				Ŷ
?					< Ba	ck N	lext >	Finish	Car	ncel

Figure 6. Board and/or Device selection page

c. Select the desired Bluetooth LE project under "wireless_examples" and "bluetooth" (project can be on baremetal or freertos), change SDK Debug Console to "UART", and click "Next" See *Figure 7*. For this example, Wireless UART example in FreeRTOS is used.

SDK Import Wizard			_ ×
A The source from the SDK will be copied into the workspace. If you want to use linked files, please unzip the 'SDK_2x_FRDM-KW36' SDK.			
Import projects			
Project name prefix: frdmkw36_	2 Project name suffix:		<i>a</i>
Use default location			
Location: C:\Users\nxa06676\Documents\MCUXpressoIDE_10.2.1_795\work	space\frdmkw36_		Browse
Project Type	Project Options		
C Project C++ Project C Static Library C++ Static Library	SDK Debug Console Semihost UART Copy sources Import other files		
Examples		2	2 🙀 🗈 🖻
type to filter			
Name	Version		~
0	< Back Next > Finis	h	Cancel

Figure 7. . Import project window

d. In Advance Settings, disable "Include semihost HardFault handler" and click "Finish". Your project will be created. See *Figure* 8.

Adding FlexCAN and LIN drivers

SDK Impor	t Wizard						
					N	K	
Advan	ced Settings						
• C/C++ Lil	brary Settings						
Set library	type (and hosting variant) R	edlib (nohost-nf)	~				
Redlib: U	Jse floating point version of p Jse character rather than strin	printf Ig based printf		NewlibNano: Use floatin NewlibNano: Use floatin	g point version of printf g point version of scanf		
Redirect	SDK "PRINTF" to C library "p semihost HardFault handler	rintf"		Redirect printf/scanf to I Redirect printf/scanf to U	TM JART		
• MCU C Co	ompiler tandard GNU C99 (-std=gnu	99)					~
MCU Link	er						
Link app	lication to RAM						
Memory	Configuration						
Memory de	tails						
Type Flash	Name PROGRAM_FLASH	Alias Flash	Location 0x0	Size 0x40000	Driver FTFE_2K.cfx]
Flash	FLEX_NVM	Flash2	0x10000000	0x40000	FTFE_2K_D.cfx		Edit
RAM	SRAM FLEX_RAM	RAM RAM2	0x1fffc000 0x14000000	0x10000 0x2000			
							1
Ð				< Back	Next > Finish		Cancel

Figure 8. . Advanced Settings window

4. Adding FlexCAN and LIN drivers

NXP's KW36 Software Development Kit (SDK) includes FlexCan and LIN drivers as well as demo examples.

Next steps show how to add the FlexCAN and LIN drivers to your Bluetooth Low Energy project using MCUXpresso IDE.

1. Go to "Manage SDK Components" as in Figure 9.

Adding FlexCAN and LIN drivers

웥 Project Explorer 🛛 🛃 Peripherals+ 🕮 Registers 🏝 Symbol Viewer	□☜ ● � ⊠ ▼ ▽ □ □ 🍛
General Strategy S	Manage SDK Components
> 🌾 Project Settings	
> 🔊 Includes	
> 😂 CMSIS	
> 🔒 bluetooth	
> 😕 board	
> 🔑 drivers	<i></i>
> 🔑 framework	
> 😂 freertos	
> 🖴 source	
> 😂 startup	
> 🖴 utilities	
> 🗁 doc	
> 🗁 libs	
Figure 9. Manage SDK Compone	ents

