CE215123 - BLE Location and Navigation with PSoC 6 MCU with BLE Connectivity

Objective

This example project demonstrates the Location and Navigation Pod application workflow.

Overview

The design demonstrates the core functionality of the Bluetooth Low Energy (BLE) Component configured as a BLE Location and Navigation Service (LNS) device in the GATT Server role. The application uses a BLE Location and Navigation Profile to report location and navigation information to a Client. Also, the Location and Navigation Pod application uses the Battery Service to notify the battery level and the Device Information Service to assert the Device Name and so on.

Requirements

Tool: PSoC Creator™ 4.2 or later

Programming Language: C (Arm® GCC 5.4-2016-q2-update or later)

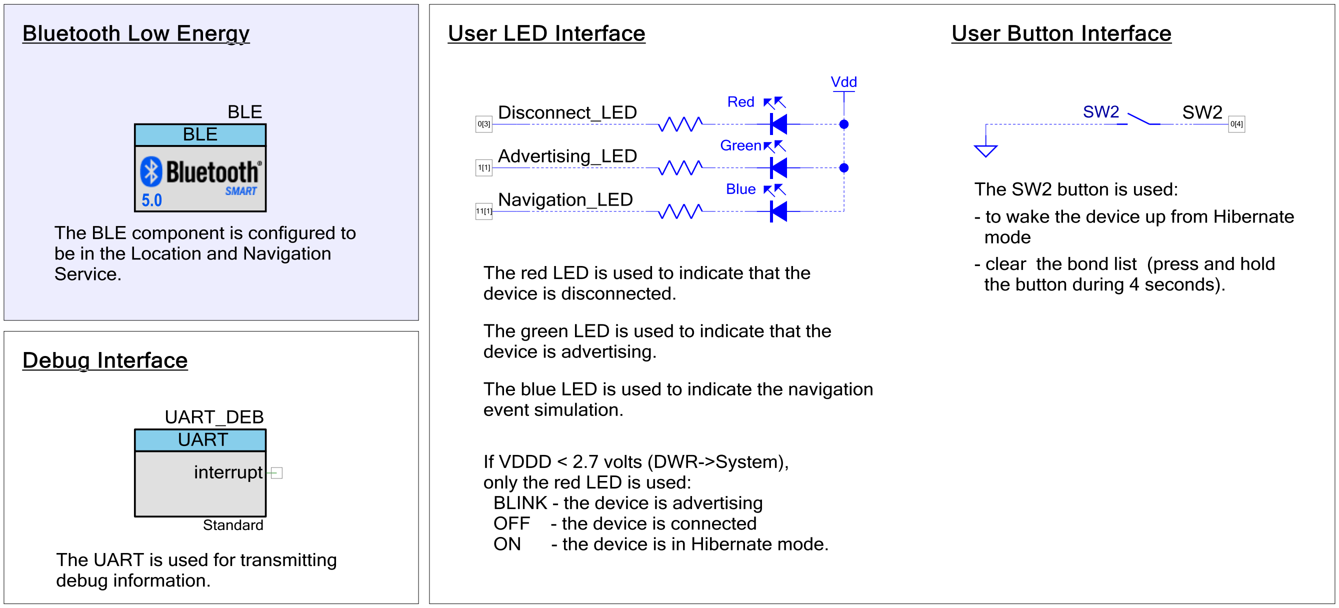
Associated Parts: All PSoC® 6 MCU with BLE Connectivity (PSoC 6 BLE) parts

Related Hardware: CY8CKIT-062 PSoC 6 BLE Pioneer Kit

Design

Figure 1 shows the top design schematic.

Figure 1. BLE Location and Navigation Top Design Schematic



The BLE component is configured to be in the Location and Navigation Service.

The UART is used for transmitting debug information.

The red LED is used to indicate that the device is disconnected.

The green LED is used to indicate that the device is advertising.

The blue LED is used to indicate the navigation event simulation.

If VDDD < 2.7 volts (DWR->System), only the red LED is used:
BLINK - the device is advertising
OFF - the device is connected
ON - the device is in Hibernate mode.

The SW2 button is used:

- to wake the device up from Hibernate mode
- clear the bond list (press and hold the button during 4 seconds).

The project demonstrates the core functionality of the BLE Component configured as a Location and Navigation Server.

After a start, the device performs the BLE Component initialization. In this project, three callback functions are required for the BLE operation. Callback function `AppCallBack()` is required to receive generic events from the BLE Stack and the service-specific callbacks `BasCallBack()` and `LnsCallBack()` are required for Battery and LNS service-specific events accordingly. The `CY_BLE_EVT_STACK_ON` event indicates successful initialization of the BLE Stack. After this event is received, the component starts advertising with the packet structure as configured in the BLE Component Customizer. The BLE Component stops advertising as soon as a 180-second advertising period expires.

The Location and Navigation Pod device can be connected to any BLE (4.0 or later) compatible device configured as the GAP Central role and GATT Client which supports the Location and Navigation Profile. The Battery and Device Information services may be optionally used. To connect to the Location and Navigation Pod device, send a connection request to the device while the device is advertising. The green LED blinks while the device is advertising. The red LED is turned ON after disconnection to indicate that no client is connected to the device. When a client connects successfully, the red and green LEDs are turned OFF. If the client is connected to the Location and Navigation Pod, the Location and Speed characteristic notifications can be enabled and then the device will simulate some location changes and notify the Location and Speed characteristic. The blue LED blinks to indicate a simulated navigation data transfer to the client.

The following conditions should be met if the Navigation characteristic is supported to receive its notifications:

- Configure the Navigation CCCD to enable the notification.
- Enable the Location and Navigation Control Point indication.
- Write the **Start navigation** command in the Location and Navigation Control Point characteristic.

For details, see the Location and Navigation Profile and Location and Navigation Service specifications adopted by Bluetooth SIG. The BLE Stack timer is used to time the simulations and LED blinking.

While connected to the Client and between connection intervals, the device is put into Low-Power mode.

Design Considerations

Using UART for Debugging

Download and install a serial port communication program. Freeware such as Bray’s Terminal, and PuTTY are available on the web.

