EVBUM2589

Bluetooth® Low Energy IoT Development Kit (B-IDK) Getting Started Guide

ON Semiconductor®

www.onsemi.com

EVAL BOARD USER'S MANUAL

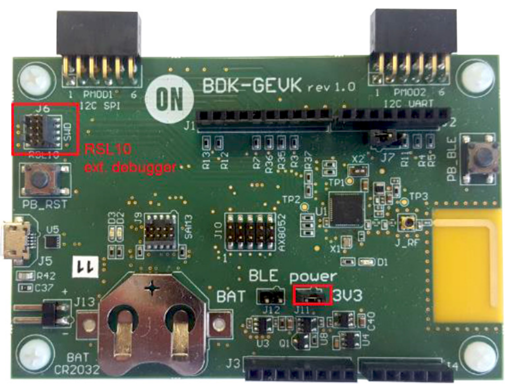


Figure 1. Board Photo

INTRODUCTION

This document helps you get started with the Bluetooth Low Energy IoT Development Kit (B−IDK). The B−IDK is a comprehensive node−to−cloud and a modular IoT platform that allows development of various BLE based use cases. Along with the hardware and software, the B−IDK includes a mobile app to interact with sensors and actuators.

The B−IDK features RSL10, Industry's lowest power Bluetooth 5 SoC and comprises of a baseboard (BDK−GEVK) and several sensor and actuator daughter cards. For a complete listing of available daughter cards, please visit https://www.onsemi.com/B−IDK. The daughter cards connect to the baseboard, via the two PMOD connectors and/or the Arduino connector to enable various use cases.

Scope

This document covers the hardware setup, software architecture, B−IDK documentation and provides instructions on downloading firmware to the board. The details regarding the mobile app and cloud connectivity are not covered in this document.

HARDWARE

- BDK−GEVK − B−IDK Baseboard
- Daughter Cards – Optional
- BDK−DCDC−GEVB – Power Shield For Use With Higher Power Daughter Cards – Optional

Default Configuration

The BDK−GEVK is shipped with the following jumper configuration. As the board supports OBD, there is no need for an external debugger. In case an external debugger is used, connect it to SWD header, J6.

Powering the Board

Multiple options are available to power the BDK−GEVK.

- USB
- Coin Cell (CR2032)
- External AC/DC Adapter plus power shield (BDK−DCDC−GEVB)
- External Supply

When higher power daughter cards (listed below) are attached to the baseboard, external supply either using the power shield or direct is required.

Higher Power Daughter Cards

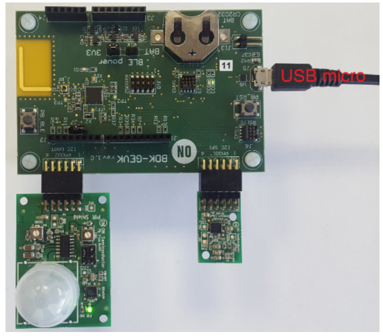
- D−LED−B−GEVK Dual LED Ballast
- D−STPR−GEVK Dual Stepper Motor Driver
- BLDC−GEVK BLDC Motor Driver

September, 2019 − Rev. 4

Publication Order Number:
EVBUM2589/D

USB

The B−IDK can be powered via the USB port when the use case doesn’t need any higher power daughter cards. An example configuration with the baseboard and a couple of sensor boards is shown below.



Coin Cell

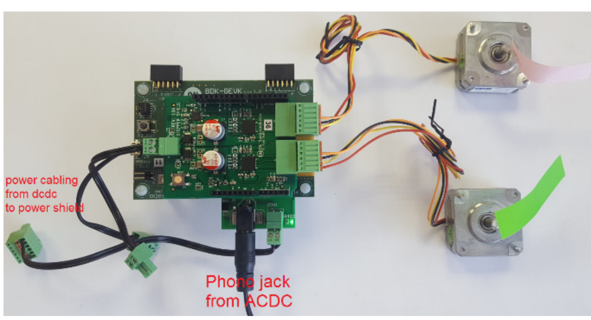
Once the firmware is flashed onto the baseboard, a coin cell (CR2032) may be used to power the system. Similar to USB based power supply, this method of powering is for use cases that don’t utilize the higher power daughter cards. The jumper configuration must match the below table to allow for various power modes.

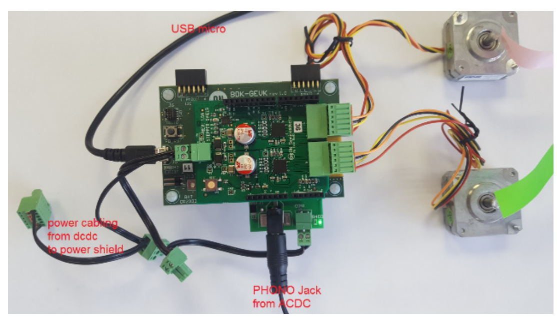
Table 1. JUMPERS

J11	J12	Usage
IN	X	Programming and Power over USB
X	IN	After programming. Only RSL10 is powered.
IN	IN	After programming. Both RSL 10 and OBD Microcontroller are powered

External AC/DC Adapter Plus Power Shield (BDK−DCDC−GEVB)

For use cases that utilize higher power daughter cards, an external AC/DC power supply (Ex: SMI24−12−V−P6) plus the power shield (BDK−DCDC−GEVB) are needed to power the system. While the 3.3 V supply to the baseboard is provided by the power shield via the Arduino connector, power cables (Green connector) are required between BDK−DCDC−GEVB and the higher power daughter card. For firmware flashing and debugging, the USB cable may be plugged in simultaneously with this mode as shown below.





External Supply

The B−IDK can be powered by an external supply via J13. In this mode, the battery cannot be installed. Jumpers J11 and J12 must be installed.

SOFTWARE

The B−IDK software allows for rapid development of various use cases. This section details the prerequisites and detailed steps in downloading firmware onto the baseboard.

Prerequisites

1. Install 64−bit version of Java from https://www.java.com/en/download/
2. Install J−Link Version 6.32f or later from https://www.segger.com/downloads/jlink (select J−Link software and documentation pack)
3. Download and install“On Semiconductor IDE Installer” from https://www.onsemi.com/PowerSolutions/product.do?id=RSL10
    a. Download the RSL10 SDK Getting Started Guide and RSL10 CMSIS pack under “RSL10 Software Package” from the above site. All of these are highlighted in the picture below. Save the CMSIS pack in a folder, for example, C:\cmsis_packs

4. Download the B−IDK CMSIS pack from https://www.onsemi.com/B−IDK and save it in the same folder as the RSL10 CMSIS pack (see 3.a above)
5. CMSIS pack at item 4. is dependent on ARM CMSIS pack as well. Please install ARM CMSIS pack 5.5.1 or higher after download from: https://github.com/ARM−software/CMSIS_5/releases
6. CMSIS pack at item 4. is also dependent on ARM CMSIS – FreeRTOS version 10.2.0 or higher for users exposed to design the code under FreeRTOS with RSL10: https://github.com/ARM−software/CMSIS−FreeRTOS/releases

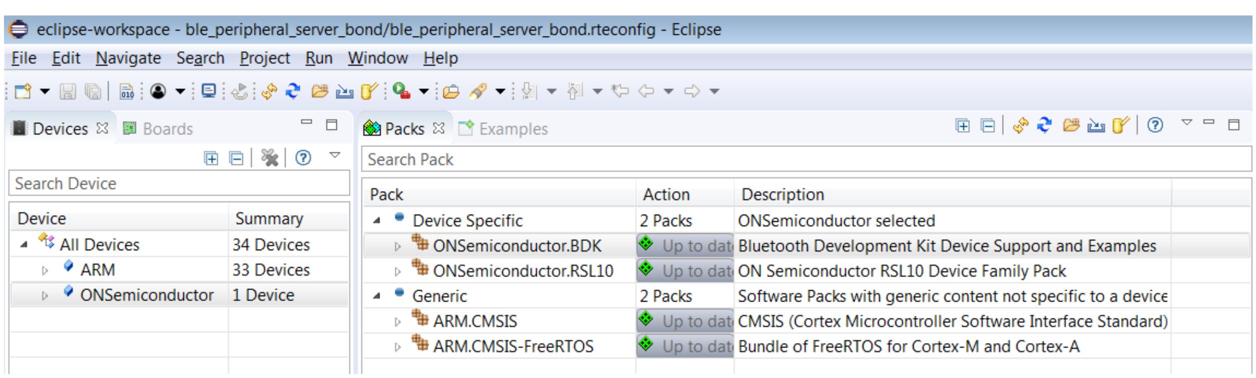
The next section provides details on importing the downloaded CMSIS packs into the SDK.

