Chapter 1: Introduction

After completing this chapter, you will understand what this class is, what topics are covered, and the overall class objectives. You will also have an overview of what development kits are available for the PSoC™ 4 and PSoC™ 6 device families.

Table of contents

Document conventions

Convention	Usage	Example
`Courier New`	Displays code and text commands	`CY_ISR_PROTO(MyISR);` `make build`
Italics	Displays file names and paths	*sourcefile.hex*
[**bracketed, bold**]	Displays keyboard commands in procedures	[**Enter**] or [**Ctrl**] [**C**]
Menu > Selection	Represents menu paths	**File > New Project > Clone**
Bold	Displays GUI commands, menu paths and selections, and icon names in procedures	Click the **Debugger** icon, and then click **Next**.

1.1 What is this class?

This class is an in-depth look into the PSoC™ 4 and PSoC™ 6 MCU families. The learning objective is to introduce you to the basic operation of the PSoC™ 4 and PSoC™ 6 MCU devices and familiarize you with their development flow within the ModusToolbox™ ecosystem. This should enable you to create your own applications for devices in the PSoC™ 4 and PSoC™ 6 families.

This is a "Level 2" class, meaning that it is intended as a detailed look into a specific product, in this case the PSoC™ 4 and PSoC™ 6 MCUs. This class is part of a series of classes that also include these other "levels":

- "Level 1" classes are intended as an entry point into a particular topic and cover a broad range of topics at a shallow depth.
- "Level 3" classes dig even deeper by looking at complete solutions such as Bluetooth®, Wi-Fi, Motor Control, or Machine Learning.

1.2 Prerequisites

- ModusToolbox™ Software Training Level 1 - Getting Started

This class will not cover what ModusToolbox™ software is, what tools it includes, or how to use any of its features. If you are unfamiliar with ModusToolbox™ software, before embarking on this class, you should take the "ModusToolbox™ Software Training Level 1 - Getting Started" class.

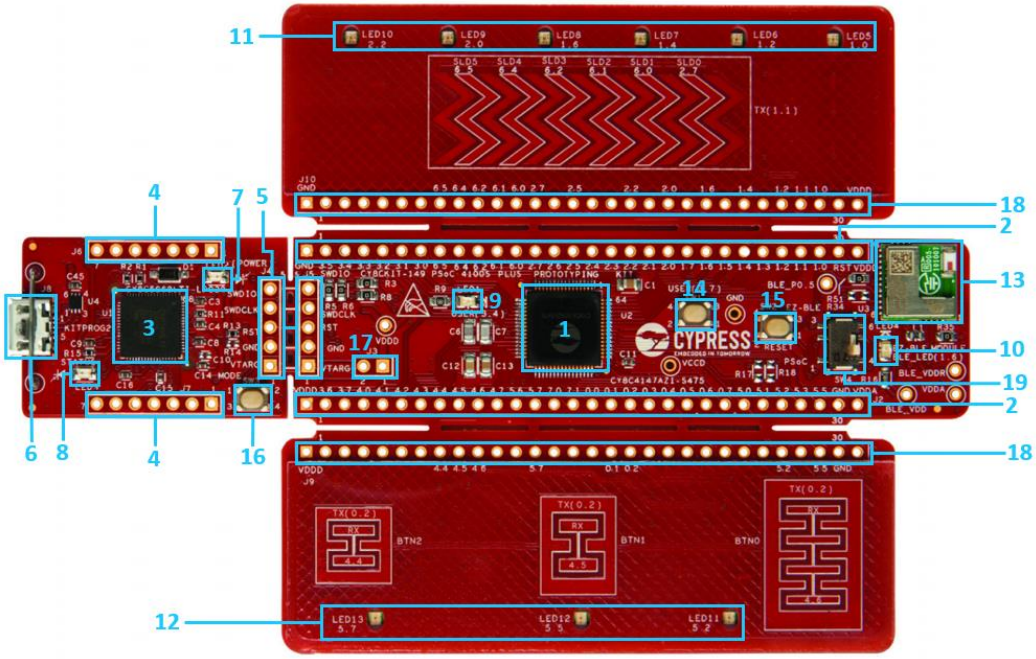
1.3 Development kits

For this class we will use the following development kits:

Note: *The PSoC™ 4 kit is optional. Almost all of the exercises in this course can be done using only the PSoC™ 6 kit.*

