Freescale Semiconductor
Application Note

AN2773
Rev. 1.1, 12/2004

Switch Demonstration

For the IEEE™ 802.15.4 and Zigbee MAC Layer

1 Introduction

This note shows users how to run the Switch Demonstration application built for the IEEE™ 802.15.4 Standard and Zigbee MAC Layer. The following sections describe how the Switch Demonstration functions and how users can get started by using their own applications. This note also includes a source file overview.

2 User Interface

The Switch Demonstration application runs on top of the IEEE™ 802.15.4 MAC layer. The Switch Demonstration is a direct implementation in that it does not include an operating system (OS) and is purely event driven. In this context, an event can be the push of a button or a message from the MAC.

At power up, a sequence of commands are sent to the MAC layer which makes the unit function as a Coordinator. As a Coordinator, the receiver is turned on and the unit is ready to answer scan and association requests.

Contents

The scenario for connecting two units is as follows. One unit will be a Coordinator and one unit a Device.

1. Power up boards. Both boards are now Coordinators. All LEDs are on.
2. Press Switch 1 (S1) on the Device unit. When S1 is pressed, the Device unit resets and starts an active scan (LED1 on). When the scan process terminates, the ScanConfirm response is searched for Coordinators that have the dedicated PanID. In this application, the PanID is 0xDEAF. If the PanID is found, the Device associates to the corresponding Coordinator on the given channel. The channel used in this application is 0x0C.
3. LED2 turns on (both Coordinator and Device) when the Associate procedure completes. If the quick start feature is included, the Device automatically associates to the last known Coordinator at power up.
4. The Device and Coordinator are now associated. By pressing Switch 4 (S4) a data packet is sent to the remote unit and LED4 on the remote unit toggles when the data packet is received. LED4 on the transmitter unit toggles if a DataConfirm is received from the MAC layer with Status = OK.

NOTE

If the PanID is not found in Step 2, LED1 is turned off and users should press S1 to make another scan. At this point, all LEDs are off. One exception is if the unit finds the PanID but fails to associate, then LED1 stays on but LED2 does not turn on as expected. Users are required to preform another scan by pressing S1.

If the Quick Start feature is included, the Device automatically associates to the last known Coordinator at power up. To clear this feature, users can press S3, which clears the Quick Start flag in the NVM block. If the unit is turned off and then on again, it returns to its initial condition as described in Step 1.

2.1 User Interface Summary

- LEDs
  - LED1: ON when a scan is started and OFF if the required PanID was not found.
  - LED2: ON when the Associate procedure is complete (now ready to send data).
  - LED3: Reserved.
  - LED4: Toggles if S4 is pressed on the remote side. Toggles when a DataConfirm is received from the MAC on the transmitter unit.
- Switches
  - S1: Start scan: MAC_Reset followed by MAC_Scan (active_scan).
  - S2: Startup sequence for a Coordinator: Mac_Reset, Mac_SetRequest (rxonidle, associatepermit, shortaddress) and Mac_Start (channel).
  - S3: Clears the quick start flag if the quick start feature is included (see section 4.2.2).
  - S4: Data transmission: Should only be executed after the associate procedure is complete. Uses a simple protocol defined for this application (see source code for details).

3 802.15.4 MAC Level

This section describes the configuration setup which consists of the commands necessary to connect two units and maintain the link (MLME). This section also provides a description of the data transmission (MCPS). Figure 1 shows the data transmission sequence.

3.1 Configuration setup

The switch demonstration uses only the basic 802.15.4 MAC primitives. After users start the scan, the Device automatically associates if a Coordinator was found. After the Association, the Device begins to poll the Coordinator. In the Switch Demonstration, the polling scheme is very aggressive because it always sends a new poll request each time the previous one has completed unless the low power feature is included (See Section 4.2.7, “Low Power”). In the low power scenario, the MCU and the MC13192/MC13193 are placed in a low power state for 160ms just before the poll request is sent.