Importing CMSIS Packages

1. Launch the RSL10 SDK ON Semiconductor IDE

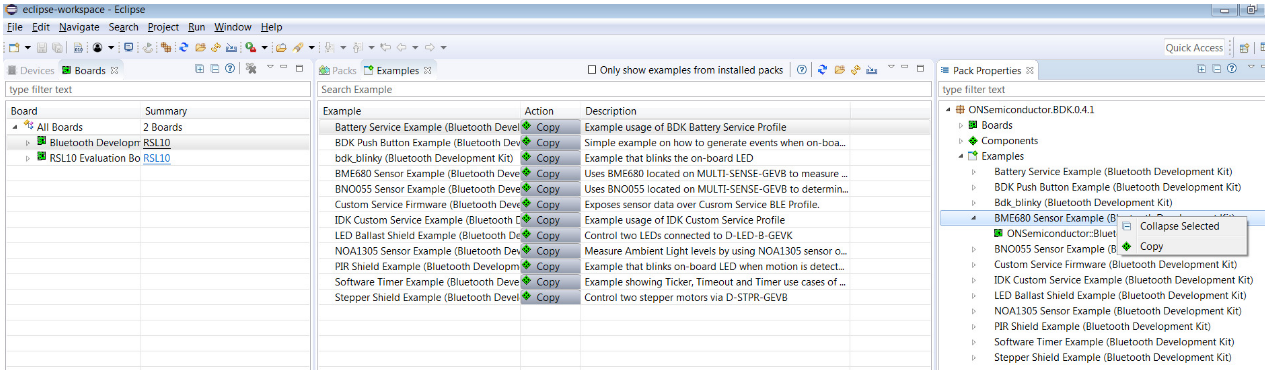
NOTE: Please import RSL10 CMSIS pack first as the B−IDK CMSIS pack (step 4 in the Prerequisites section) depends on the RSL10.

2. Refer to Chapter 3 of RSL10 SDK Getting Started Guide (step 3.a) for step−by−step instructions on importing the CMSIS packs.
3. Once all packs are successfully imported, they can be viewed in the CMSIS pack manager perspective as shown below.

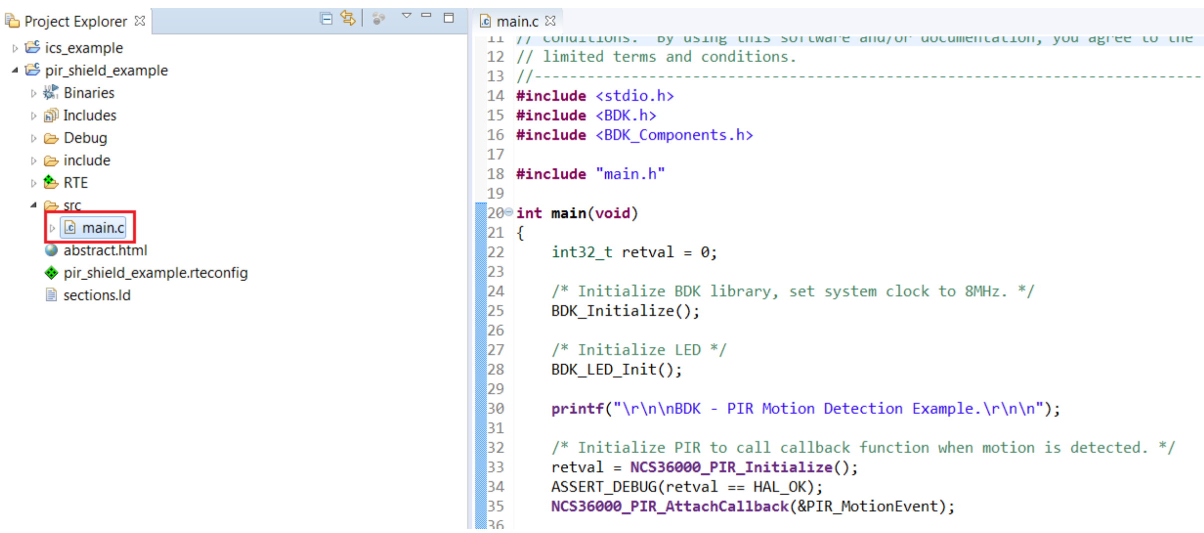


Compiling and Flashing

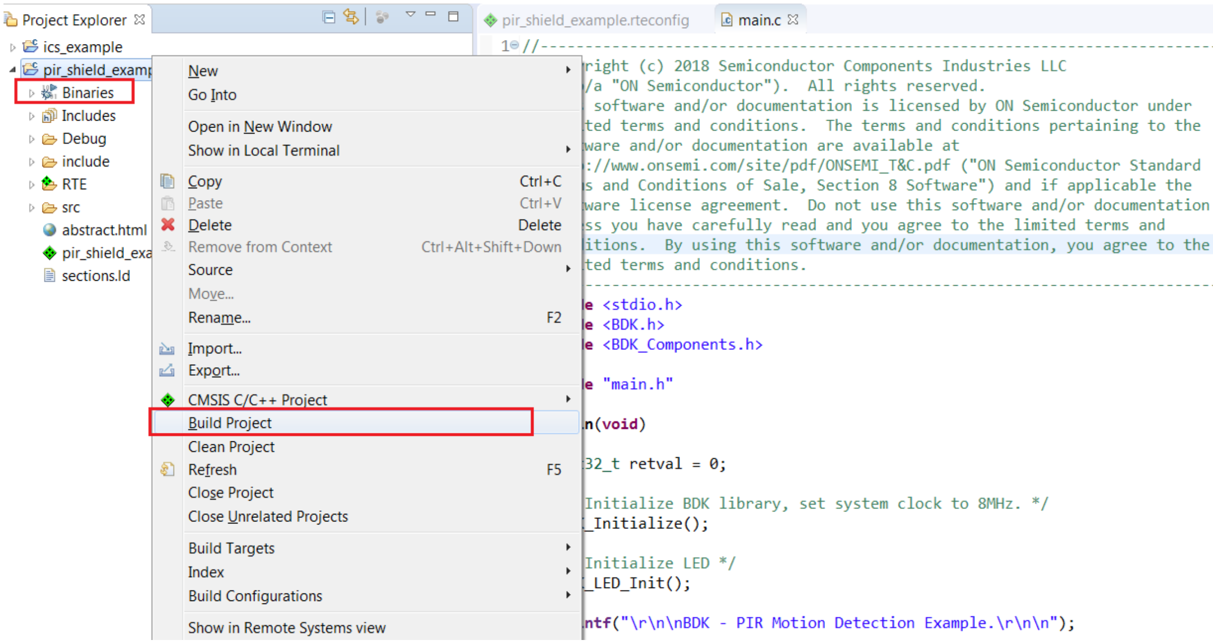
1. Choose an example (for example, pr_shield_example) to flash by copying it to the workspace.



NOTE: Once the example is copied, it can be viewed under Project Explorer. All source files including main are located in the src folder.

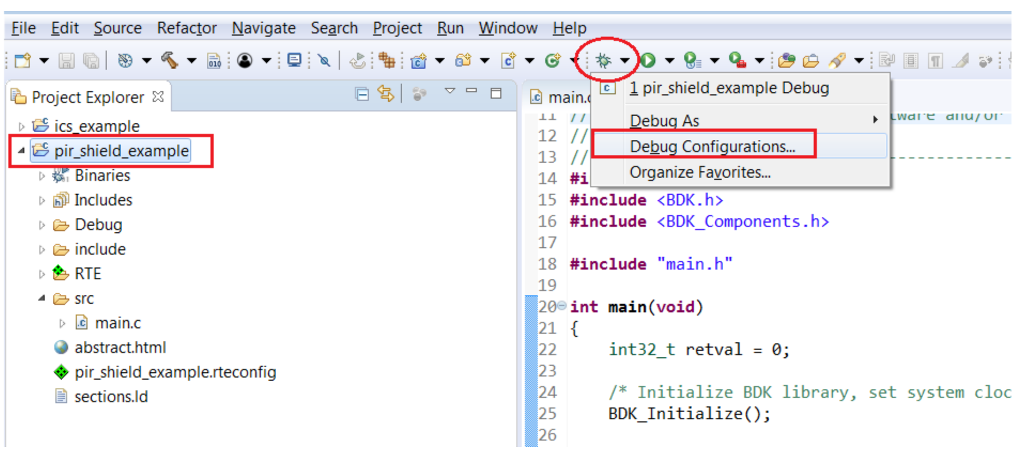


2. Right click and build the project. This creates binaries to be flashed to BDK−GEVK.

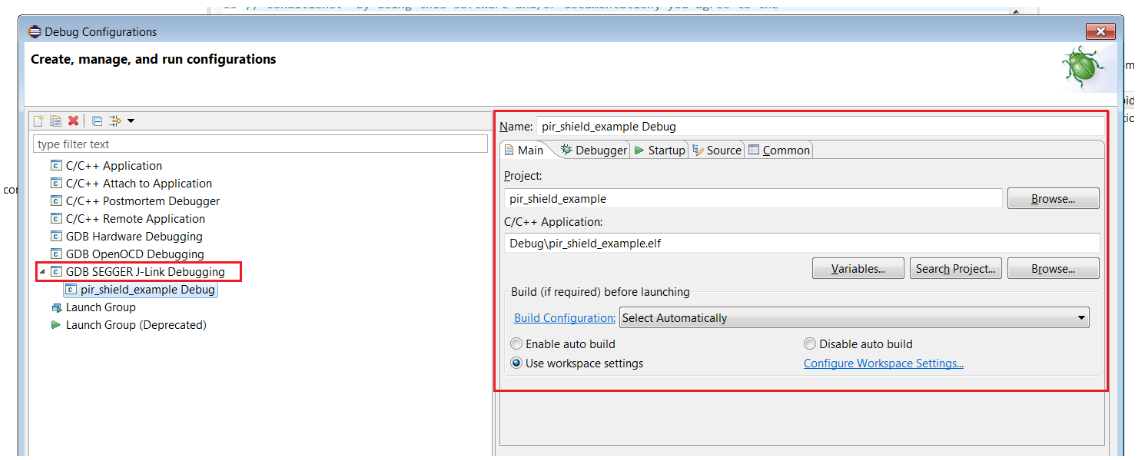


NOTE: If the binaries are not seen, press F5 (refresh).