2. Select "flexcan" and "lin" from driver window and click "OK". See Figure 10.

🗙 Manage SDK components for p	project frdmkw36_wireless_examples_	bluetooth_w_uart_freertos	— 🗆 X
			Č.
Available SDK componer	nts		
Copy sources Import other files			
os 🖉 🕅 🖻	driver 🖉 🗹 🙀 🖽 🖻	CMSIS_driver 🖉 🗹 🙀 🖽 🚍	utilities 🛛 🖉 🙀 🕀 🖻
type to filter	type to filter	type to filter	type to filter
Name Version	Name Version ✓ 🕞 dspi 2.1.4 ⑤ edma 2.1.1 ✓ ⑤ flaxan 2.2.0 ✓ ⑤ flexcan 2.2.0 ✓ ⑥ flexcan 2.0 ✓ ⑥ flexcan 2.0 ✓ ⑧ flexcan 2.0 ✓ ⑧ flexcan 2.0 ✓ ⑧ flext 2.0 ✓ ⑧ flext 2.0 ✓ ⑧ flext 2.0 ✓ ⑧ flext 2.0	Name Version CMSIS_Dr 4.5.0 dspi_cmsi 2.0.0 i2c_cmsis 2.0.1 lpuart_cm 2.0.0	Name Version
?			OK Cancel
٠ •			Concer

Figure 10. Driver window

3. A new window will appear to confirm the files which will be added/updated, click "Yes" *Figure* 11.

🔀 SDK Component Management	_		×
The following files will be added or updated if require	red:		
Component source Component source common common fsl_common.c common.c fsl_common.b	Project Path(s) drivers	Info 2.0.0 SRC C_INCLUDE	^
 Iin Iin Iin.c fsl_lin.c fsl_lin_lpuart.c Idevices/MKW36Z4/drivers fsl_lin.h fsl_lin.h fsl_lin_lpuart.h 	drivers	2.0.0 SRC C_INCLUDE	
 ✓ Include_common ✓ -: CMSIS/Include image: arm_common_tables.h image: arm_const_structs.h 	CMSIS	4.5.0 C_INCLUDE	~
Skip add/remove components confirmation in future	Yes	No	

Figure 11. SDK Component Management window

4. A second window will appear to confirm some file to be removed, click "No" or your project will break *Figure* 12.



Figure 12. SDK Component Management window 2

At this point, FlexCAN and LIN drivers are loaded into your Bluetooth LE project.

5. Adding FlexCAN and LIN demo examples into a Bluetooth LE project

NXP's KW36 Software Development Kit (SDK) includes demo examples for FlexCAN and LIN protocols. This demo examples can be imported and modified to include a FlexCAN or/and LIN communication coexisting with your Bluetooth LE Project.

FlexCAN and LIN demo examples are located in "driver_examples" from FRDM-KW36 SDK folder as shown in next figure. Unzip folder to have control in copying the application files.



Figure 13. Driver examples

Next subchapters show how to add a FRDM-KW36 FlexCAN and LIN demo examples into a Bluetooth LE project and how to modify the FlexCAN and LIN application files to coexist with the Bluetooth LE software stack using MCUXpresso.

5.1. FlexCAN

The "interrupt_transfer" demo from the FRDM-KW36 SDK make use of FlexCAN drivers in noneblocking interrupt way. In the example, 2 boards are connected through CAN bus. There is a board A and a board B which can send and receive messages between them.

Follow next steps to add the "interrupt_transfer" demo application and do modifications to coexist with your Bluetooth LE project.

 Locate and copy "flexcan_interrupt_transfer.c" file stored in boards>frdmkw36>driver_examples>flexcan>interrupt_transfer from the unzipped FRDM-KW36 SDK.

ooards > frdmkw36 > driver_examples > flexcan > interrupt_transfer
Name
a) board.c
🧠 board.h
a) clock_config.c
all clock_config.h
🗋 example.xml
Ilexcan_interrupt_transfer.c
flexcan_interrupt_transfer.xml
a) pin_mux.c
a) pin_mux.h
📄 readme.txt