3.2 Data transmission

On the 802.15.4 MAC layer, data from the Device to the Coordinator can be sent directly. Data from the Coordinator should be sent indirectly. That is, the Device polls the Coordinator. When sending a data packet from the Device to the Coordinator, the Device in the Switch Demonstration briefly stops the polling scheme. This is not required, but it is the expected behavior. Switch Demonstration data transmission is controlled by event flags. A flag is set when the user wants to send a packet (S4 is pressed). This flag is for a direct transmission and is checked every time a poll confirm is received. If set, the Device sends a data packet and then starts polling again when the data confirm message is received. For indirect data transmission, the flag is checked in the main loop. When the flag is set, the main loop sends a message to the MCPS handler, which then sends the data when it is polled by the Device.

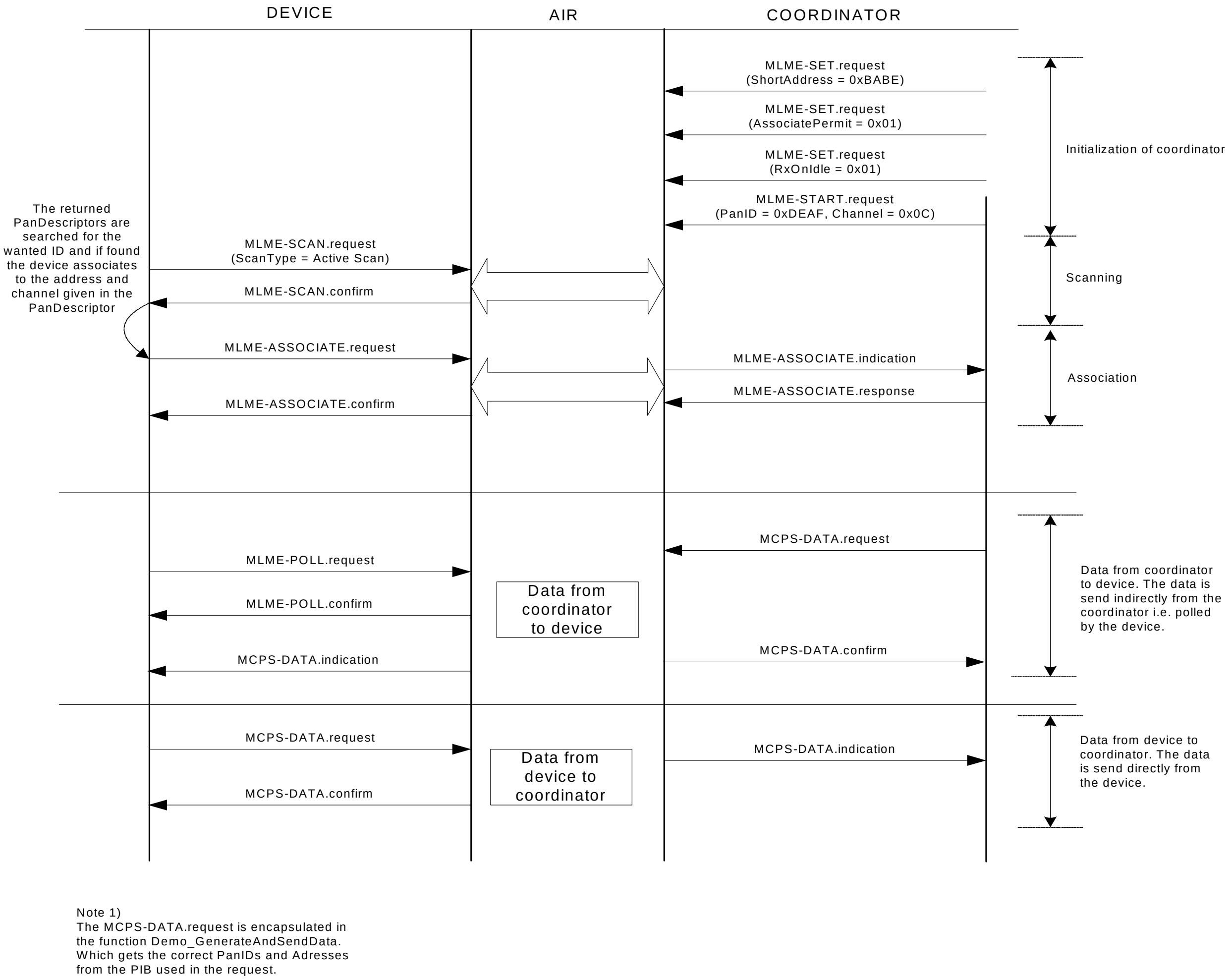

DEVICE
AIR
COORDINATOR
MLME-SET.request
(ShortAddress = 0xBABE)
MLME-SET.request
(AssociatePermit = 0x01)
MLME-SET.request
(RxOnIdle = 0x01)
MLME-START.request
(PanID = 0xDEAF, Channel = 0x0C)
Initialization of coordinator
The returned PanDescriptors are searched for the wanted ID and if found the device associates to the address and channel given in the PanDescriptor
MLME-SCAN.request
(ScanType = Active Scan)
MLME-SCAN.confirm
Scanning
MLME-ASSOCIATE.request
MLME-ASSOCIATE.indication
MLME-ASSOCIATE.response
MLME-ASSOCIATE.confirm
Association
MCPS-DATA.request
MLME-POLL.request
MLME-POLL.confirm
Data from coordinator to device
MCPS-DATA.indication
MCPS-DATA.confirm
Data from coordinator to device. The data is send indirectly from the coordinator i.e. polled by the device.
MCPS-DATA.request
MCPS-DATA.indication
Data from device to coordinator
MCPS-DATA.confirm
Data from device to coordinator. The data is send directly from the device.
Note 1)
The MCPS-DATA.request is encapsulated in the function Demo_GenerateAndSendData. Which gets the correct PanIDs and Adresses from the PIB used in the request.