3. Once the build is done, the code is ready to be flashed to the BDK−GEVK. Select the project (pir_shield_example), and go to the debug configurations as shown below.

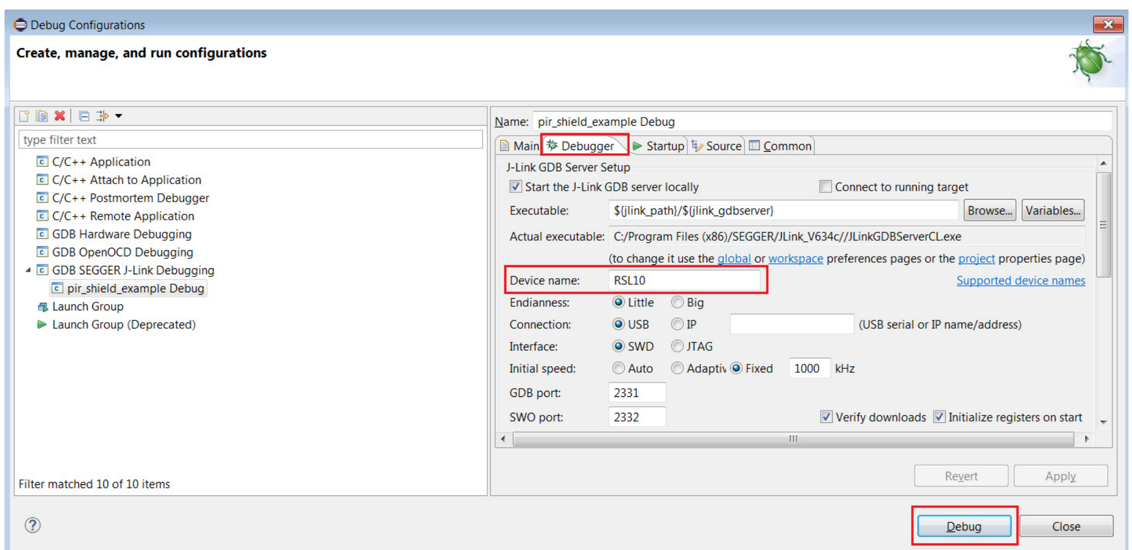


4. Double click **GDB Segger J−Link Debugging** to create the debug configuration for the selected example.

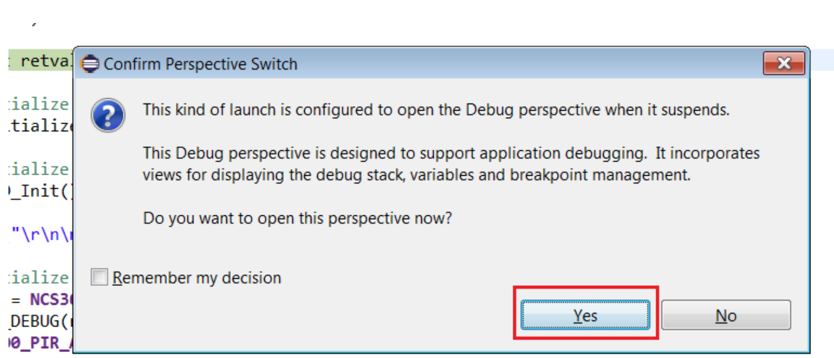


NOTE: The debug configuration for the selected example is automatically saved and there's no need to re−create it.

5. On the **Debugger** tab, set RSL10 as the device name. Click **Debug** to launch the code.



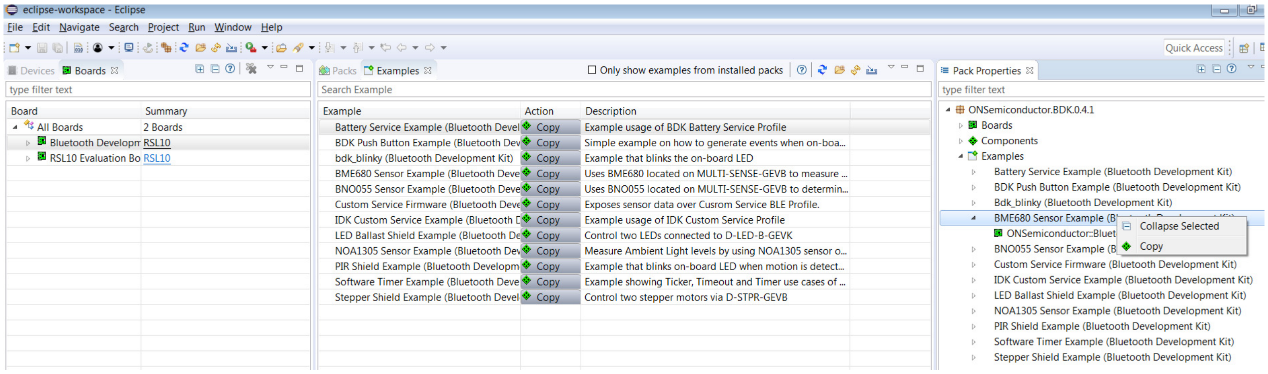
6. For application debugging, confirm perspective switch by clicking Yes.



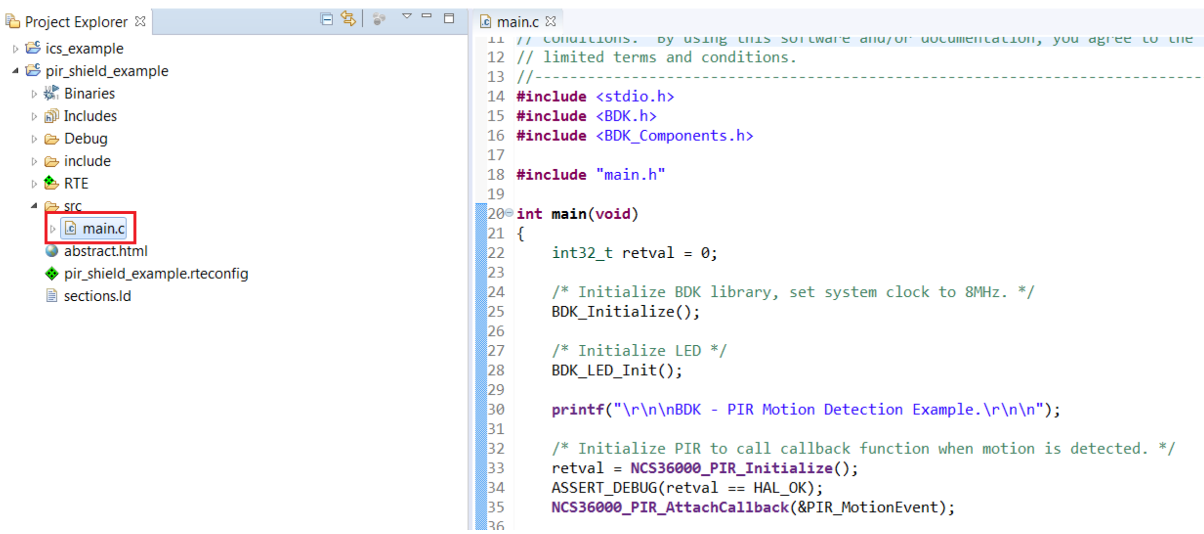
7. The debug session is now launched. Click Resume (F8) to start the target CPU.