Figure 14. flexcan_interrupt_transfer.c file location

2. Paste the "flexcan_interrupt_transfer.c" file into "source" folder from your MCUXpresso project workspace.

Documents > MCUXpressolDE_10.2.1_795 > workspace > frdmkw36_wireless_examples_bluetooth_w_uart_freertos > source
Name
🦲 common
app_config.c
app_preinclude.h
Ilexcan_interrupt_transfer.c
FreeRTOSConfig.h
🧃 gatt_db.h
🥥 gatt_uuid128.h
MKW36Z512xxx4_connectivity.ld
a) mtb.c
a) wireless_uart.c
a) wireless_uart.h



3. Go to MCUXpresso IDE, click F5 (refresh) and "flexcan_interrupt_transfer.c" file will be added into source folder in your Bluetooth LE project *Figure* 16.



Figure 16. flexcan_interrupt_transfer.c added in MCUXpresso IDE

- 4. Create a function "can_demo_init" to initialize FlexCAN driver:
 - a. In <u>chapter 7</u>, source and header files for a Bluetooth LE to CAN bridge demo can be downloaded to verify all modifications in main function from "flexcan_interrupt_transfer.c" to create the "can_demo_init" function. Check for changes in board A and board B.
 - b. In "flexcan_interrupt_transfer.c", the main function includes driver initialization. It can be modified to change from "main" to "can_demo_init" by cleaning the file and leave only the necessary for CAN driver initialization.
 - c. Be aware that IDs declaration is different for board A and board B ("txIdentifier" and "rxIdentifier").
 - d. "CAN_InitPins" function was created for Bluetooth LE to CAN bridge demo to initialize FlexCAN port module clock and pin multiplexing.
 - e. For easier use, "app_tx_can" and "app_rx_can" are functions created to set FlexCAN module in RX or TX mode.
- 5. Call the "can_demo_init" function to initialize FlexCAN driver in your Bluetooth LE application in "main_task" function from "ApplMain.c" file.
- 6. Additionally, CAN FD frame is set to 8 data bytes in "flexcan_fd_frame_t" declaration from "fsl_flexcan.h". For Bluetooth LE to CAN bridge demo, it was increased to 16 data bytes by modifying "flexcan_fd_frame_t" structure *Figure* 17.

```
struct
        ł
            uint8_t dataByte3;
            uint8_t dataByte2;
            uint8 t dataBytel;
            uint8 t dataByte0;
            uint8 t dataByte7;
            uint8_t dataByte6;
            uint8_t dataByte5;
            uint8 t dataByte4;
            uint8 t dataBytell;
            uint8_t dataByte10;
            uint8 t dataByte9;
            uint8_t dataByte8;
            uint8 t dataByte15;
            uint8 t dataByte14;
            uint8 t dataByte13;
            uint8_t dataByte12;
        };
    };
} flexcan fd frame t;
```

Figure 17. flexcan_interrupt_transfer.c added in MCUXpresso IDE

5.2. LIN

The "lin_master" and "lin_slave" demo applications demonstrate how to use the LIN bus signal to establish a LIN communication between 2 FRDM-KW36 boards.

Follow below steps to add "lin_master" and "lin_slave" demo applications and do modifications to coexist with your Bluetooth LE projects.

 Locate and copy the "lin_master.c", "lin_config.c" and "lin_config.h" files stored in boards>frdmkw36>driver_examples>lin>master from the unzipped FRDM-KW36 SDK *Figure* 18.





2. Paste the "lin_master.c", "lin_config.c" and "lin_config.h" files into "source" folder from your MCUXpresso project workspace *Figure* 19.

MCUXpressoIDE_10.2.1_795 > workspace > frdmkw36_ble_can_lin_bridge_wireless_examples_bluetooth_w_uart_freertos_board_a > source





3. Go to MCUXpresso IDE, click F5 (refresh) and "lin_master.c", "lin_config.c" and "lin_config.h" files will be added into source folder in your Bluetooth LE project *Figure* 20.