Figure 1. Data Transmission Sequence Diagram

4 Using the Switch Demonstration

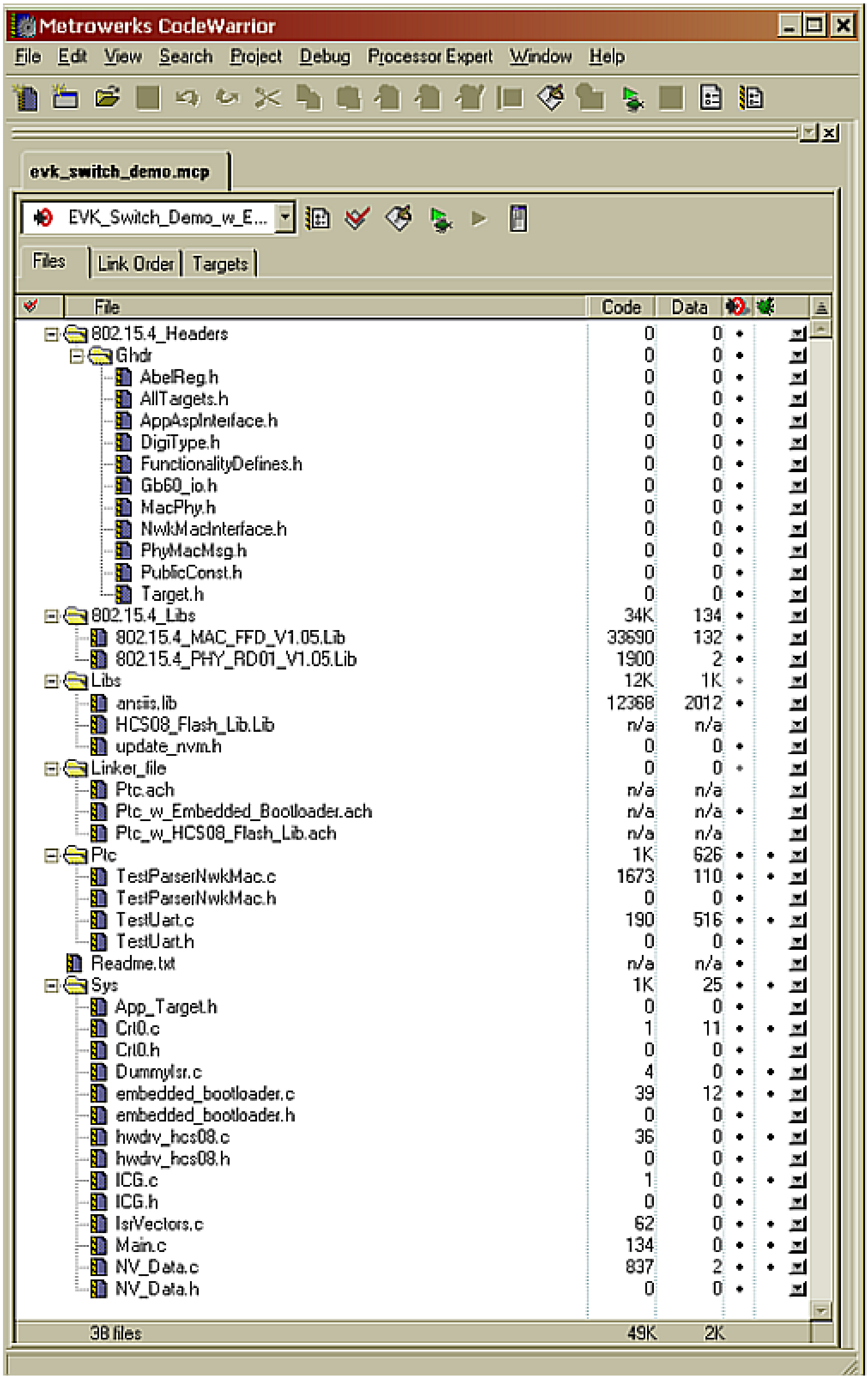
The source code for the Switch Demonstration application is provided in the 802.15.4 and Zigbee Evaluation Kit (EVK) software package. To access the source files, it is recommended that users open the project file for CodeWarrior (EVK_Switch_demonstration.mcp). This file is found under the MCP directory at the chosen install path. The Switch Demonstration consists of two main components:

1. The libraries of the MAC and PHY layer. These are compiled for the target hardware and can not be changed by users.
2. The source code for the Switch Demonstration application. This can be changed or replaced by users.

4.1 Project File Overview

Figure 2 shows the project files. These files are grouped into the following categories:

1. 802.15.4_Libs — A Pre-compiled MAC/PHY library for the specified PCB/HW. The MAC lib is also precompiled for the different Device types, in this case a FFD (Full Function Device).
2. 802.15.4_Headers — These are the header files for the provided libraries.
3. Libs — These are the libraries other than the PHY and MAC used in the application. One example is the ansiis.lib provided by Metroworks and the HCS08_Flash_Lib provided by Freescale.
4. PTC — The application specific files involve primarily the interfaces to/from the MAC layer and UART. The file naming is maintained for historical reasons. The TestParserNwkMac.c file handles the interface between the Network layer and the Mac layer. Because the Switch Demonstration does not have a Network layer to control it, most of the application functionality is placed in this file. The TestUart.c file contains the UART Interrupt Service Requests (ISR) and interface functions that provide an interface to the application for receiving/sending messages on the interface. In the Switch Demonstration, this is included for monitoring purposes only and is not the intended purpose of the Switch Demonstration application.
5. Sys — The system files include the Internal Clock Generator module (ICG.c), Embedded Bootloader interface description (embedded_bootloader.c), initialization code, main loop (main.c), Non-Volatile memory definition (NV_Data.c), and others.
6. Linker_file — The architecture file includes memory map definitions.