Compiling and Flashing

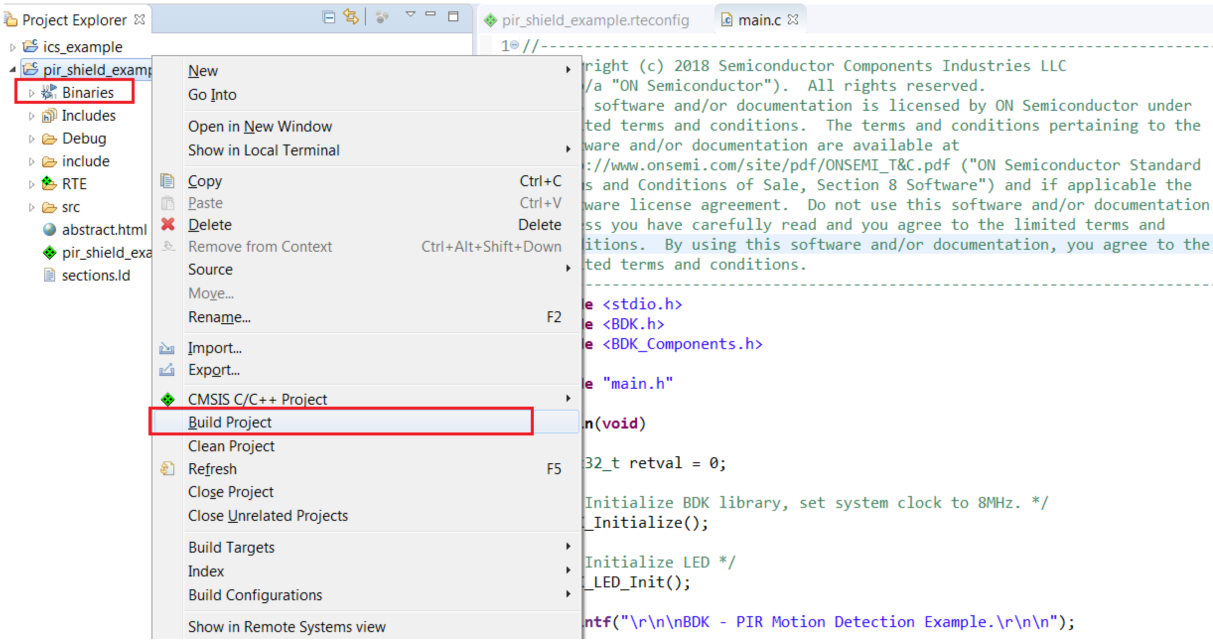
1. Choose an example (for example, pr_shield_example) to flash by copying it to the workspace.



NOTE: Once the example is copied, it can be viewed under Project Explorer. All source files including main are located in the src folder.

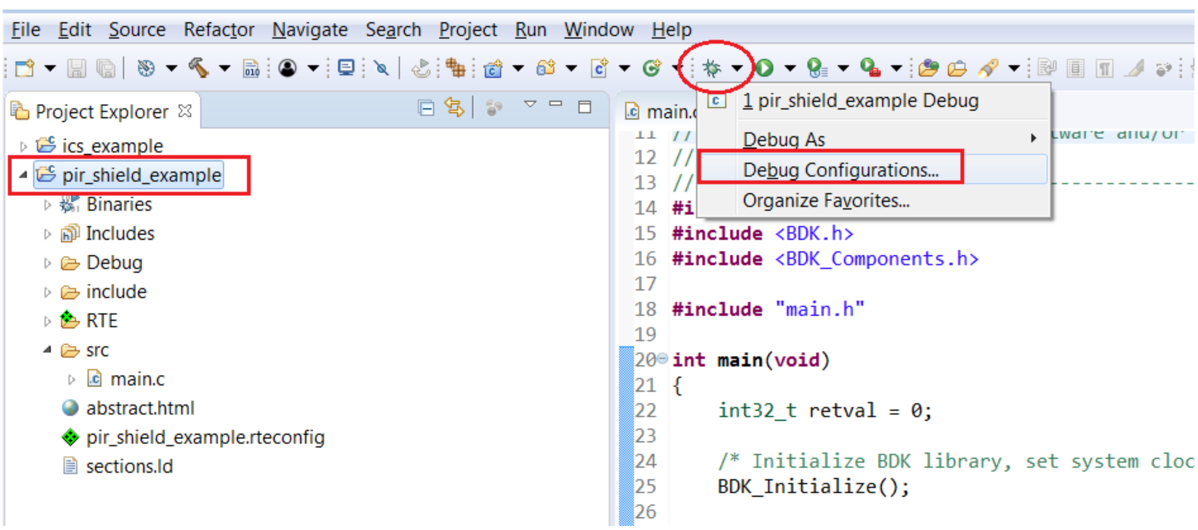


2. Right click and build the project. This creates binaries to be flashed to BDK−GEVK.

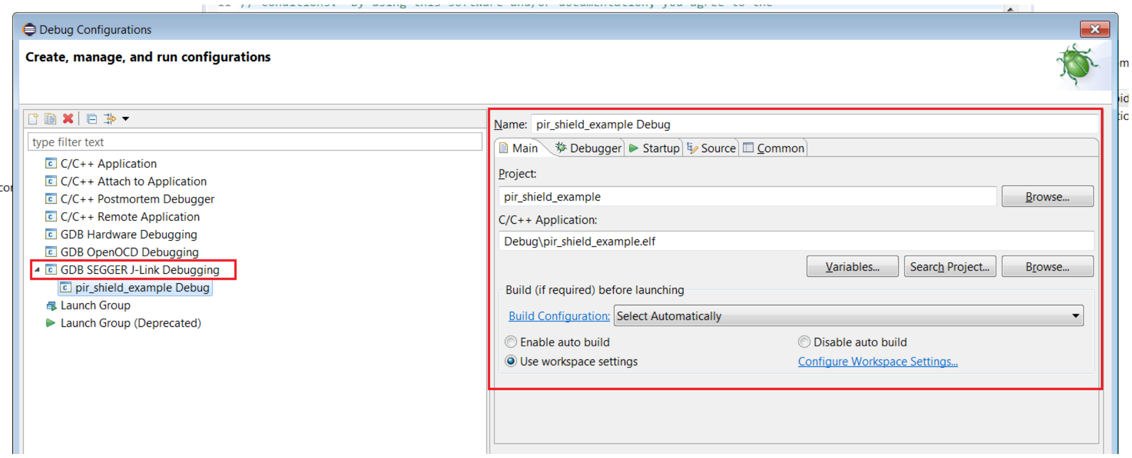


NOTE: If the binaries are not seen, press F5 (refresh).

3. Once the build is done, the code is ready to be flashed to the BDK−GEVK. Select the project (pir_shield_example), and go to debug configurations as shown below.

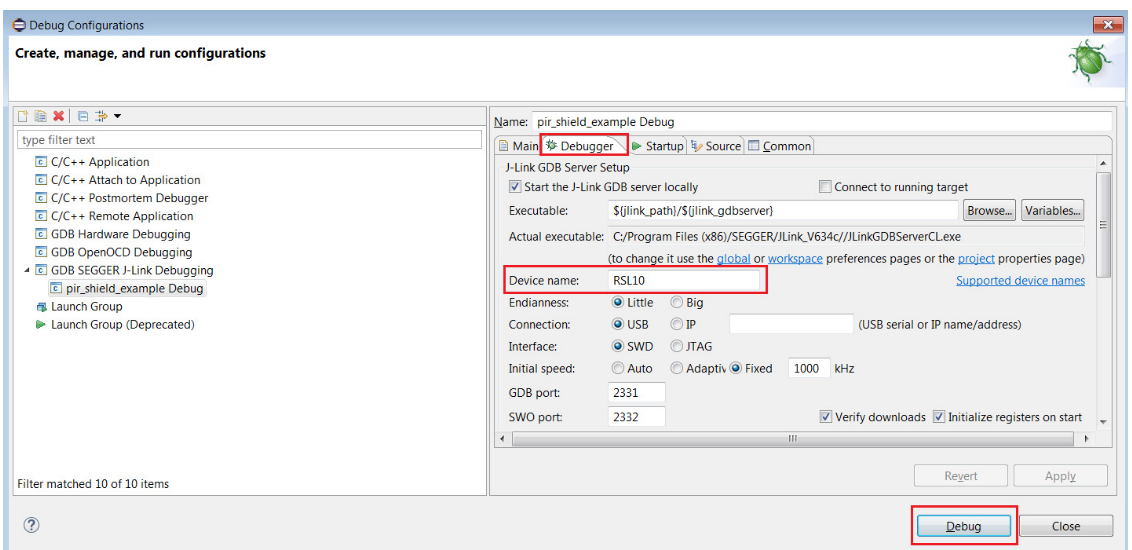


4. Double click GDB Segger J−Link Debugging to create the debug configuration for the selected example.

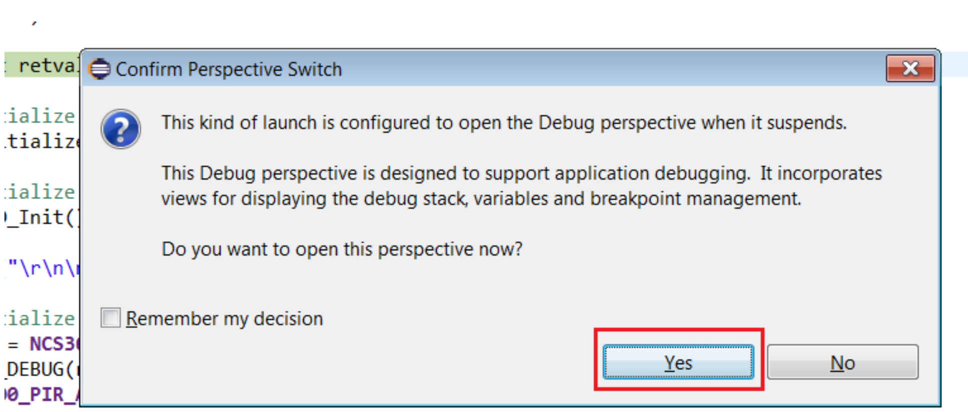


NOTE: The debug configuration for the selected example is automatically saved and there's no need to re−create it.