1.3.1 PSoC™ 4 kit

CY8CKIT-149 – A PSoC™ 4100S Plus MCU prototyping kit with an "EZ-BLE" PRoC Module

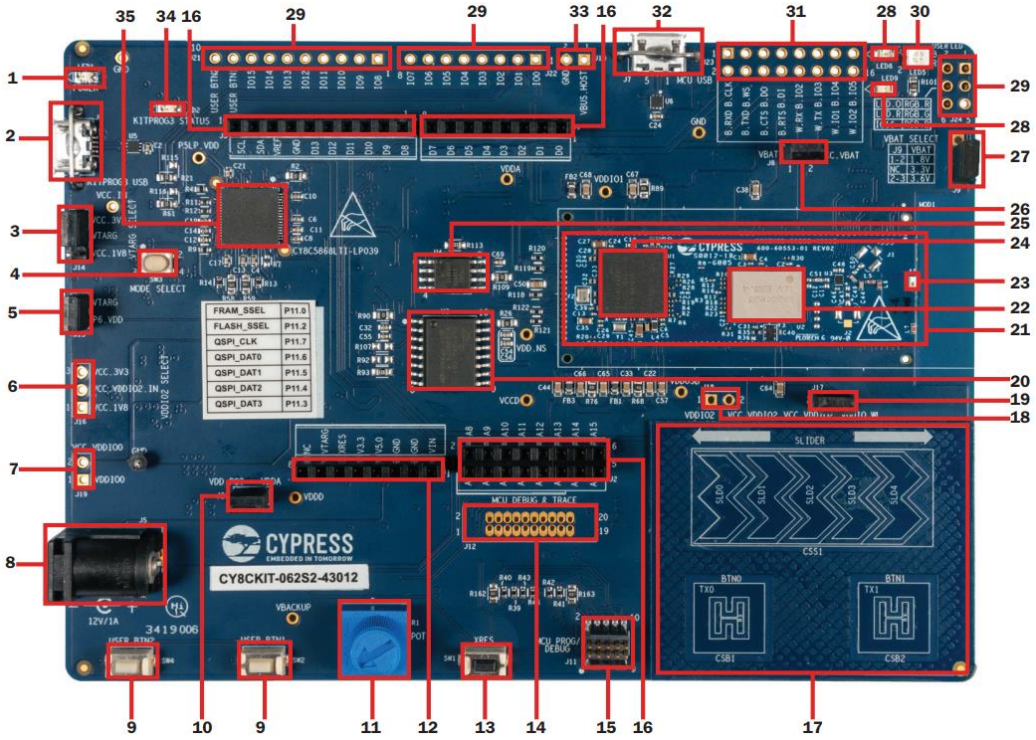


1. PSoC™ 4100S Plus device (CY8C4147AZI-S475)
2. PSoC™ 4100S Plus I/O headers (J1 and J2)
3. KitProg2 (PSoC™ 5LP MCU) device (CY8C5868LTI-LP039)
4. KitProg2 I/O headers (J6 and J7)
5. SWD connection headers (J4 and J5)
6. USB 2.0 Micro-B connector (J8)
7. One amber LED, LED2 (Power)
8. One amber LED, LED3 (KitProg2 Status)
9. One blue LED, LED1 (User)
10. One blue LED, LED4 (EZ-BLE PRoC™ User)
11. Six Green LEDs (LED5, LED6, LED7, LED8, LED9, LED10) (CAPSENSE™ Slider User)
12. Three Green LEDs (LED11, LED12, LED13) (CAPSENSE™ Button User)
13. EZ-BLE PRoC™ Module
14. One Push Button SW1 (User)
15. One Push Button SW2 (Reset)
16. One Push Button SW3 (KitProg2 Mode)
17. Current Measurement Jumper J3 (foot-print only) (shorted by 0 ohm resistor R53)
18. CAPSENSE™ headers (J9 and J10)
19. Program select switch (SW4)

Note: *Familiarize yourself with the locations of the user button (14) and the reset button (15). The user button is used in the exercises while the reset button can be used to reset the kit. After reset, the programmed firmware will execute starting from the beginning.*