File	Code	Data		
802.15.4_Headers	0	0	•	
Ghdr	0	0	•	
AbelReg.h	0	0	•	
AllTargets.h	0	0	•	
AppAspInterface.h	0	0	•	
DigiType.h	0	0	•	
FunctionalityDefines.h	0	0	•	
Gb60_io.h	0	0	•	
MacPhy.h	0	0	•	
NwkMacInterface.h	0	0	•	
PhyMacMsg.h	0	0	•	
PublicConst.h	0	0	•	
Target.h	0	0	•	
802.15.4_Libs	34K	134	•	
802.15.4_MAC_FFD_V1.05.Lib	33690	132	•	
802.15.4_PHY_RD01_V1.05.Lib	1900	2	•	
Libs	12K	1K	•	
ansiis.lib	12368	2012	•	
HCS08_Flash_Lib.Lib	n/a	n/a		
update_nvm.h	0	0	•	
Linker_file	0	0	•	
Ptc.ach	n/a	n/a		
Ptc_w_Embedded_Bootloader.ach	n/a	n/a	•	
Ptc_w_HCS08_Flash_Lib.ach	n/a	n/a		
Ptc	1K	626	•	•
TestParserNwkMac.c	1673	110	•	•
TestParserNwkMac.h	0	0	•	
TestUart.c	190	516	•	•
TestUart.h	0	0	•	
Readme.txt	n/a	n/a	•	
Sys	1K	25	•	•
App_Target.h	0	0	•	
Crt0.c	1	11	•	•
Crt0.h	0	0	•	
DummyIsr.c	4	0	•	•
embedded_bootloader.c	39	12	•	•
embedded_bootloader.h	0	0	•	
hwdrv_hcs08.c	36	0	•	•
hwdrv_hcs08.h	0	0	•	
ICG.c	1	0	•	•
ICG.h	0	0	•	
IsrVectors.c	62	0	•	•
Main.c	134	0	•	•
NV_Data.c	837	2	•	•
NV_Data.h	0	0	•	
38 files	49K	2K		

Figure 2. Project File Overview

4.2 Features

The following features are included with the Switch Demonstration application.

4.2.1 MAC Access

The primary function of the Switch Demonstration application is to show how to use the MAC layer primitives. Users are provided the MAC layer (and below) as libraries which are easily linked together with the application.

The Switch Demonstration application shows how to use the most frequently used MAC primitives through the provided Service Access Points (SAP) and how to get messages back from the MAC layer through the provided message queues (see TestParserNwkMac.c file for details).

The MAC features are as follows:

- Set PIB — Used by the Device and Coordinator to set various variables in the PIB.
- Mac Start — Starts the Coordinator (PanID, channel).
- Scan — Normally used by the Device to collect information on Coordinators in the area. This information can then be used in the Association procedure (channel, short address).
- Association — The Device requests a short address from the Coordinator.
- Data Transfer (direct and indirect) — The Device can send direct data to the Coordinator. The Coordinator can only send indirect data to the Device. The Coordinator therefore needs to be polled by the Device to get the data sent. According to this scheme the Device can be an extreme low powered unit, because it only requires having the received open in a short period after each poll.
- Poll — Used by Device to get data from Coordinator.

Refer to the *802.15.4 MAC/PHY Software Reference Manual*, 802154MPSRM/D for more details.

4.2.2 Non-volatile Memory (NVM) Access

This feature enables usage of the existing NVM block specified by Freescale. For example, the configuration parameters for the system clock generation and the UART baud rate initialization. (See the TestUart.c file for details).