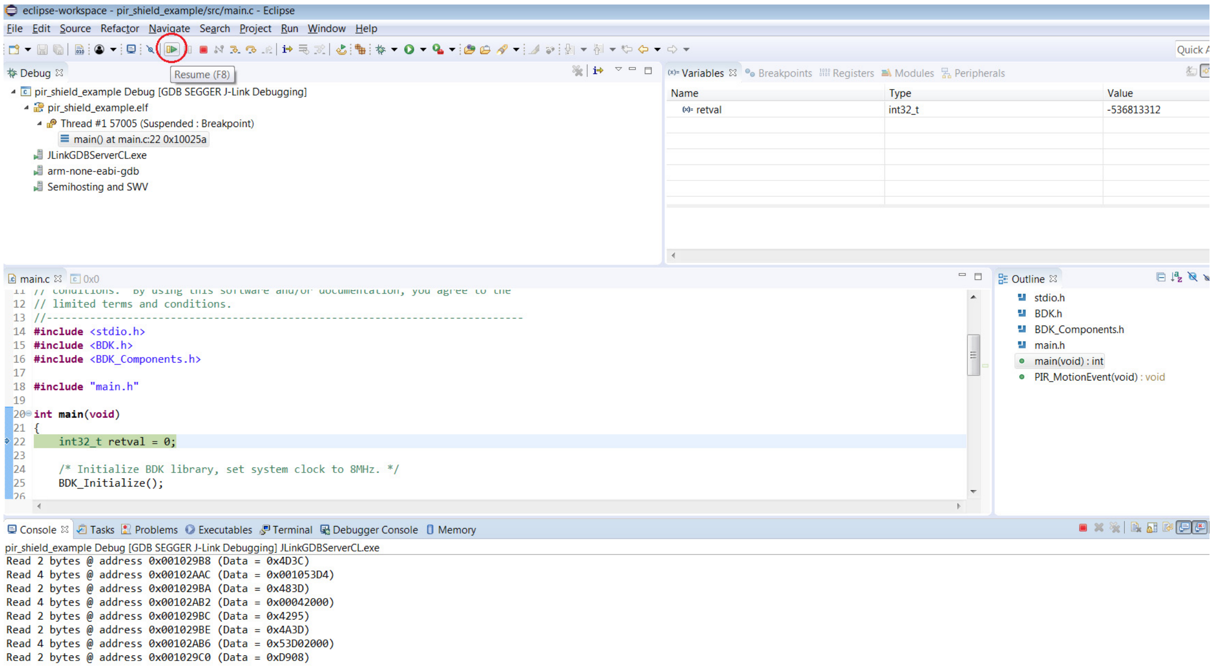
5. On the Debugger tab, set RSL10 as the device name. Click Debug to launch the code.



6. For application debugging, confirm perspective switch by clicking Yes.



7. The debug session is now launched. Click Resume (F8) to start the target CPU.



Logging/Debugging

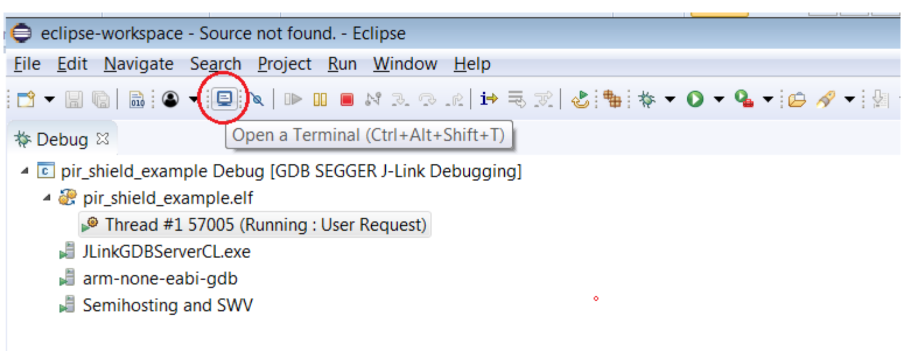
The following options are available to log/debug the downloaded firmware:

- Eclipse RTT Console
- J−Link RTT
- AX8052F100 UART−SPI bridge

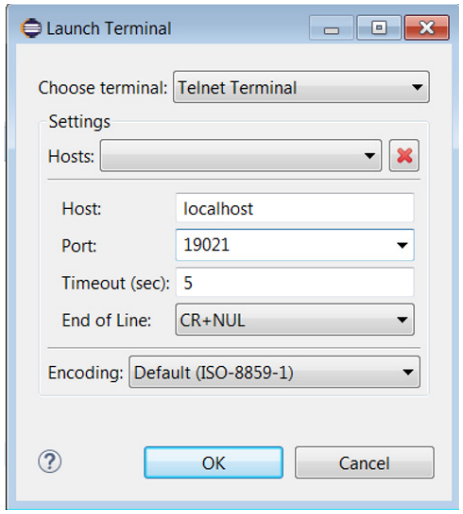
This section provides instructions for each of the above options.

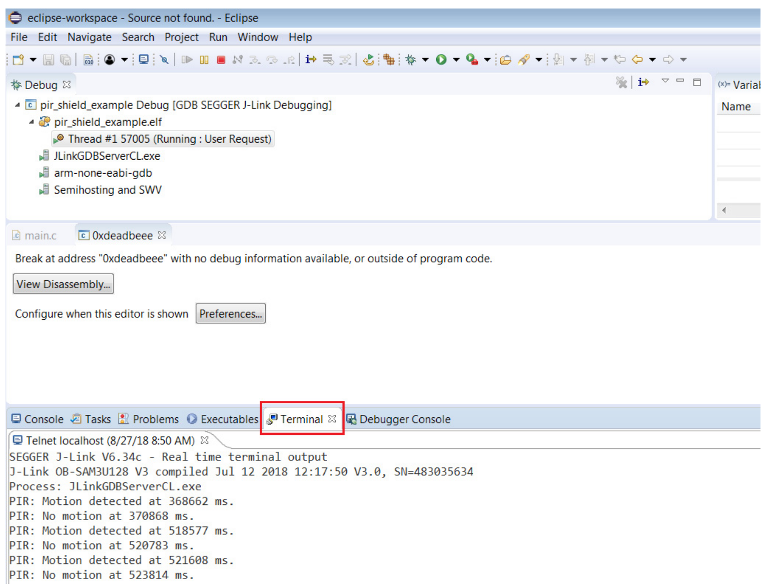
Using Eclipse Console

1. Click the Open a Terminal Icon



2. Enter the values shown below and launch the session. The incoming events are printed on the terminal window.

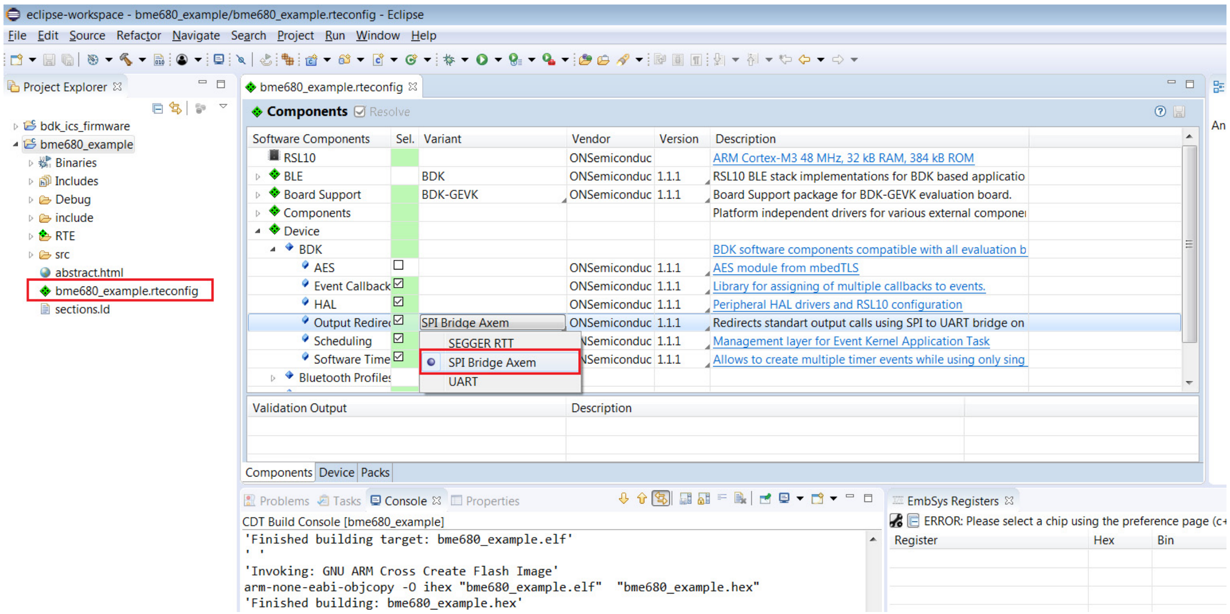




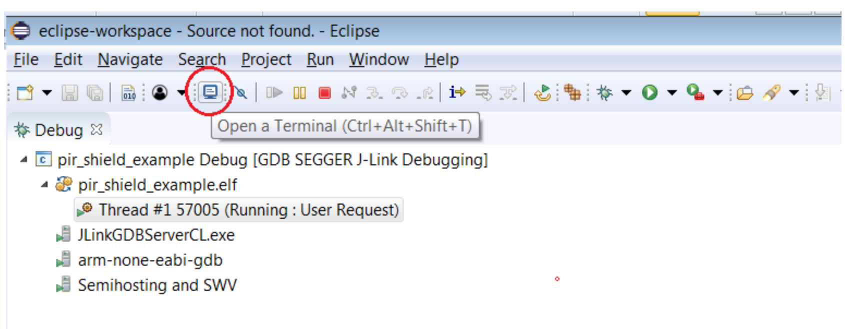
Using Eclipse Serial Console via UART−SPI Bridge.