Figure 20. LIN master application files added in MCUXpresso IDE

- 4. Repeat steps 1 to 3 for slave node, search for "lin_slave.c", "lin_config.c" and "lin_config.h" files located in boards>frdmkw36>driver_examples>lin>slave from the unzipped FRDM-KW36 SDK.
 - a. Take in consideration that slave node shall be imported into other project which will communicate with master node.

Adding FlexCAN and LIN demo examples into a Bluetooth LE project



Figure 21. LIN slave application files added in MCUXpresso IDE

- 5. Create a function "lin_demo_init_master" to initialize LIN driver in master node:
 - a. In <u>chapter 7</u>, source and header files for a Bluetooth LE to LIN bridge demo can be downloaded to verify all modifications in main function from "lin_master.c" to create the "lin_demo_init_master" function. Check for changes in board A software.
 - b. In "lin_master.c", the main function includes driver initialization. It can be modified to change from "main" to "lin_demo_init_master" by cleaning the file and leave only the necessary for LIN driver initialization.
 - c. Change linCurrentState and linUserConfigMaster structures from local variables as global variables in "lin_master.c" file.
- 6. Create a function "lin_demo_init_slave" to initialize LIN driver in slave node:
 - a. In <u>chapter 7</u>, source and header files for a Bluetooth LE to LIN bridge demo can be downloaded to verify all modifications in main function from "lin_slave.c" to create the "lin_demo_init_slave" function. Check for changes in board B software.
 - b. In "lin_slave.c", the main function includes driver initialization. It can be modified to change from "main" to "lin_demo_init_slave" by cleaning the file and leave only the necessary for LIN driver initialization.
 - c. Change linCurrentState and linUserConfigSlave structures from local variables as global variables in "lin_slave.c" file.
- 7. "BOARD_InitLinPpuart" function was created for Bluetooth LE to LIN bridge demo to initialize LIN port module clock and pin multiplexing.
- 8. Call the "lin_demo_init_master" function to initialize LIN module in your Bluetooth LE application in "main_task" function from "ApplMain.c" file for master node.
- 9. Call the "lin_demo_init_slave" function to initialize LIN module in your Bluetooth LE application in "main_task" function from "ApplMain.c" file for slave node.
- 10. The adding of "lin_master.c" or "lin_slave.c" to a BLE project causes duplication on LPUART0_LPUART1_IRQHandler. To fix it, follow next steps:

a. Search and delete the code line:

#define DEMO_LIN_IRQHandler LPUART0_LPUART1_IRQHandler

- b. Search and delete DEMO_LIN_IRQHandler definition.
- c. Modify LPUART0_LPUART1_IRQHandler function from UART_Adapter.c to add the LIN_IRQHandler in instance 1 validation Figure 22



Figure 22. LPUART0_LPUART1_IRQHandler function

6. Adding AES 128 security to FlexCAN data frames

AES 128 security can be added into the FlexCAN data frames.

Follow next steps to implement AES 128 security to FlexCAN data frames.

1. Add security library.

a. #include "SecLib.h"

2. Create the AES 128 key.

```
uint8_t key128[128/8] = { 0x55, 0x6c, 0x74, 0x72,
0x61, 0x53, 0x65, 0x63,
0x61, 0x73, 0x73, 0x77,
0x6f, 0x72, 0x64, 0x31 }; /*AES128 key*/
```

Figure 23. AES 128 Key

3. Use "AES_128_Encrypt" function to encrypt data to TX, and use "AES_128_Decrypt" function to decrypt RX data.

Bluetooth LE-CAN-LIN Bridge Demo uses AES 128 security in the FlexCAN data frames. Source and header files can be downloaded in next chapter.

7. Bluetooth LE-CAN-LIN Bridge Demo

This chapter explains the setup, behavior and usage of the Bluetooth LE-CAN-LIN bridge demo.

A link to download the source and header files of the demo can be found in <u>section 7.3.1</u> as well as explanation on how to run the demo.