The application section of the NVM block is used in the Switch Demonstration to store the parameters from the Coordinator each time a scan is executed. If a unit previously has been associated with a Coordinator, the unit reads these parameters from the NVM block at power up. In this case, users are not required to do anything. This is considered a Quick Start. This makes it possible to completely turn off the Device to ensure minimal power consumption.

In the app_target.h file, the “#define USE_NVM_QUICK_START” includes this feature. Notice that the NVM is only accessible if the Flash Library or the Embedded Bootloader is included in the project.

Refer to the *802.15.4 MAC/PHY Software Reference Manual*, 802154MPSRM/D for more details.

4.2.3 Embedded Bootloader

This section describes how to build an application with the Embedded Bootloader. One of the targets in the EVK_Switch_demonstration.mcp files has the Embedded Bootloader functionality enabled.

Refer to the *Embedded Bootloader Reference Manual*, 802154EBRM/D for more details.

4.2.4 Flash Lib

This feature shows how to build an application with the Flash library. One of the targets in EVK_Switch_demonstration.mcp file has the Flash Library functionality enabled.

Refer to the *MCSO8 Flash Library Application Note*, AN2770 for more information.

4.2.5 GPIO Access

The Switch Demonstration uses one General Purpose I/O port in the HCS08 Microcontroller for the switches (configured as input) and one for the LEDs (configured as output). The configuration of these GPIOs is encapsulated in macros found in the Target.h file. For more complex GPIO control, refer to the latest HCS08 Microcontroller documentation.

4.2.6 UART Access

UART access in the Switch Demonstration is not intended for use as a host controller interface, but simply as a monitor of different messages. The PrintTestInterface function is provided to send a text string over the UART interface. The PrintTestInterface is placed in the TestUart.c file and is currently only used in the code on power up as follows:

```
PrintTestInterface(“I am alive\r\n”).
```

4.2.7 Low Power

If the Low Power feature is included in the project, low power mode is entered each time a poll request is about to be sent (see TestParserNwkMac.c for details). In the Switch Demonstration, the low power feature used is a variant of how the MC13192 is put in DOZE mode. In the MC13192 Doze Mode, the MCU is stopped (STOP3 mode) until the MC13192 wakes it up (after 160ms). This illustrates a very powerful method of saving power. In the app_target.h file, the “#define USE_LOW_POWER” includes this feature. Also in the Target.h file, the “mENTER_LOWPOWER” macro is defined to alter the required registers on the MCU and execute the stop instruction.

4.2.8 Porting to Other Hardware Boards

The Switch Demonstration allows uses to port to other hardware. The Switch Demonstration collects all hardware assignments (port settings) in an easy to understand, single file (app_target.h) which enables users to quickly move the application to a different hardware layout. In many cases, a re-compile of the PHY lib is required if the pin connections between the MC13192 and the MCU are changed.

NOTES

How to Reach Us:

USA/Europe/Locations Not Listed:
Freescale Semiconductor Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
1-800-521-6274 or 480-768-2130

Japan:
Freescale Semiconductor Japan Ltd.
Technical Information Center
3-20-1, Minami-Azabu, Minato-ku
Tokyo 106-8573, Japan
81-3-3440-3569

Asia/Pacific:
Freescale Semiconductor Hong Kong Ltd.
2 Dai King Street
Tai Po Industrial Estate
Tai Po, N.T., Hong Kong
852-26668334

Home Page:
http://Freescale.com

Information in this document is provided solely to enable system and software implementers to use Freescale Semiconductor products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.

Freescale Semiconductor reserves the right to make changes without further notice to any products herein. Freescale Semiconductor makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale Semiconductor 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 Freescale Semiconductor 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. Freescale Semiconductor does not convey any license under its patent rights nor the rights of others. Freescale Semiconductor products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Freescale Semiconductor product could create a situation where personal injury or death may occur. Should Buyer purchase or use Freescale Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify and hold Freescale Semiconductor and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Freescale Semiconductor was negligent regarding the design or manufacture of the part.