1.3.2 PSoC™ 6 kit

CY8CKIT-062S2-43012 – A PSoC™ 6-2M MCU and a CYW43012 Wi-Fi + Bluetooth® Combo

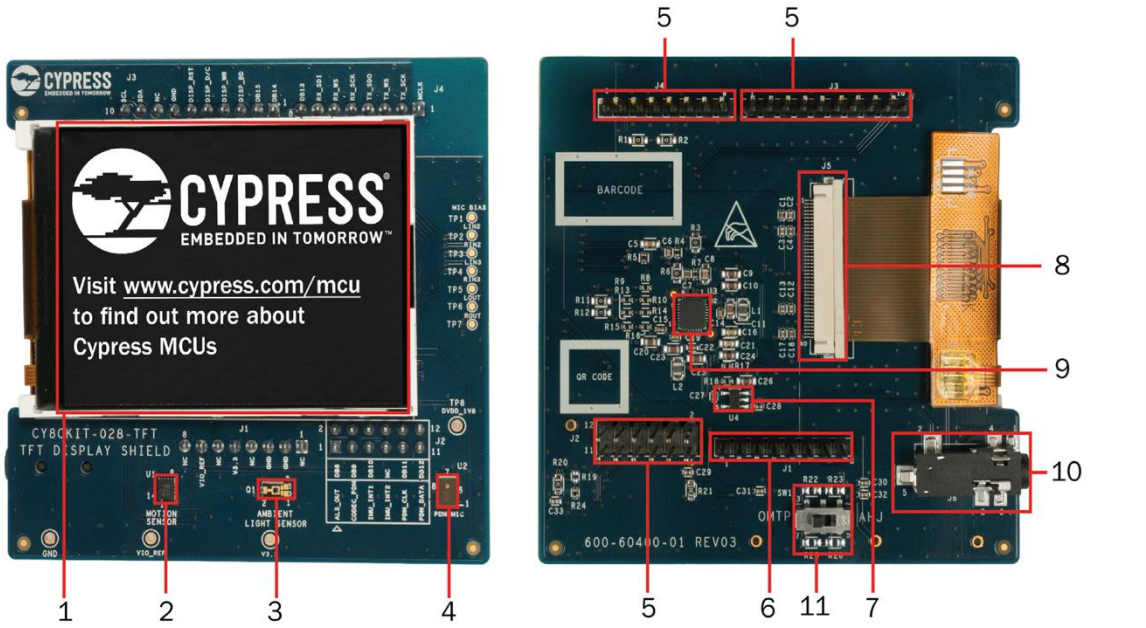


1. Power LED (LED1)
2. KitProg3 USB connector (J6)
3. PSoC™ 6 MCU VDD power selection jumper (J14)
4. KitProg3 programming mode selection button (SW3)
5. PSoC™ 6 MCU VDD current measurement jumper (J15)
6. PSoC™ 6 MCU VDDIO2 and CYW43012 VDDIO power selection jumper (J16)
7. PSoC™ 6 MCU VDDIO0 current measurement jumper (J19)
8. External power supply VIN connector (J5)
9. PSoC™ 6 MCU user buttons (SW2 and SW4)
10. Potentiometer connection jumper (J25)
11. Potentiometer (R1)
12. Arduino-compatible power header (J1)
13. PSoC™ 6 MCU reset button (SW1)
14. PSoC™ 6 MCU debug and trace header (J12)
15. PSoC™ 6 MCU program and debug header (J11)
16. Arduino Uno R3-compatible I/O headers (J2, J3, and J4)
17. CAPSENSE™ slider (SLIDER) and buttons (N0 and BTN1)
18. PSoC™ 6 MCU VDDIO2 current measurement jumper (J18)
19. CYW43012 VDDIO current measurement jumper(J17)
20. Infineon serial NOR flash memory (S25FL512S, U3)
21. Infineon PSoC™ 6 (2M) MCU with CYW43012 Carrier Module (CY8CMOD-062S2-43012, MOD1)
22. CYW43012 based Murata Type 1LV module
23. Wi-Fi/Bluetooth® antenna
24. PSoC™ 6 MCU
25. Infineon serial Ferroelectric RAM (CY15B104QSN, U4)
26. CYW43012 VBAT current measurement jumper (J8)
27. CYW43012 VBAT power selection jumper (J9):
28. PSoC™ 6 MCU user LEDs (LED8 and LED9)
29. PSoC™ 6 I/O header (J21, J22, J24)
30. RGB LED (LED5)
31. Wi-Fi/Bluetooth® GPIO header (J23)
32. PSoC™ 6 USB device connector (J7)
33. Optional USB Host power supply header (J10)
34. KitProg3 status LED (LED2)
35. KitProg3 (PSoC™ 5LP MCU) programmer and debugger (CY8C5868LTI-LP039, U2)
36. MicroSD Card holder (J20) (on back of board)