7.1. Overview

Bluetooth LE-CAN-LIN bridge demo is based on the Wireless UART Demo for FRDM-KW36 where CAN and LIN bridges were added to communicate with other FRDM-KW36 board. This application is intended to demonstrate KW36 features in a scenario where a Bluetooth LE+CAN+LIN bridge is needed.

The demo consists of two FRDM-KW36 boards. Board A is connected through Bluetooth LE to the NXP mobile application "IoT Toolbox". Board A is also connected to a FRDM-KW36 board B through CAN and LIN.

A list of predefined commands can be used in the Wireless Console/UART application of the NXP IoT Toolbox and sent to board A via Bluetooth LE. User can select if command is sent by CAN or LIN from board A to board B which decodes the command and performs a specific action. Board B can also send commands via CAN or LIN to board A which transmits the data via Bluetooth LE to smartphone and a message is displayed in Wireless Console/UART app.

Figure 24 shows the data flow of the demo in a specific case where a toggle led command was sent by CAN from board A to board B. **Figure 25** shows the inverse behavior. This way, board A works as a bridge.



Figure 24. Command flow from board A to board B

Bluetooth LE-CAN-LIN Bridge Demo



7.2. Prerequisites

The following items are needed for this demo:

- Boards
 - o 1 FRDM-KW36 with "Bluetooth LE-CAN-LIN Bridge Demo Board A" firmware loaded
 - 1 FRDM-KW36 with "Bluetooth LE-CAN-LIN Bridge Demo Board B" firmware loaded
- Software
 - 0 1 smartphone with NXP IoT Toolbox app installed
 - Available in iOS and Android App stores
 - Android 4.4 and IOS 9 as minimum versions.
 - MCUXpresso v10.2.1 or later
- Tools
 - \circ 12V DC source
 - \circ 5 female to female jumper wires

7.3. Set Up

This section explains the steps to set up software, hardware and application.

7.3.1. Software

The Bluetooth LE-CAN-LIN demo can be loaded into FRDM-KW36 boards by using binaries files as explained in section 7.3.1.1 or other option is by creating a new Wireless UART project in MCUXpresso and replace the source and header files modified for the demo as explained in section 7.3.1.2.

Bluetooth LE-CAN-LIN demo binaries, and source and header files can be downloaded from this link: https://www.nxp.com/webapp/sps/download/license.jsp?colCode=AN12273-SW&Parent_nodeId=1525379522858711408521&Parent_pageType=product

7.3.1.1. Binaries

Follow next steps to load the application firmware into the two FRDM-KW36 boards by using binaries files:

- 1. Connect a FRDM-KW36 to your PC via USB port.
- 2. Drag and drop "bluetooth_le_can_lin_bridge_a.bin" stored in "bin" folder from AN12273SW to FRDM-KW36 disk. Firmware will be downloaded automatically.



Figure 26. Drag and drop board A binary file

- 3. Disconnect the FRDM-KW36 board A and connect other FRDM-KW36 to your PC by USB port.
- 4. Drag and drop "bluetooth_le_can_lin_bridge_b.bin" stored in "bin" folder from downloaded AN12273SW to FRDM-KW36 disk. Firmware will be downloaded automatically.



Figure 27. Drag and drop board B binary file

7.3.1.2. Source and header files

Follow next steps to load the application firmware into the two FRDM-KW36 boards by creating a new wireless uart demo project in MCUXpresso and replacing the source and header files modified for the demo:

- 1. Follow <u>chapter 3</u> to create two new Wireless Uart projects (FreeRTOS) for MCUXpresso.
- 2. Follow <u>chapter 4</u> to add FlexCAN and LIN drivers into your 2 projects.
- 3. Copy and paste the modified source and header files for the demo stored in "board a" folder from downloaded AN12273SW into your new Wireless Uart project for board a stored in your MCUXpresso workspace folder.
 - a. Location: "Your project name" \source
 - i. Files:
 - lin_config.c
 - lin_config.h
 - lin_slave.c
 - wireless_uart.c
 - wireless_uart.h
 - flexcan_interrupt_transfer.c
 - b. Location: "Your project name" \source\common
 - i. Files:
 - ApplMain.c
 - c. Location: "Your project name" \board
 - i. Files:
 - pin_mux.c

- pin_mux.h
- d. Location: "Your project name" \drivers
 - i. Files:
 - fsl_flexcan.h
- e. Location: "Your project name" \framework\SerialManager\Source\UART_Adapter
 - i. Files:
 - UART_Adapter.c
- 4. Copy and paste the modified source and header files for the demo stored in "board b" folder from downloaded AN12273SW into your new Wireless Uart project for board b stored in your MCUXpresso workspace folder.
 - a. Location: "Your project name" \source
 - i. Files:
 - lin_config.c
 - lin_config.h
 - lin_master.c
 - wireless_uart.c
 - wireless_uart.h
 - flexcan_interrupt_transfer.c
 - b. Location: "Your project name" \source\common
 - i. Files:
 - ApplMain.c
 - c. Location: "Your project name" \board
 - i. Files:
 - pin_mux.c
 - pin_mux.h
 - board.h
 - d. Location: "Your project name" \drivers
 - i. Files:
 - fsl_flexcan.h
 - e. Location: "Your project name" \framework\SerialManager\Source\UART_Adapter i. Files:
 - UART_Adapter.c
- 5. Press "F5" on MCUXpresso IDE to refresh your project. The new files will be added.
- 6. Your 2 projects are ready to be compiled and run into your FRDM-KW36 board A and your FRDM-KW36 board B.

7.3.2. Hardware

1. Connect pins between FRDM-KW36 board A and FRDM-KW36 board B.

- CAN connector:
 - J23: pins 1-2 (CANH, CANL)
- Lin connector:
 - J13: pin 1
- Power:
 - J23: pins 3-4 (P12V, GND)
 - Or
 - J13: pins 2-4 (P12V, GND)
- Connect the 12V DC source to one of the boards on J32. Connections are shown in *Figure* 28. LED3 and LED 4 will start blinking red and white on FRDM-KW36 board A.



Figure 28. Connections between boards

7.3.3. Application

- 1. To start scanning, press SW2 button in FRDM-KW36 board A, then LED3 is flashing.
- 2. Open the NXP IoT Toolbox in one smartphone supporting BLE and select Wireless Console/UART option shown in *Figure* 29.



3. Connect the smartphone to FRDM-KW36 A. When the node connects to a peer device, LED3 turns solid.





Figure 30. Connect to NXP IoT Toolbox

4. Once connected, a welcome message is printed as shown in *Figure* 31.

HI TELCEL 🗢 16:32						1	▼ \$ 92% (***)		
<			loT	Tool	box				ECT
Wireless Console									
BLE-CAN-LIN Bridge Demo Copyright (c) 2018 NXP Semiconductor Type "help" to see all registered commands									
Q	W	Е	R	т	Y	U	1	0	Ρ
Α	S	D	F	G	Н	J	к	L	Ñ
+	Z	x			/ [з Г	N	N	\bigotimes
123) Q		e	spaci	ю		Envi	ar

Figure 31. Welcome menu

7.3.4. Usage

This section explains how to run the application.

1. Write and send a command using the Wireless Console/UART connected to FRDM-KW36 board A.

Table 1 shows the complete list of available commands to write and send using NXP IoT Toolbox app, and Table 2 shows description of using switch buttons on board B.

NOTE

Commands are case sensitive therefore assure they are written in lowercase.