1. Connect the PC and kit with a USB cable.
2. Open the device manager program in your PC, find a COM port that the kit is connected to, and note the port number.
3. Open the serial port communication program and select the previously noted COM port.
4. Configure the Baud rate, Parity, Stop bits, and Flow control information in the PuTTY configuration window. The default settings: Baud rate – 115200, Parity – None, Stop bits – 1, Flow control – XON/XOFF. These settings must match the configuration of the PSoC Creator UART component in the project.
5. Start communicating with the device as explained in the Operation section.

The UART debugging can be disabled by setting the `DEBUG_UART_ENABLED` to `DISABLED` in the *common.h* file.

Switching the CPU Cores Usage

This section describes how to switch between different CPU cores usage (Single core and Dual core) in the BLE Peripheral Driver Library (PDL) examples.

The BLE Component has the CPU Core parameter that defines the cores usage. It can take the following values:

- **Single core (Complete Component on CM0+)** – only CM0+ core will be used.
- **Single core (Complete Component on CM4)** – only CM4 core will be used.
- **Dual core (Controller on CM0+, Host and Profiles on CM4**) – both cores will be used: CM0+ for the Controller and CM4 for the Host and Profiles.

The BLE examples' structure allows easy switching between different CPU cores options.

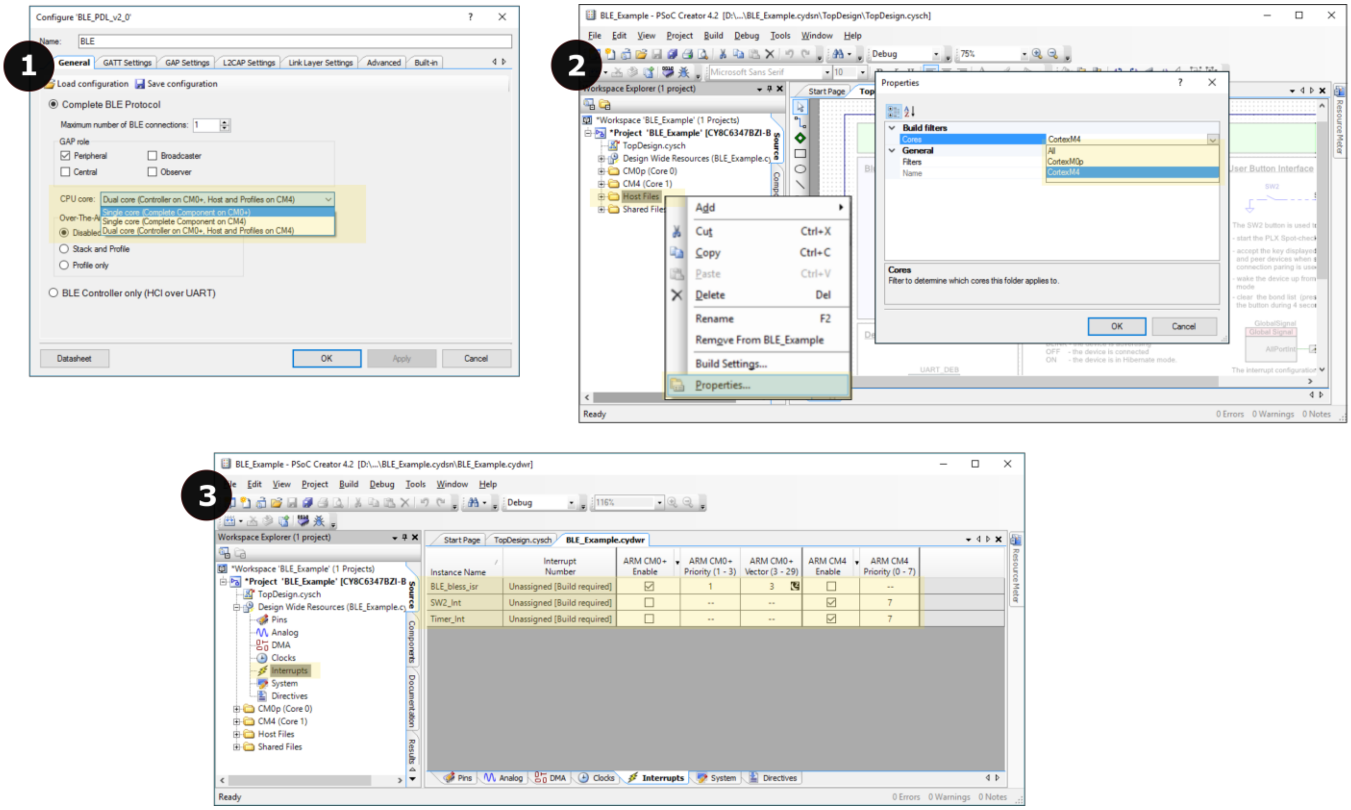
Important to remember:

- All application host-files must be run on the host core.
- The BLE Subsystem (BLESS) interrupt must be assigned to the core where the controller runs.
- All additional interrupts (SW2, MCWDT, etc.) used in the example must be assigned to the host core.

Do the following to switch the CPU cores usage:

6. In the BLE Component Customizer **General** tab, select appropriate CPU core option.
7. Change the core properties to CortexM4 or CortexC0p for the project folder Host Files based on the CPU core option selected in step 1. It should be:
   - For **Single core (Complete Component on CM0+)** option: CM0+
   - For **Single core (Complete Component on CM4)** option: **CM4**
   - For **Dual core (Controller on CM0+, Host and Profiles on CM4)** option: **CM4**
8. Assign the BLE_bless_isr and other peripheral (button – SW2, timer(s) etc.) interrupts to appropriate core in DWR-> interrupts tab:
   - For **Single core (Complete Component on CM0+)** option: BLE_bless_isr and peripheral interrupts on **CM0+**
   - For **Single core (Complete Component on CM4)** option: BLE_bless_isr and peripheral interrupts on **CM4**
   - For **Dual core (Controller on CM0+, Host and Profiles on CM4)** option: BLE_bless_isr interrupt on **CM0+,** other peripheral interrupts on **CM4**

Figure 2. Steps for Switching the CPU Cores Usage



Name: BLE

● Complete BLE Protocol

Maximum number of BLE connections: 1

GAP role

☑ Peripheral ☐ Broadcaster

☐ Central ☐ Observer

CPU core: Dual core (Controller on CM0+, Host and Profiles on CM4)