Note: *Familiarize yourself with the locations of the white user buttons (9) and the black reset button (13). The user buttons are used in the exercises while the reset button can be used to reset the kit. After reset, the programmed firmware will execute starting from the beginning.*

1.3.3 Arduino compatible shield

CY8CKIT-028-TFT



1. 2.4-inch TFT display
2. Motion Sensor (U1)
3. Ambient Light Sensor (Q1)
4. PDM microphone (U2)
5. Arduino compatible I/O headers (J2, J3, J4)
6. Arduino compatible power header (J1)
7. TFT display power control load switch (U4)
8. TFT display connector (J5)
9. Audio CODEC (U3)
10. Audio Jack (J6)
11. Audio Jack Selection (OMTP/AHJ) Switch (SW1)

1.4 Introduction to PSoC™

PSoC™ is a true programmable embedded SoC integrating configurable analog and digital peripheral functions, memory and a microcontroller on a single chip.

1.4.1 PSoC™ 4 MCUs

PSoC™ 4 devices are Arm®-based programmable devices, featuring the low-power Cortex®-M0 and Cortex®-M0+ cores combined with Infineon's unique programmable mixed-signal hardware IP and CAPSENSE™, resulting in the industry's most flexible and scalable low-power mixed-signal architecture.

1.4.2 PSoC™ 6 MCUs

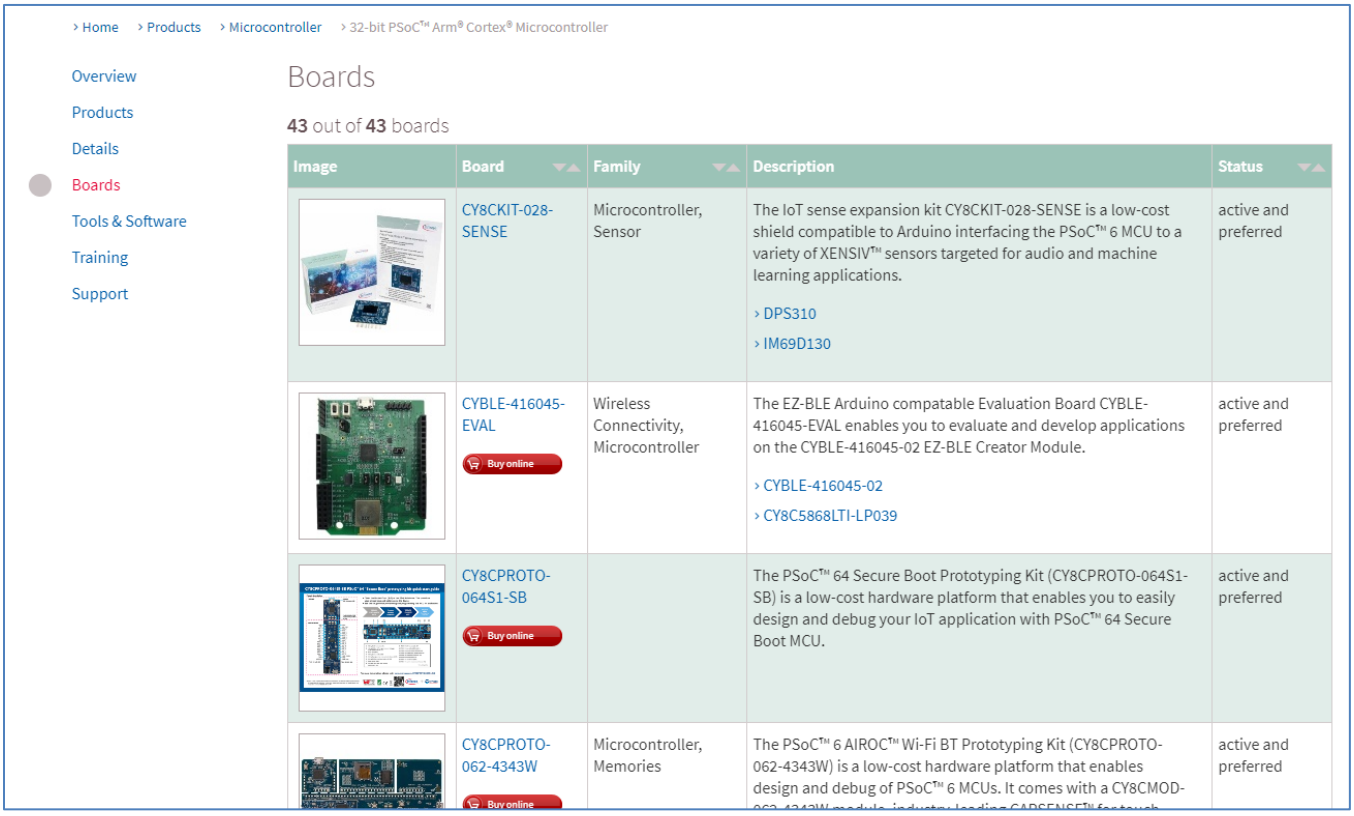
The PSoC™ 6 MCU is Infineon's newest PSoC™ MCU, purpose-built for the IoT, delivering the industry's lowest power, most flexibility and most secure solution. It provides dual-core Arm Cortex®-M4 and Arm Cortex®-M0+ CPUs running at 150-MHz and 100-MHz on an industry-leading ultra-low-power 40nm process that consumes as little as 22-µA/MHz in active power mode. It provides best-in-class flexibility with wired and wireless connectivity options, software defined peripherals and industry-leading CAPSENSE™, and integrated hardware-based Trusted Execution Environment (TEE) with secure data storage.