When you do not want to use the Segger RTT viewer as serial console, the BDK−GEVK board is equipped with UART−SPI uC AX8052F100 flashed with special firmware, taking care of the entire serial communication with values returned on Terminal.

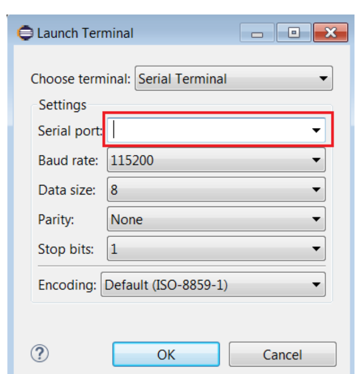
3. Click on the example’s rteconfig file and choose under *Device/BDK/Output redirection*, SPI Bridge AXEM. Save, compile and flash the whole project.

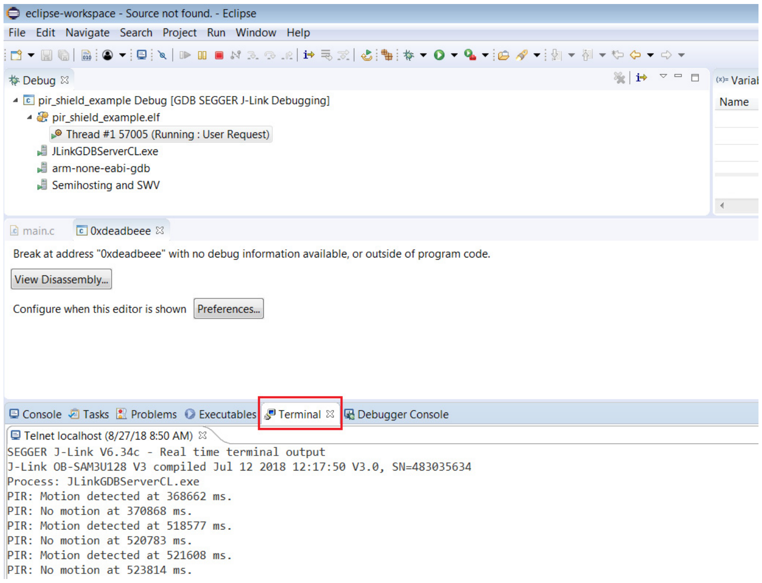


4. When the project runs, Click the **Open a Terminal** Icon.

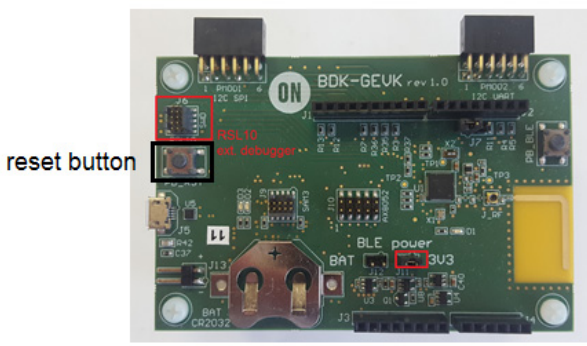


5. Enter the appropriate COM port as shown below, and launch the session. The incoming events are entered on the terminal window.



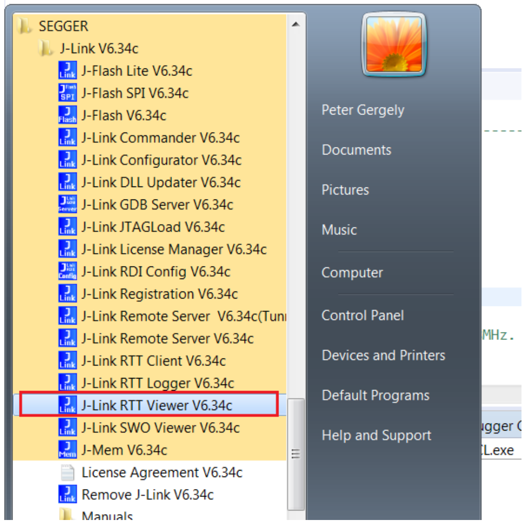


NOTE: You may reset (PB_RST) the BDK−GEVK (shown below) to launch the RTT terminal without needing to launch Eclipse.

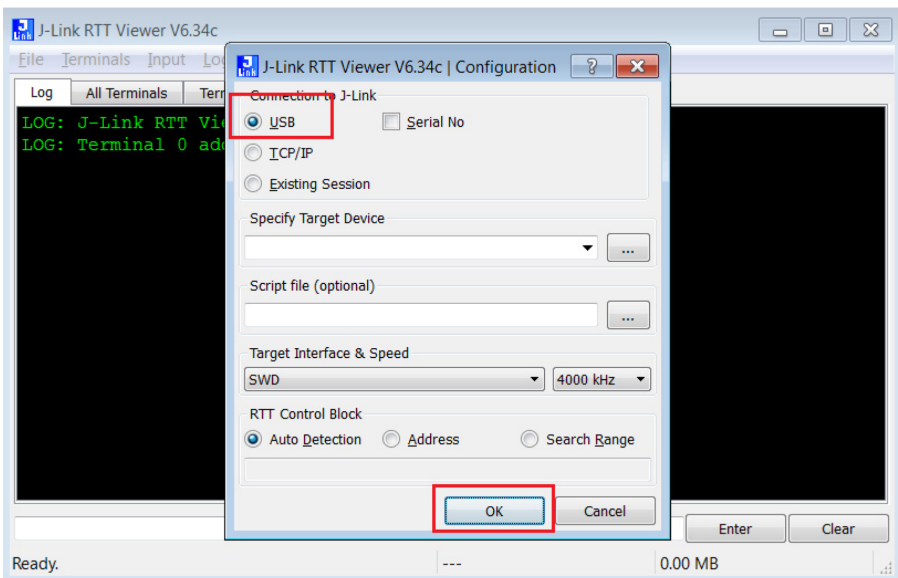


Using J−Link RTT

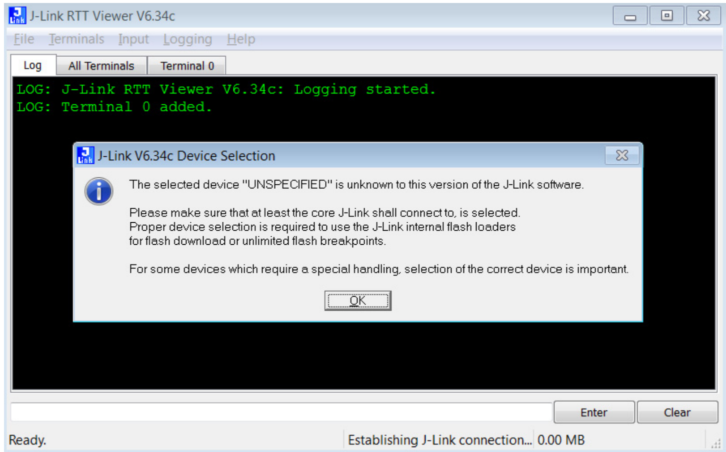
6. After step 14 is done, open J−Link RTT viewer (should be installed when J−Link software package was installed per Step 2).

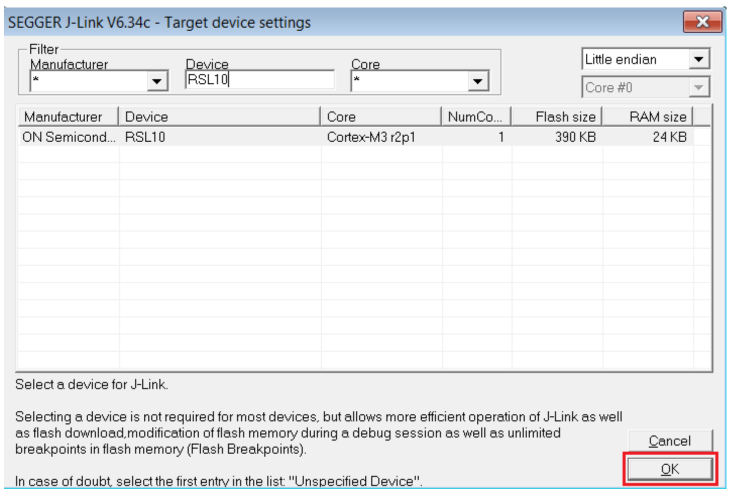


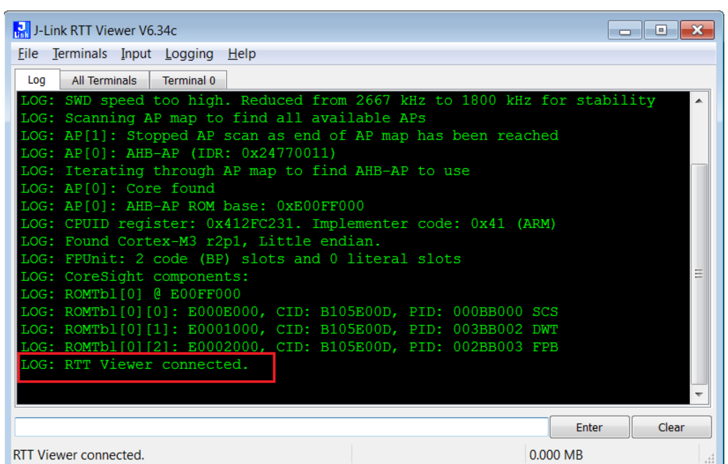
7. Select USB and click OK.