Single core (Complete Component on CM0+)
Single core (Complete Component on CM4)
Dual core (Controller on CM0+, Host and Profiles on CM4)

Over-The-Ai

● Disabled

○ Stack and Profile

○ Profile only

○ BLE Controller only (HCI over UART)

Cores: CortexM4

All
CortexM0p
CortexM4

Instance Name	Interrupt Number	ARM CM0+ Enable	ARM CM0+ Priority (1 - 3)	ARM CM0+ Vector (3 - 29)	ARM CM4 Enable	ARM CM4 Priority (0 - 7)
BLE_bless_isr	Unassigned [Build required]	☑	1	3	☐	--
SW2_Int	Unassigned [Build required]	☐	--	--	☑	7
Timer_Int	Unassigned [Build required]	☐	--	--	☑	7

Hardware Setup

The code example was designed for the CY8CKIT-062 PSoC 6 BLE Pioneer Kit.

Table 1 lists the pin assignments and connections required on the development board for supported kits..

Table 1. Pin Assignment

Pin Name	Development Kit	Comment
	CY8CKIT-062	
\UART_DEB:rx\	P5[0]	
\UART_DEB:tx\	P5[1]	
\UART_DEB:rts\	P5[2]	
\UART_DEB:cts\	P5[3]	
Advertising_LED	P1[1]	The green color of the RGB LED
Disconnect_LED	P0[3]	The red color of the RGB LED
Navigation_LED	P11[1]	The blue color of the RGB LED
SW2	P0[4]	

LED Behavior for VDDD Voltage < 2.7 V

If the V_{DDD} voltage is set to less than 2.7 V in the DWR settings of the **System** tab, only the red LED is used. The red LED blinks to indicate that the device is advertising. The red LED is OFF when the device is connected to a peer device. When the device is in Hibernate mode, the red LED stays ON.

Components

Table 2 lists the PSoC Creator Components used in this example as well as the hardware resources used by each of the components.

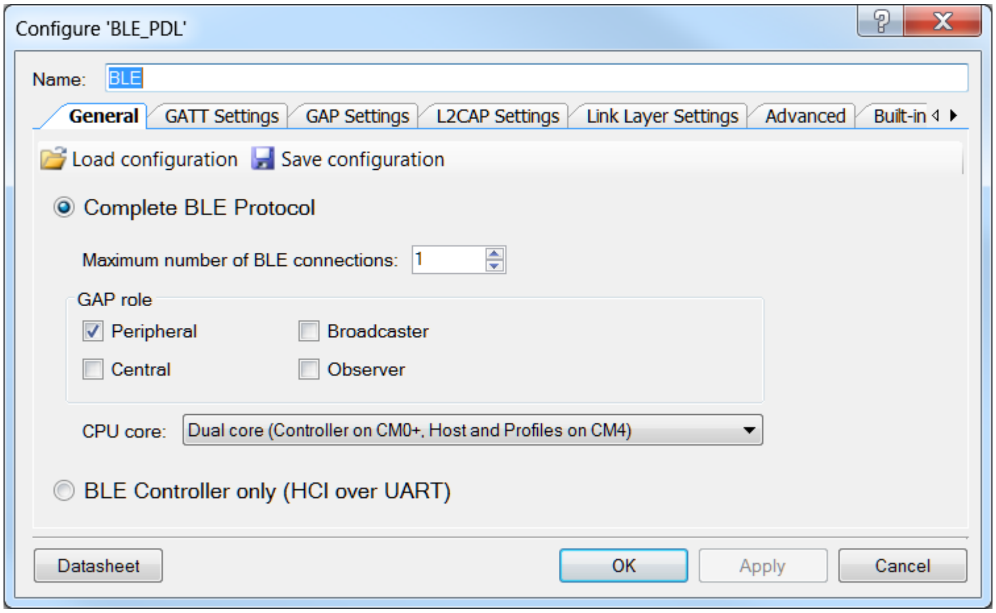
Table 2. PSoC Creator Components List

Component	Hardware Resources
UART_DEB	1 SCB
BLE	1 BLE, 1 Interrupt
SW2	1 pin
Disconnect_LED, Advertising_LED, Navigation _LED	3 pins

Parameter Settings

The BLE Component is configured as a Location and Navigation Server in the GAP Peripheral role. Also, the Battery and Device Information Services are included.