1.4.3 Development kits and Evaluation boards

Many development kits are available for both PSoC™ 4 and PSoC™ 6 devices. To find the list of kits, visit:

https://www.infineon.com/cms/en/product/microcontroller/32-bit-psoc-arm-cortex-microcontroller

Click the link that says Boards on the left side of the page for a list of all available kits.



1.5 PSoC™ KitProg programmer

The programmer firmware on the PSoC™ development kits is called KitProg3 (some kits ship with KitProg2 firmware, but we'll show you how to update it later). It runs on a PSoC™ 5LP chip also located on the kit. This firmware talks to your computer via USB and to the PSoC™ target device via a protocol called Serial Wire Debug (SWD). The host application on your computer needs to talk to the programmer to debug the PSoC™ MCU and to download firmware into the PSoC™ flash. There are a bunch of different protocols out there for accomplishing this task. However, a few years ago Arm developed a standard called CMSIS-DAP, which has two variants that are implemented in the KitProg firmware (Bulk and DAPLink).

Note: Older versions of KitProg firmware also support HID mode, which we typically don't use anymore.

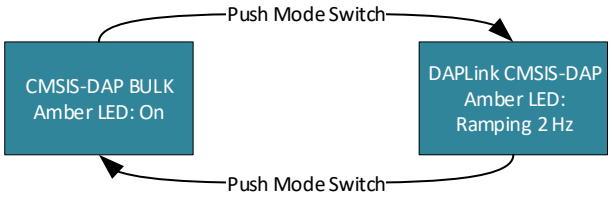
In addition to the CMSIS functions, there is also a function called "Mass storage". When the mass storage functionality is turned on, the programmer appears as a "flash drive" on your computer. You can copy – using the file manager – hex files to the flash drive, which will then be programmed. This function typically runs at the same time as the DAPLink functionality.

The programming firmware typically provides one or more communication bridge modes that allow the PSoC™ MCU to talk to your PC via I^2C or UART. These also typically run at the same time as the programming firmware.

The KitProg will appear to your computer to be multiple USB endpoints that implement each of the functions described in the previous paragraphs.

In order to program the PSoC™ MCU, KitProg needs to be in the right mode – meaning the mode that has the functionality that works with your environment. You can switch modes by pressing the mode button on the development kit, or by using the firmware loader program. Each PSoC development kit has an LED that will be solid or ramping (~2 Hz) to indicate the mode. See the following table.