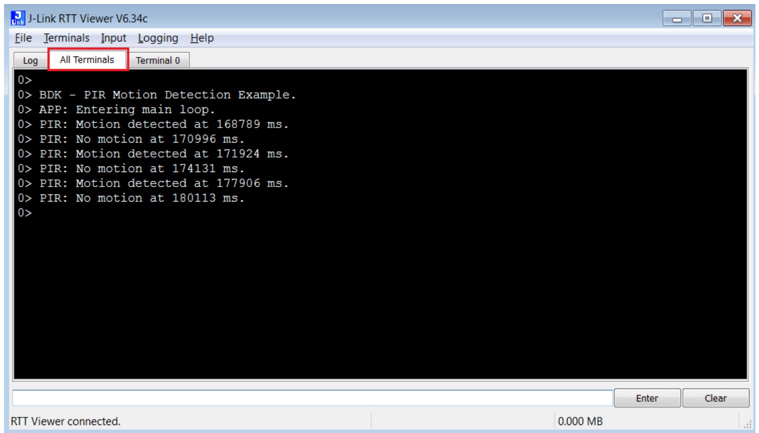


8. RTT prompts you to select the appropriate microcontroller. Select RSL10 and click OK. The serial terminal is ready to use and the events from RSL10 can be observed by clicking the All Terminals Window.

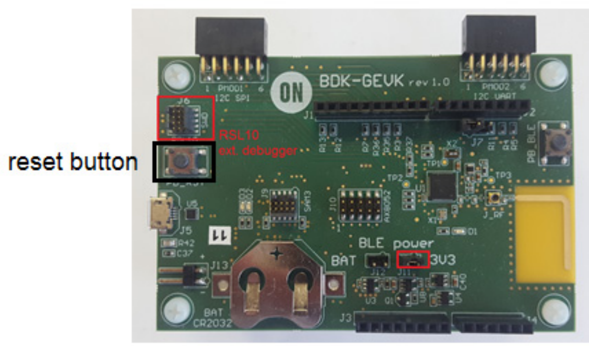








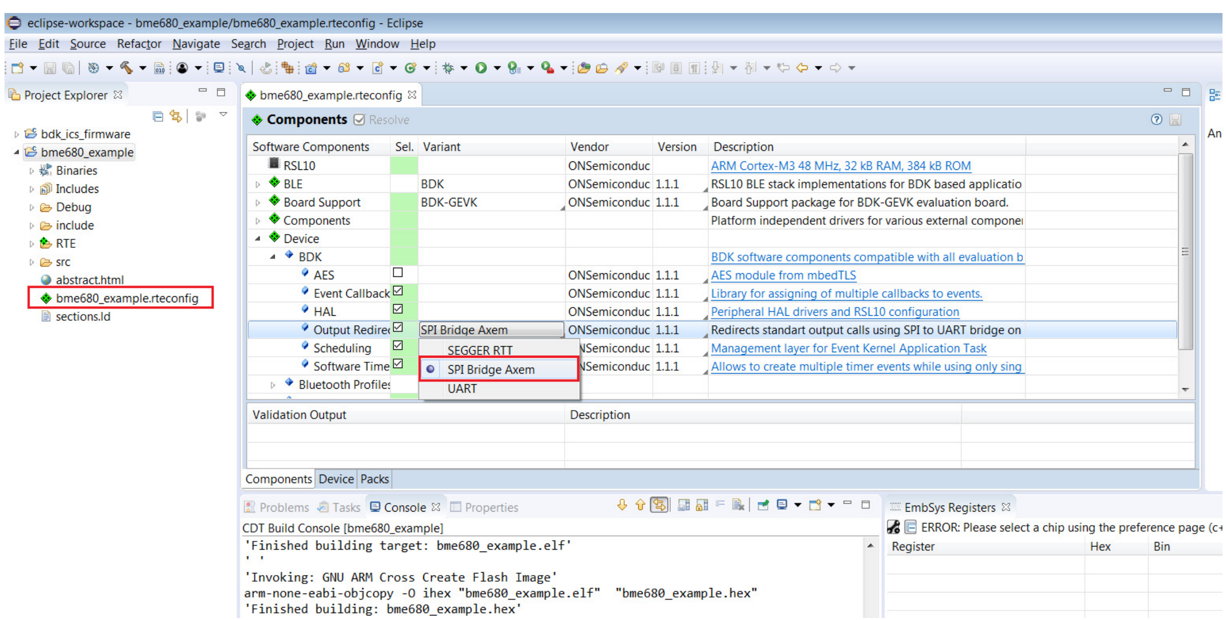
NOTE: You may reset (PB_RST) the BDK−GEVK (shown below) to launch RTT terminal without needing to launch Eclipse.



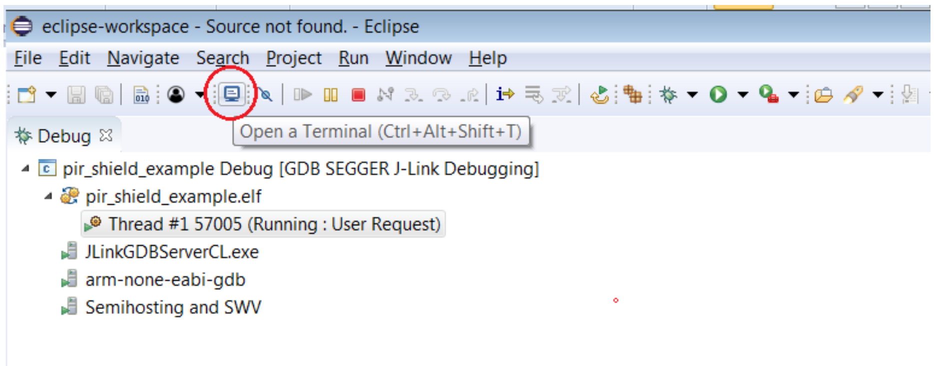
Using Eclipse Serial Console via UART−SPI Bridge

The BDK−GEVK board is equipped with UART−SPI microcontroller AX8052F100 flashed with special firmware, to enable serial communication with values returned to Terminal.

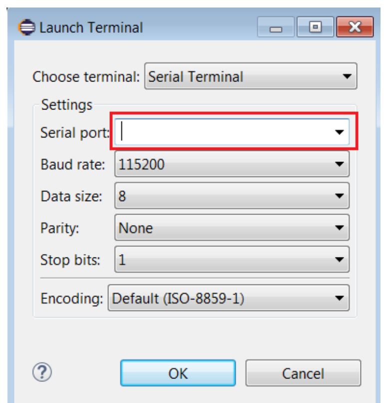
9. Click on example's rteconfig file and choose "SPI Bridge AXEM" under *Device/BDK/Output redirection*. Save, compile and flash the whole project.