Figure 3. General Settings



Name: BLE

● Complete BLE Protocol

Maximum number of BLE connections: 1

GAP role

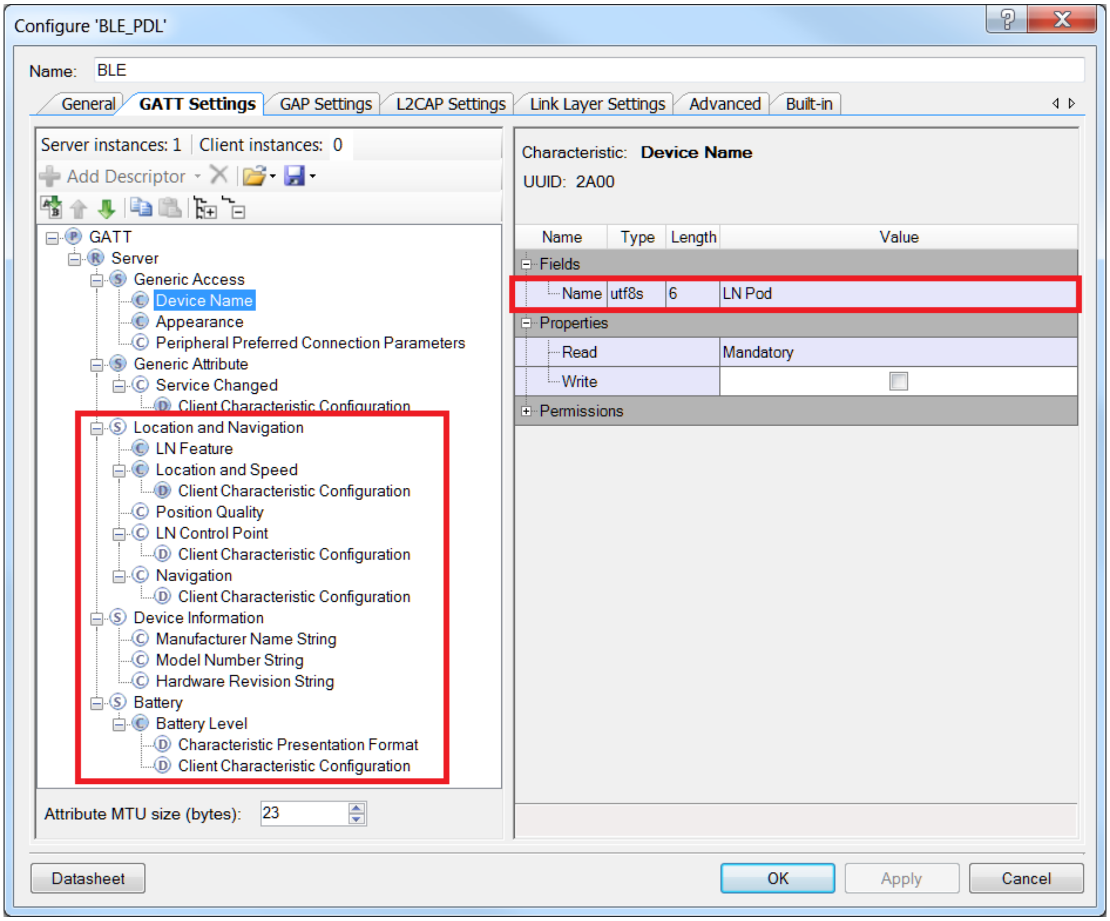
☑ Peripheral ☐ Broadcaster

☐ Central ☐ Observer

CPU core: Dual core (Controller on CM0+, Host and Profiles on CM4)

○ BLE Controller only (HCI over UART)

Figure 4. GATT Settings



Name: BLE

Server instances: 1 Client instances: 0

- GATT
  - Server
    - Generic Access
      - Device Name
      - Appearance
      - Peripheral Preferred Connection Parameters
    - Generic Attribute
      - Service Changed
        - Client Characteristic Configuration
    - Location and Navigation
      - LN Feature
      - Location and Speed
        - Client Characteristic Configuration
      - Position Quality
      - LN Control Point
        - Client Characteristic Configuration
      - Navigation
        - Client Characteristic Configuration
    - Device Information
      - Manufacturer Name String
      - Model Number String
      - Hardware Revision String
    - Battery
      - Battery Level
        - Characteristic Presentation Format
        - Client Characteristic Configuration

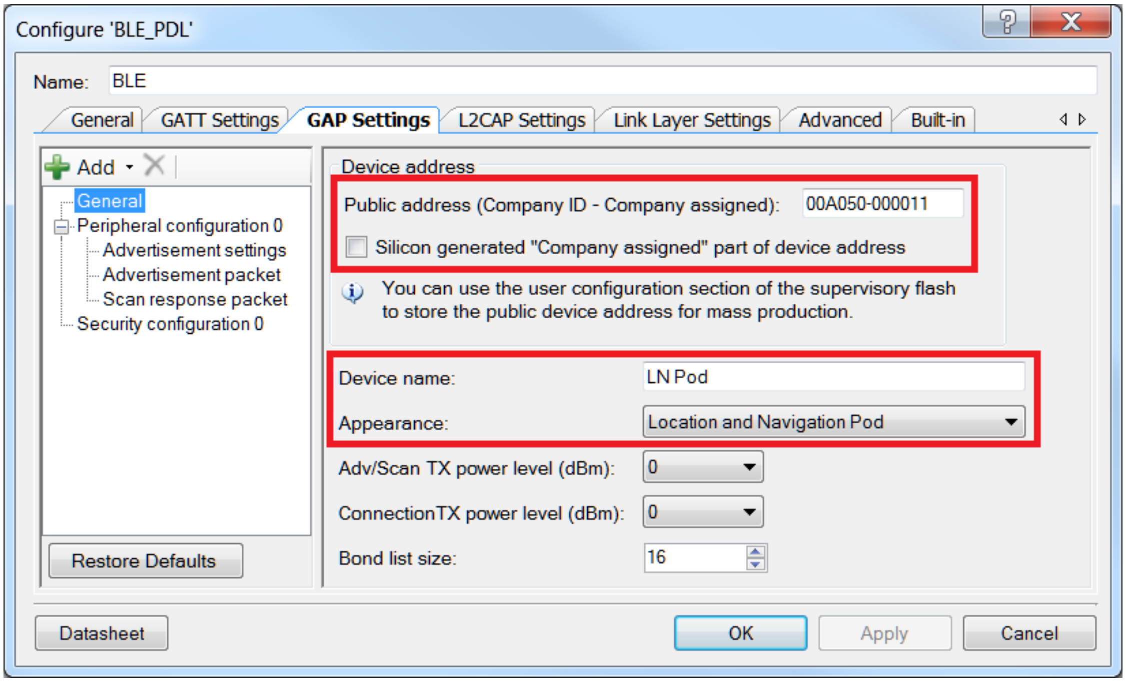
Attribute MTU size (bytes): 23

Characteristic: **Device Name**

UUID: 2A00

Name	Type	Length	Value
Fields			
Name	utf8s	6	LN Pod
Properties			
Read			Mandatory
Write			☐
Permissions			

Figure 5. GAP Settings



Name: BLE

Device address

Public address (Company ID - Company assigned): 00A050-000011

□ Silicon generated "Company assigned" part of device address

You can use the user configuration section of the supervisory flash to store the public device address for mass production.