Figure 32. Command sent for BLE+CAN Demo

Demo Commands	Description
help	Print available commands list on app.
lin on	Send LIN wake up signal to board A
lin off	Send LIN go to sleep signal to board A
lin toggle led red	Toggle RGB red led on board B via LIN
lin toggle led green	Toggle RGB red green on board B via LIN
lin toggle led blue	Toggle RGB red blue on board B via LIN
lin get temperature	Get board B temperature via LIN. It's printed in app.
lin get led	Get the status (0/1) of RGB led colors (red/green/blue) of board B via LIN. It's printed in app.
can toggle led red	Toggle RGB red led on board B via CAN
can toggle led green	Toggle RGB red green on board B via CAN
can toggle led blue	Toggle RGB red blue on board B via CAN
can get temperature	Get board B temperature via CAN. It's printed in app.
can get led	Get the status (0/1) of RGB led colors (red/green/blue) of board B via CAN. It's printed in app.

Table 1. BLE+CAN+LIN Demo Commands

2. Press SW2 or SW3 in FRDM-KW36 board B and a message on NXP IoT Toolbox app. *Table* 2 shows a brief description.

FRDM-KW36 board B actions	Description
Press SW2	Press SW2 in board B and a message will
	be sent to board A via CAN and it will be
	displayed in Wireless Console/UART.
Press SW3	Press SW2 in board B and a message will
	be sent to board A via LIN and it will be
	displayed in Wireless Console/UART.

Table 2. Board B Switch Buttons

7.4. Software Modifications

This section explains some software modifications that were implemented for the Bluetooth LE-CAN-LIN Bridge Demo.

• The demo is based on the Wireless UART demo. The device role is set to central as default in this demo. To establish connection with the cellphone, the device role needs to be changed to peripheral by default in BleApp_Config function. This change was only implemented for board A. See Figure 33.

```
static void BleApp_Config()
   uint8_t mPeerId = 0;
fifdef MULTICORE HOST
   if (GattDbDynamic_CreateDatabase() != gBleSuccess_c)
        panic(0,0,0,0);
        return;
#endif /* MULTICORE HOST */
    /* Configure as GAP Dual Role */
    BleConnManager GapDualRoleConfig();
    /* Register for callbacks */
    App_RegisterGattServerCallback(BleApp_GattServerCallback);
    App_RegisterGattClientProcedureCallback(BleApp_GattClientCallback);
    GattServer RegisterHandlesForWriteNotifications (NumberOfElements (mCharMonitoredHandles
    BleServDisc_RegisterCallback(BleApp_ServiceDiscoveryCallback);
    for(mPeerId = 0; mPeerId < gAppMaxConnections_c; mPeerId++)
        maPeerInformation[mPeerId].appState = mAppIdle_c;
       maPeerInformation[mPeerId].deviceId = gInvalidDeviceId_c;
        maPeerInformation[mPeerId].clientInfo.hService = gGattDbInvalidHandleIndex_d;
        maPeerInformation[mPeerId].clientInfo.hUartStream = gGattDbInvalidHandleIndex_d;
    ł
    /* start node as GAP peripheral */
    mGapRole = gGapPeripheral c:
   mAdvState.advOn = FALSE;
   mScanningOn = FALSE:
    /* Start services */
    Wus_Start(smWuServiceConfig);
   mBasServiceConfig.batteryLevel = BOARD GetBatteryLevel();
   Bas_Start(smBasServiceConfig);
    /* Allocate application timer */
    mAppTimerId = TMR_AllocateTimer();
    mErrorCommandTimerID = TMR_AllocateTimer(); //allocate timer for error command
```

Figure 33. Peripheral device as default

 In board A software, "APP_ProcessString" function was created to process the string received from the IOT Toolbox via Bluetooth LE. It's called in "BleApp_ReceivedUartStream" where "APP_ProcessString" replaced "Serial_AsyncWrite" function call for wireless uart application.