CMSIS Mode	Application	Mass Storage	Bridges	Solution	Description	LED
BULK	Eclipse IDE	No	UART I2C	PSoC™ 6 devices, PSoC™ 4 devices& AFR	The latest version of the protocol which uses USB bulk mode – by far the fastest.	Solid
DAPLink	Mbed Studio or Mbed CLI	Yes	UART	Mbed OS	A modified version of CMSIS-DAP that enables web debugging	2 Hz Ramping



The PSoC™ 4 development kit we will use in this class, when updated to the latest KitProg, only has the CMSIS-DAP BULK mode.

1.6 PSoC™ Firmware Loader

Firmware loader (fw-loader) is a tool that we deliver as part of ModusToolbox™ software. It is a command-line tool that allows you to install new KitProg firmware onto a PSoC™ kit, and switch modes programmatically.

You can find it in the following directory:

<install_dir>/ModusToolbox/tools_<version>/fw-loader/bin

It is also available independently on GitHub at http://github.com/infineon/firmware-loader.

The following table shows some of the basic fw-loader commands:

Command	Description
`fw-loader --help` (or no argument)	Print out help information.
`fw-loader --device-list`	List all the KitProg devices attached to your computer.
`fw-loader --update-kp3`	Install the latest firmware onto your KitProg.
`fw-loader --mode kp3-daplink`	Put the KitProg into DAPLink CMSIS-DAP mode.
`fw-loader --mode kp3-bulk`	Put the KitProg into CMSIS-DAP Bulk mode.
`fw-loader --mode kp3-hid`	Put the KitProg into CMSIS-DAP HID mode (not typically used).
`fw-loader --mode kp3-bootloader`	Put the KitProg into bootloader mode (not typically used).

You will practice using this tool in exercise 2.

1.7 Exercises

Exercise 1: Look at kit documentation

1. Visit the websites for the kits used in this class and familiarize yourself with the kit documentation.

   All the relevant documentation can be found at the bottom of the kit's website in the "Related Files" section.

Exercise 2: Update KitProg firmware

Your kits may have KitProg2 installed rather than KitProg3. In this exercise we will determine what version of KitProg your kits have and update them if necessary.

Repeat these steps for each of your development kits.

1. Connect your kit to your computer using the provided USB cable. If your kit has multiple USB connectors, be sure to connect to the one labeled KITPROG.

2. Run a command line terminal

   Windows: run `modus-shell` from the Start menu or enter `modus-shell` in the Windows search box.
   MacOS or Linux: start a standard command terminal

3. Navigate to the directory *<install_dir>/ModusToolbox/tools_<version>/fw-loader/bin*

Note: *On Windows, steps 2 and 3 can be done together by either running `fw-loader` from the Start menu list or by entering `fw-loader` in the Windows search box.*

4. Check the KitProg firmware version on the kit:

   ```
   ./fw-loader --device-list
   ```

5. Does a device show up? What mode is it in? What version of KirProg does it have installed? Update it to the latest version of KitProg3:

   ```
   ./fw-loader --update-kp3
   ```

Trademarks
All referenced product or service names and trademarks are the property of their respective owners.

Published by
Infineon Technologies AG
81726 Munich, Germany

IMPORTANT NOTICE
The information given in this document shall in no event be regarded as a guarantee of conditions or characteristics ("Beschaffenheitsgarantie") .

With respect to any examples, hints or any typical values stated herein and/or any information regarding the application of the product, Infineon Technologies hereby disclaims any and all warranties and liabilities of any kind, including without limitation warranties of non-infringement of intellectual property rights of any third party.

In addition, any information given in this document is subject to customer's compliance with its obligations stated in this document and any applicable legal requirements, norms and standards concerning customer's products and any use of the product of Infineon Technologies in customer's applications.

The data contained in this document is exclusively intended for technically trained staff. It is the responsibility of customer's technical departments to evaluate the suitability of the product for the intended application and the completeness of the product information given in this document with respect to such application.

For further information on the product, technology, delivery terms and conditions and prices please contact your nearest Infineon Technologies office (**www.infineon.com**).

WARNINGS
Due to technical requirements products may contain dangerous substances. For information on the types in question please contact your nearest Infineon Technologies office.

Except as otherwise explicitly approved by Infineon Technologies in a written document signed by authorized representatives of Infineon Technologies, Infineon Technologies' products may not be used in any applications where a failure of the product or any consequences of the use thereof can reasonably be expected to result in personal injury.