Device name: LN Pod

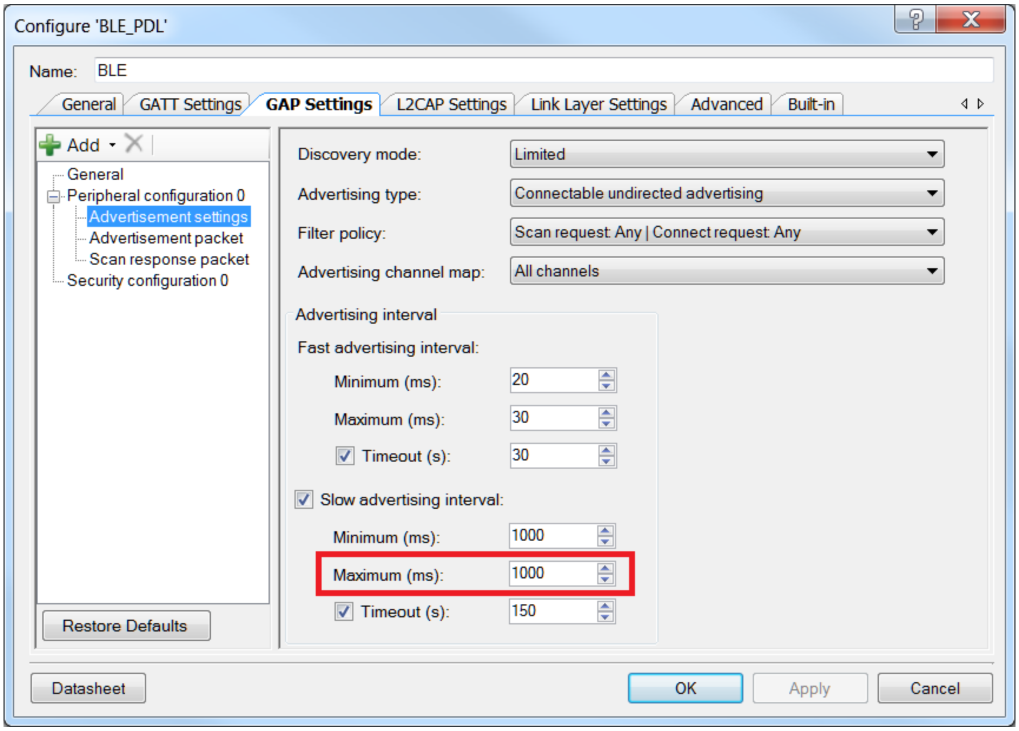
Appearance: Location and Navigation Pod

Adv/Scan TX power level (dBm): 0

ConnectionTX power level (dBm): 0

Bond list size: 16

Figure 6. GAP Settings: Advertisement Settings



Name: BLE

Discovery mode: Limited

Advertising type: Connectable undirected advertising

Filter policy: Scan request: Any | Connect request: Any

Advertising channel map: All channels

Advertising interval

Fast advertising interval:

Minimum (ms): 20

Maximum (ms): 30

☑ Timeout (s): 30

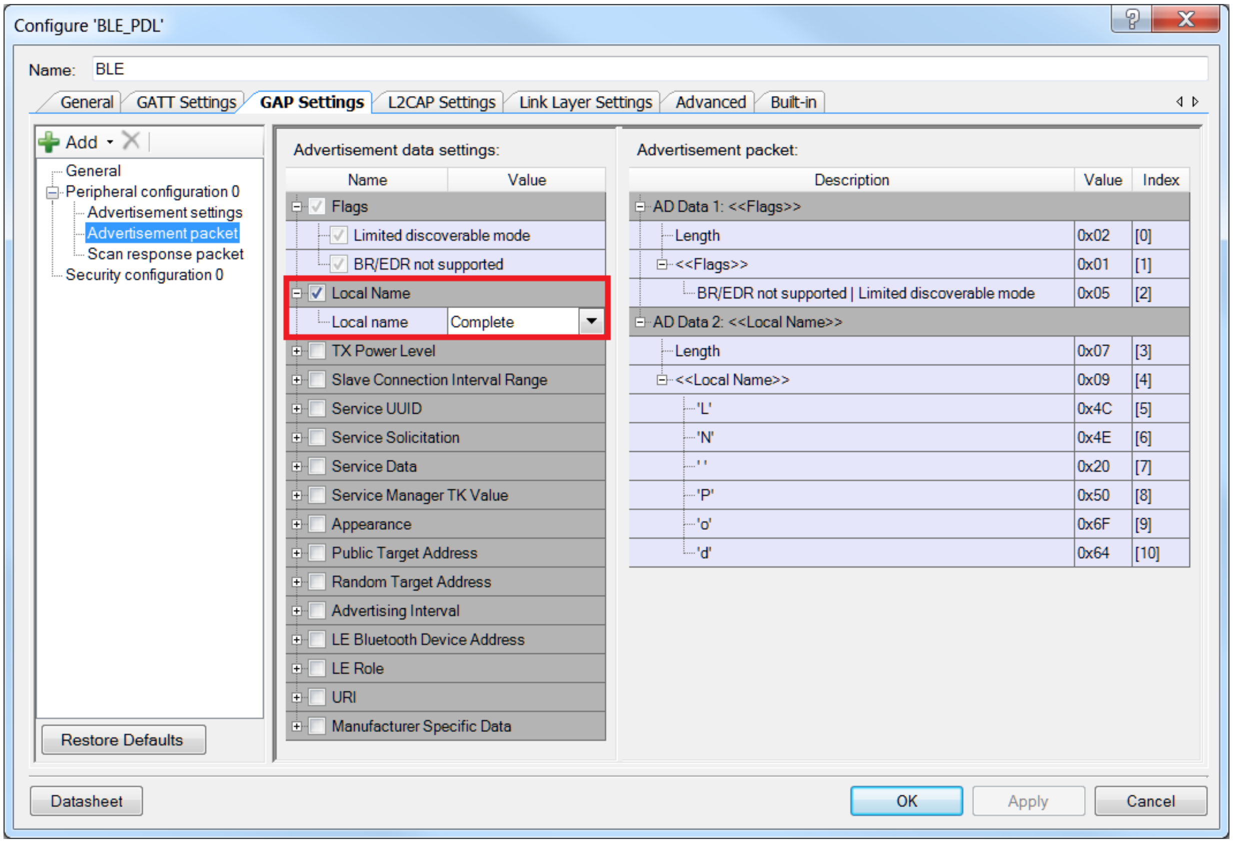
☑ Slow advertising interval:

Minimum (ms): 1000

Maximum (ms): 1000

☑ Timeout (s): 150

Figure 7. GAP Settings: Advertisement Packet



Name: BLE

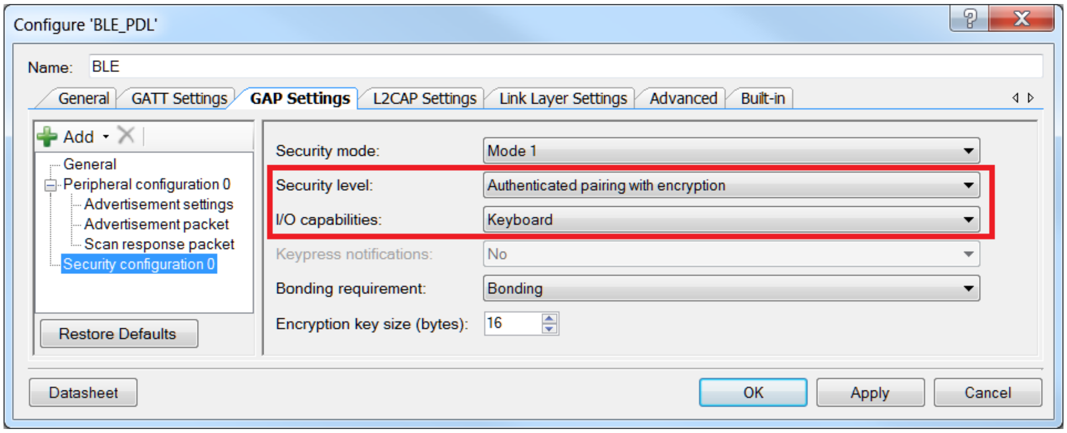
Advertisement data settings:

Name	Value
☑ Flags	
☑ Limited discoverable mode	
☑ BR/EDR not supported	
☑ Local Name	
Local name	Complete
□ TX Power Level	
□ Slave Connection Interval Range	
□ Service UUID	
□ Service Solicitation	
□ Service Data	
□ Service Manager TK Value	
□ Appearance	
□ Public Target Address	
□ Random Target Address	
□ Advertising Interval	
□ LE Bluetooth Device Address	
□ LE Role	
□ URI	
□ Manufacturer Specific Data	

Advertisement packet:

Description	Value	Index
AD Data 1: <<Flags>>		
Length	0x02	[0]
<<Flags>>	0x01	[1]
BR/EDR not supported | Limited discoverable mode	0x05	[2]
AD Data 2: <<Local Name>>		
Length	0x07	[3]
<<Local Name>>	0x09	[4]
'L'	0x4C	[5]
'N'	0x4E	[6]
' '	0x20	[7]
'P'	0x50	[8]
'o'	0x6F	[9]
'd'	0x64	[10]

Figure 8. Security Settings



Name: BLE

Security mode: Mode 1

Security level: Authenticated pairing with encryption

I/O capabilities: Keyboard

Keypress notifications: No

Bonding requirement: Bonding

Encryption key size (bytes): 16

Operation

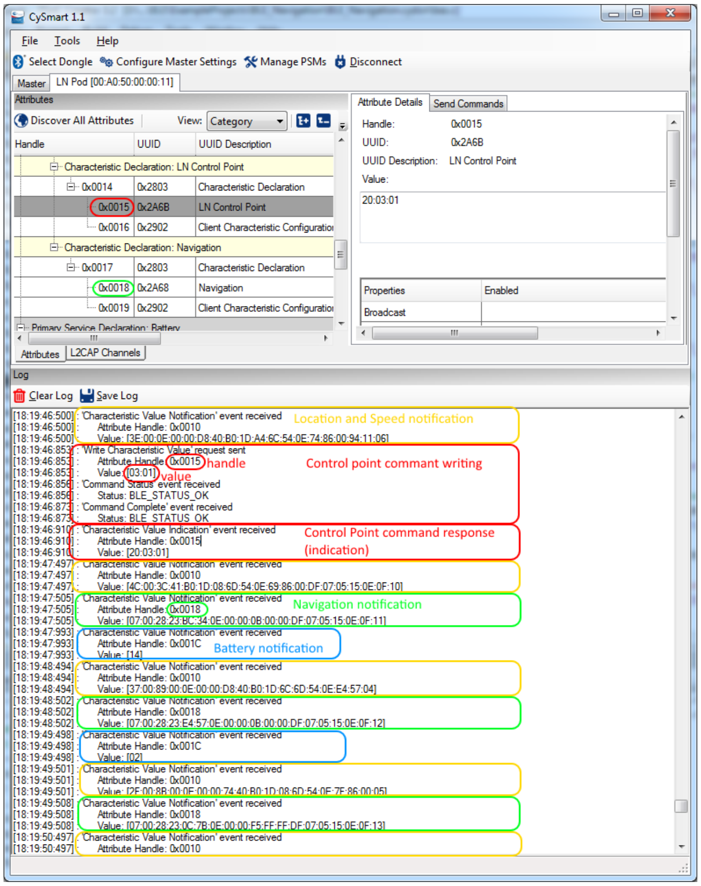
The project sends the Location and Navigation Service characteristic's notifications or indications and Battery Level notifications to the Central Client device, which displays to the user. The LEDs blink as described in the Design section

The project is intended to work in a pair with any BLE-compatible device (for example, phone, tablet) with the appropriate software (for example, Android, iOS with application that supports Location and Navigation Profile installed).

Also, the Location and Navigation Pod device can be used together with CySmart app for Windows. It is required to match the security settings between the Location and Navigation Pod device and CySmart Client and perform pairing (bonding) before any writing (enabling notifications and so on.) into the server's GATT database. For further instructions on how to use the CySmart application, see CySmart User Guide.