- "lin_go_to_sleep" and "check_if_lin_sleeping" functions were created to send LIN bus to sleep and check when it should wake up.
- Some modifications on LIN configuration files were done in "lin_config.c" and "lin_config.h" for master and slave
 - a. lin_frame_tbl was modified to support the desired frames for application.
 - b. LI0_lin_configuration_RAM and LI0_lin_configuration_ROM were filled with ID's specifics for this application.
- Extra frames for LIN communication were added on "lin_frame_tbl", and number of frames on "number_of_configurable_frames" from "g_lin_protocol_user_cfg_array" and "LIN_NUM_OF_FRMS".
- "APP_ProcessCommand_Lin" and "APP_ProcessCommand_Can" are functions to handle application commands. Those functions are declared and called on "flexcan_interrupt_transfer.c" and "lin_master.c" or "lin_slave.c" and defined on "wireless_uart.c".

8. Revision history

Revision number	Date	Substantive changes
0	10/2018	Initial release.
1	11/2018	General updates

Table 3. Revision history

How to Reach Us:

Home Page: nxp.com

Web Support: nxp.com/support Information in this document is provided solely to enable system and software implementers to use NXP products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document. NXP reserves the right to make changes without further notice to any products herein.

NXP makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does NXP assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters that may be provided in NXP data sheets and/or specifications can and do vary in different applications, and actual performance may vary over time. All operating parameters, including "typicals," must be validated for each customer application by customer's technical experts. NXP does not convey any license under its patent rights nor the rights of others. NXP sells products pursuant to standard terms and conditions of sale, which can be found at the following address: nxp.com/SalesTermsandConditions.

While NXP has implemented advanced security features, all products may be subject to unidentified vulnerabilities. Customers are responsible for the design and operation of their applications and products to reduce the effect of these vulnerabilities on customer's applications and products, and NXP accepts no liability for any vulnerability that is discovered. Customers should implement appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP, the NXP logo, NXP SECURE CONNECTIONS FOR A SMARTER WORLD, COOLFLUX, EMBRACE, GREENCHIP, HITAG, I2C BUS, ICODE, JCOP, LIFE VIBES, MIFARE, MIFARE CLASSIC, MIFARE DESFire, MIFARE PLUS, MIFARE FLEX, MANTIS, MIFARE ULTRALIGHT, MIFARE4MOBILE, MIGLO, NTAG, ROADLINK, SMARTLX, SMARTMX, STARPLUG, TOPFET, TRENCHMOS, UCODE, Freescale, the Freescale logo, AltiVec, C 5, CodeTEST, CodeWarrior, ColdFire, ColdFire+, C Ware, the Energy Efficient Solutions logo, Kinetis, Layerscape, MagniV, mobileGT, PEG, PowerQUICC, Processor Expert, QorlQ, QorlQ Qonverge, Ready Play, SafeAssure, the SafeAssure logo, StarCore, Symphony, VortiQa, Vybrid, Airfast, BeeKit, BeeStack, CoreNet, Flexis, MXC, Platform in a Package, QUICC Engine, SMARTMOS, Tower, TurboLink, and UMEMS are trademarks of NXP B.V. All other product or service names are the property of their respective owners. Arm, AMBA, Arm Powered, Artisan, Cortex, Jazelle, Keil, SecurCore, Thumb, TrustZone, and µVision are registered trademarks of Arm Limited (or its subsidiaries) in the EU and/or elsewhere. Arm7, Arm9, Arm11, big.LITTLE, CoreLink, CoreSight, DesignStart, Mali, Mbed, NEON, POP, Sensinode, Socrates, ULINK and Versatile are trademarks of Arm Limited (or its subsidiaries) in the EU and/or elsewhere. All rights reserved. Oracle and Java are registered trademarks of Oracle and/or its affiliates. The Power Architecture and Power.org word marks and the Power and Power.org logos and related marks are trademarks and service marks licensed by Power.org.

© 2018 NXP B.V.

Document Number: AN12273 Rev. 1 11/2018



arm