A simple example on how to use the CySmart Windows application as a Location and Navigation Service Client:

1. Connect the CySmart BLE dongle to a USB port on the PC.
2. Launch the CySmart app and select the connected dongle in the dialog window.
3. Press **SW1** to reset the development kit to start advertising.
4. Click **Start Scan** to discover available devices.
5. Select **LN Pod** in the list of available devices and connect to it.
6. Click **Pair**, then **Discover All Attributes**, and **Enable All Notifications** in the CySmart app.
7. Select the Location and Navigation Control Point characteristic value and enter the **03 01** command, which means "Navigation Control: Start Navigation" (all these commands and data structures are described in detail in the LNS Specification).
8. Observe the Navigation and Location and Speed characteristic notifications with simulated data (target is continuously moving around Cypress Ukraine office):



Handle	UUID	UUID Description
Characteristic Declaration: LN Control Point		
0x0014	0x2803	Characteristic Declaration
0x0015	0x2A6B	LN Control Point
0x0016	0x2902	Client Characteristic Configuratio
Characteristic Declaration: Navigation		
0x0017	0x2803	Characteristic Declaration
0x0018	0x2A68	Navigation
0x0019	0x2902	Client Characteristic Configuratio
Primary Service Declaration: Battery		

Handle: 0x0015
UUID: 0x2A6B
UUID Description: LN Control Point
Value:
20:03:01

Properties	Enabled
Broadcast	

```
[18:19:46:500] : 'Characteristic Value Notification' event received
[18:19:46:500] :     Attribute Handle: 0x0010
[18:19:46:500] :     Value: [3E:00:0E:00:00:D8:40:B0:1D:A4:6C:54:0E:74:86:00:94:11:06]
[18:19:46:853] : 'Write Characteristic Value' request sent
[18:19:46:853] :     Attribute Handle: 0x0015
[18:19:46:853] :     Value: [03:01]
[18:19:46:856] : 'Command Status' event received
[18:19:46:856] :     Status: BLE_STATUS_OK
[18:19:46:873] : 'Command Complete' event received
[18:19:46:873] :     Status: BLE_STATUS_OK
[18:19:46:910] : 'Characteristic Value Indication' event received
[18:19:46:910] :     Attribute Handle: 0x0015
[18:19:46:910] :     Value: [20:03:01]
[18:19:47:497] : 'Characteristic Value Notification' event received
[18:19:47:497] :     Attribute Handle: 0x0010
[18:19:47:497] :     Value: [4C:00:3C:41:B0:1D:08:6D:54:0E:69:86:00:DF:07:05:15:0E:0F:10]
[18:19:47:505] : 'Characteristic Value Notification' event received
[18:19:47:505] :     Attribute Handle: 0x0018
[18:19:47:505] :     Value: [07:00:28:23:BC:34:0E:00:00:0B:00:00:DF:07:05:15:0E:0F:11]
[18:19:47:993] : 'Characteristic Value Notification' event received
[18:19:47:993] :     Attribute Handle: 0x001C
[18:19:47:993] :     Value: [14]
[18:19:48:494] : 'Characteristic Value Notification' event received
[18:19:48:494] :     Attribute Handle: 0x0010
[18:19:48:494] :     Value: [37:00:89:00:0E:00:00:D8:40:B0:1D:6C:6D:54:0E:E4:57:04]
[18:19:48:502] : 'Characteristic Value Notification' event received
[18:19:48:502] :     Attribute Handle: 0x0018
[18:19:48:502] :     Value: [07:00:28:23:E4:57:0E:00:00:0B:00:00:DF:07:05:15:0E:0F:12]
[18:19:49:498] : 'Characteristic Value Notification' event received
[18:19:49:498] :     Attribute Handle: 0x001C
[18:19:49:498] :     Value: [02]
[18:19:49:501] : 'Characteristic Value Notification' event received
[18:19:49:501] :     Attribute Handle: 0x0010
[18:19:49:501] :     Value: [2F:00:8B:00:0E:00:00:74:40:B0:1D:08:6D:54:0E:7F:86:00:05]
[18:19:49:508] : 'Characteristic Value Notification' event received
[18:19:49:508] :     Attribute Handle: 0x0018
[18:19:49:508] :     Value: [07:00:28:23:0C:7B:0E:00:00:F5:FF:FF:DF:07:05:15:0E:0F:13]
[18:19:50:497] : 'Characteristic Value Notification' event received
[18:19:50:497] :     Attribute Handle: 0x0010
```

The corresponding example of UART log:

```
BLE Location and Navigation Example
BLE Stack Version: 5.0.0.718
CY_BLE_EVT_STACK_ON, StartAdvertisement
CY_BLE_EVT_SET_DEVICE_ADDR_COMPLETE
CY_BLE_EVT_LE_SET_EVENT_MASK_COMPLETE
CY_BLE_EVT_GET_DEVICE_ADDR_COMPLETE: 00a050000011
CY_BLE_EVT_SET_TX_PWR_COMPLETE
```

```
CY_BLE_EVT_SET_TX_PWR_COMPLETE
CY_BLE_EVT_GAPP_ADVERTISEMENT_START_STOP, state: 2
CY_BLE_EVT_GAP_KEYS_GEN_COMPLETE
CY_BLE_EVT_TIMEOUT: 1
CY_BLE_EVT_GAPP_ADVERTISEMENT_START_STOP, state: 1
CY_BLE_EVT_GAPP_ADVERTISEMENT_START_STOP, state: 2

CY_BLE_EVT_GATT_CONNECT_IND: 3, 13
CY_BLE_EVT_GAP_DEVICE_CONNECTED: connIntv = 48 ms
CY_BLE_EVT_CONNECTION_UPDATE_COMPLETE: connIntv = 7 ms
CY_BLE_EVT_CONNECTION_UPDATE_COMPLETE: connIntv = 48 ms
CY_BLE_EVT_GAP_AUTH_REQ: bdHandle=13, security=3, bonding=1, ekeySize=10, err=0
CY_BLE_EVT_GAP_SMP_NEGOTIATED_AUTH_INFO: bdHandle=13, security=2, bonding=1, ekeySize=10, err=0
CY_BLE_EVT_GAP_PASSKEY_ENTRY_REQUEST
Enter the passkey displayed on the peer device:
Enter a 6-digit passkey:
060892
  passkey: 060892 Passkey is sent
CY_BLE_EVT_GAP_ENCRYPT_CHANGE: 0
CY_BLE_EVT_GAP_KEYINFO_EXCHNGE_CMPLT
CY_BLE_EVT_GAP_AUTH_COMPLETE: security:2, bonding:1, ekeySize:10, authErr 0
CY_BLE_EVT_PENDING_FLASH_WRITE
Store bonding data, status: 0, pending: 0
CY_BLE_EVT_GATTS_INDICATION_ENABLED
CY_BLE_EVT_GATTS_XCNHG_MTU_REQ

CY_BLE_EVT_GATTS_READ_CHAR_VAL_ACCESS_REQ: handle: 11
Location and Speed Notification is Enabled
Store bonding data, status: 0, pending: 0

L&S Ntf: 37 00 89 00 0e 00 00 d8 40 b0 1d 6c 6d 54 0e e4 57 04
L&S Ntf: 2f 00 8b 00 0e 00 00 74 40 b0 1d 08 6d 54 0e 7f 86 00 05
L&S Ntf: 3e 00 0e 00 00 d8 40 b0 1d a4 6c 54 0e 74 86 00 94 11 06

CY_BLE_EVT_GATTS_READ_CHAR_VAL_ACCESS_REQ: handle: 19
Navigation Notification is Enabled
Store bonding data, status: 0, pending: 0

L&S Ntf: 4c 00 3c 41 b0 1d 08 6d 54 0e 69 86 00 df 07 05 15 0e 0f 00
L&S Ntf: 37 00 89 00 0e 00 00 d8 40 b0 1d 6c 6d 54 0e e4 57 04
L&S Ntf: 2f 00 8b 00 0e 00 00 74 40 b0 1d 08 6d 54 0e 7f 86 00 05

CY_BLE_EVT_GATTS_READ_CHAR_VAL_ACCESS_REQ: handle: 16
LN Control Point Indication is Enabled
Store bonding data, status: 0, pending: 0

CP is written: 03 01
Opcode: Navigation Control
Parameter: Start navigation
CP Ind: 20 03 01
LN Control Point Indication is Confirmed
L&S Ntf: 3e 00 0e 00 00 d8 40 b0 1d a4 6c 54 0e 74 86 00 94 11 06
Navigation Ntf: 07 00 28 23 94 11 0e 00 00 f5 ff ff df 07 05 15 0e 0f 08
L&S Ntf: 4c 00 3c 41 b0 1d 08 6d 54 0e 69 86 00 df 07 05 15 0e 0f 08
Navigation Ntf: 07 00 28 23 bc 34 0e 00 00 0b 00 00 df 07 05 15 0e 0f 09
L&S Ntf: 37 00 89 00 0e 00 00 d8 40 b0 1d 6c 6d 54 0e e4 57 04
Navigation Ntf: 07 00 28 23 e4 57 0e 00 00 0b 00 00 df 07 05 15 0e 0f 0a
```

Related Documents

Application Notes		
AN210781	Getting Started with PSoC 6 MCU with Bluetooth Low Energy (BLE) Connectivity	Describes the PSoC 6 MCU with BLE Connectivity, and how to build a basic code example.
AN215656	PSoC 6 MCU Dual-Core CPU System Design	Presents the theory and design considerations related to this code example.
Software and Drivers		
CySmart – BLE Test and Debug Tool		CySmart is a BLE host emulation tool for Windows PCs. The tool provides an easy-to-use GUI to enable the user to test and debug their BLE Peripheral applications.
PSoC Creator Component Datasheets		
Bluetooth Low Energy (BLE_PDL) Component		The Bluetooth Low Energy (BLE_PDL) Component provides a comprehensive GUI-based configuration window to facilitate designing applications requiring BLE connectivity.
Device Documentation		
PSoC 6 MCU: PSoC 63 with BLE Datasheet Programmable System-on-Chip		PSoC 6 MCU: PSoC 63 with BLE Architecture Technical Reference Manual (TRM)
Development Kit (DVK) Documentation		
CY8CKIT-062-BLE PSoC 6 BLE Pioneer Kit		

Document History

Document Title: CE215123 - BLE Location and Navigation with PSoC 6 MCU with BLE Connectivity

Document Number: 002-15123

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	5968182	NPAL	11/21/2017	New Code Example

Worldwide Sales and Design Support

Cypress maintains a worldwide network of offices, solution centers, manufacturer's representatives, and distributors. To find the office closest to you, visit us at Cypress Locations.

Products

ARM® Cortex® Microcontrollers	cypress.com/arm
Automotive	cypress.com/automotive
Clocks & Buffers	cypress.com/clocks
Interface	cypress.com/interface
Internet of Things	cypress.com/iot
Memory	cypress.com/memory
Microcontrollers	cypress.com/mcu
PSoC	cypress.com/psoc
Power Management ICs	cypress.com/pmic
Touch Sensing	cypress.com/touch
USB Controllers	cypress.com/usb
Wireless Connectivity	cypress.com/wireless

PSoC® Solutions

PSoC 1 | PSoC 3 | PSoC 4 | PSoC 5LP | PSoC 6

Cypress Developer Community

Forums | WICED IOT Forums | Projects | Videos | Blogs | Training | Components

Technical Support

cypress.com/support

All other trademarks or registered trademarks referenced herein are the property of their respective owners.

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709

TO THE EXTENT PERMITTED BY APPLICABLE LAW, CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS DOCUMENT OR ANY SOFTWARE OR ACCOMPANYING HARDWARE, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. To the extent permitted by applicable law, Cypress reserves the right to make changes to this document without further notice. Cypress does not assume any liability arising out of the application or use of any product or circuit described in this document. Any information provided in this document, including any sample design information or programming code, is provided only for reference purposes. It is the responsibility of the user of this document to properly design, program, and test the functionality and safety of any application made of this information and any resulting product. Cypress products are not designed, intended, or authorized for use as critical components in systems designed or intended for the operation of weapons, weapons systems, nuclear installations, life-support devices or systems, other medical devices or systems (including resuscitation equipment and surgical implants), pollution control or hazardous substances management, or other uses where the failure of the device or system could cause personal injury, death, or property damage ("Unintended Uses"). A critical component is any component of a device or system whose failure to perform can be reasonably expected to cause the failure of the device or system, or to affect its safety or effectiveness. Cypress is not liable, in whole or in part, and you shall and hereby do release Cypress from any claim, damage, or other liability arising from or related to all Unintended Uses of Cypress products. You shall indemnify and hold Cypress harmless from and against all claims, costs, damages, and other liabilities, including claims for personal injury or death, arising from or related to any Unintended Uses of Cypress products.