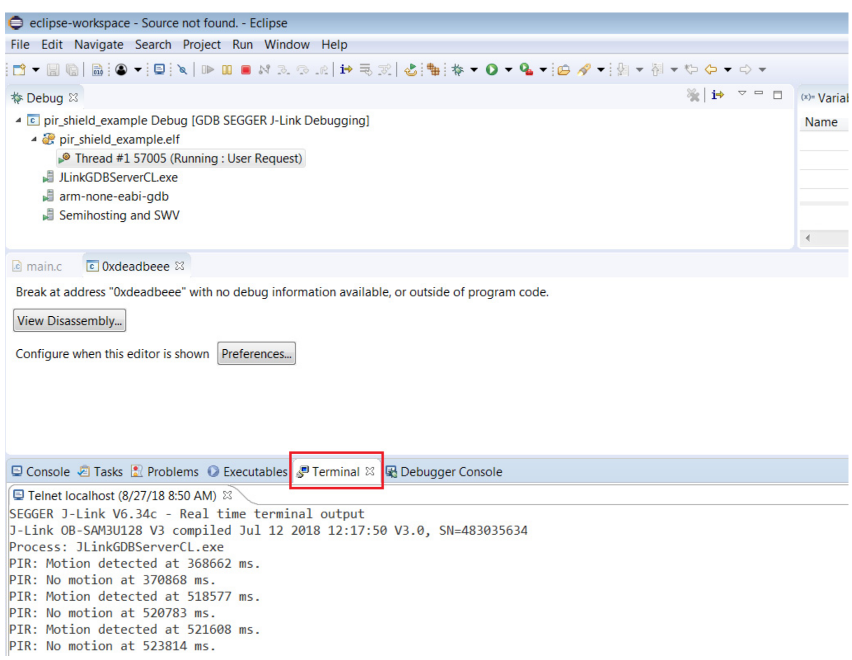


10. When the project runs, Click the Open a Terminal Icon.



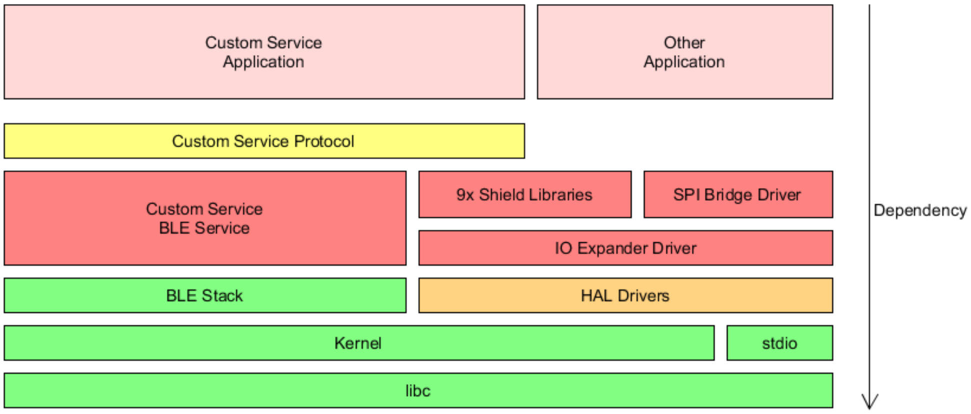
11. Enter the appropriate COM port as shown below and launch the session. The incoming events are printed on the terminal window.





SOFTWARE ORGANIZATION

For users modifying the example code and building new projects, the following sections detail the B−IDK software organization. The stack overview is shown below.



B−IDK CMSIS Software Organization

CMSIS pack and the associated software components handle multiple evaluation boards as different bundles of the standardized Board Support Cclass.

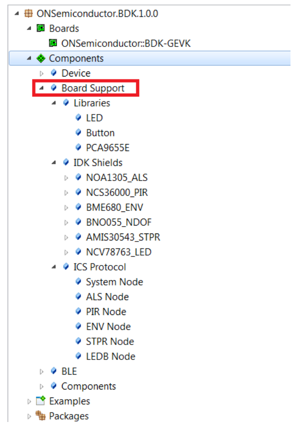
- This bundle shows only components supported by ON Semiconductor for a given board
- No confusing component variants

Common libraries and HAL are in a separate group within the Device class

Cbundle	Cclass	Cgroup	Csub	Cvariant	Description
BDK-GEVK	Board Support				Board support package for BDK-GEVK evalution board
		Libraries			Board specific libraries
			LED		On-board LED support
			Button		On-board push button support
			PCA9655E		16-bit I2C IO Expander library
		IDK Shields			Support for Arduino / PMOD extension boards
			PIR-GEVB		PIR Motion detection using NCS36000
			ALS-GEVB		Measure Ambient light levels using NOA1305 ambient light sensor
			MULTI-SENSE-GEVB	rev2.1	Combines 3 sensors: BME680, BNO055, NOA1305
			BLDC-GEVK		
			D-LED-B-GEVK		
			D-STPR-GEVK		
		ICS Protocol			Libraries that allow connected BLE devices to take control over sensors / actuators using ICS Service.
			System Node		Protocol implementation and sytem node used by other sensor / actuator nodes.
			PIR Node		Exposes motion data provided by NCS36000 from PIR-GEVB
			ALS Node		Exposes ambient light levels measured by NOA1305 from ALS-GEVB
			ENV Node		Exposes environmental data measured by BME680 from MULTI-SENSE-GEVB
			AO Node		Exposes absolute orientation measured by BNO055 from MULTI-SENSE-GEVB
			STPR Node		Allows remote control of two stepper motors connected to D-STPR-GEVB.
			LEDB Node		Allows remote control of two power LEDs connected to D-LED-B-GEVK
			BLDC Node		Allows to remote control BLDC motor connected to BLDC-GEVB.
	Components				Platform independent software drivers for controlling of various external IC.
		LED Driver			
			NCV78763		Dual LED Driver and Power Ballast, for Automotive Front Lighting, 1.6 A, 2nd Generation
		Ambient Light Sensor			
			NOA1305		Ambient Light Sensor with I2C Interface and DarkCurrent Compensation
		Motor Driver			
			AMIS-3054		Micro−stepping stepper motor driver with SPI interface for bipolar stepper motors
			LV8907UW		Sensor-less Three-phaseBrushless DC MotorController, with GateDrivers, for Automotive
		Environmental Sensor			
			bme680		Low power gas, pressure, temperature & humidity sensor
		Motion sensor			
			bno055		Intelligent 9-axis absolute orientation sensor
		Touch Sensor			
			LC717A00AR		Capacitance-Digital-Converter for Electrostatic Capacitive Touch Sensors
	Device				
		BDK			
			HAL		RSL10 Peripheral abstraction layers for BDK applications.
			Scheduling		Event Kernel wrapper for BDK applications.
			Software Timer		Allows to create multiple timer events while using only single hardware timer.
			Event Callback		Library for executing multiple event handlers when an event occurs.
			Output Redirection		Redirects standard library output calls (printf, ...) to specified channel
				SEGGER RTT	Output is transmitted using UART peripheral
				UART	Output is transmitted over SWD using the on-board or external J-LINK deug probe
			AES		
BDK	BLE				
		Peripheral Server			
			Battery Service		Exposes current battery level to connected client and application.
			ICS Service		IDK Custom Service used to transmit sensor data using ICS Protocol library.
			Peripheral Server		BLE Peripheral Server implementation for BDK applications.

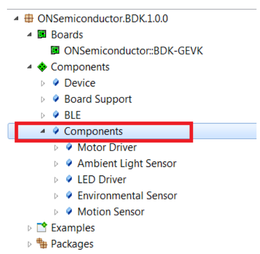
Board Support

- Libraries to support BDK−GEVK, GPIO Expander, Various daughter cards and custom protocol (required for the mobile app)



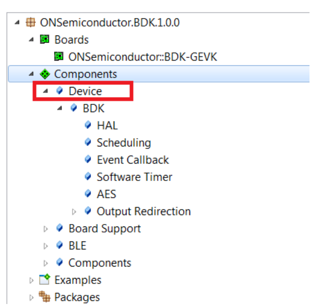
Components

- Libraries attached to board support



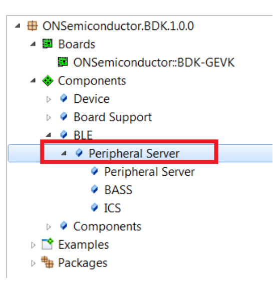
Device

- Abstraction layers for interfaces, timers, AES, serial re−direction, etc.



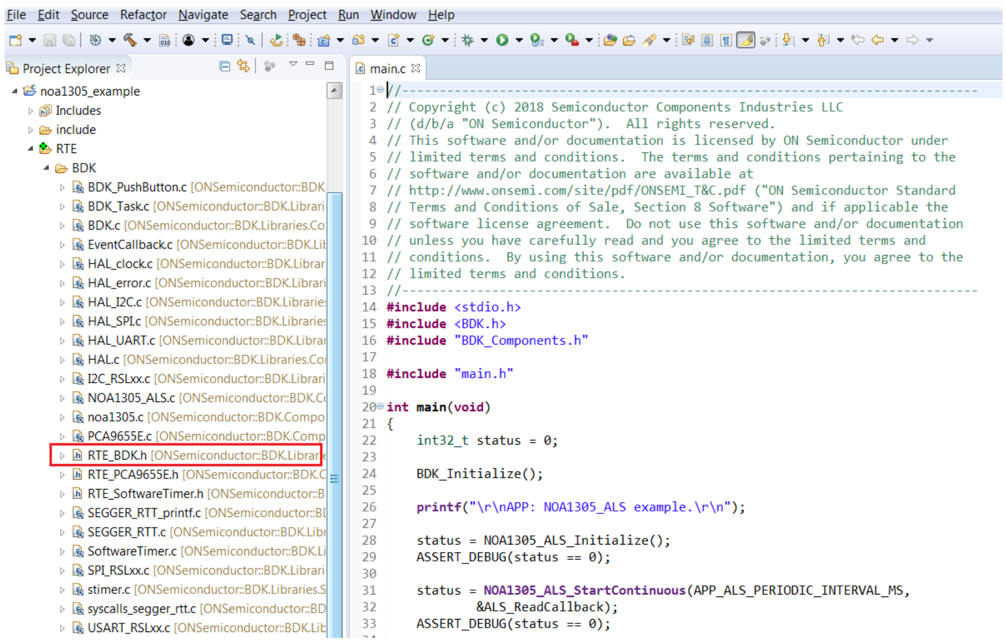
BLE

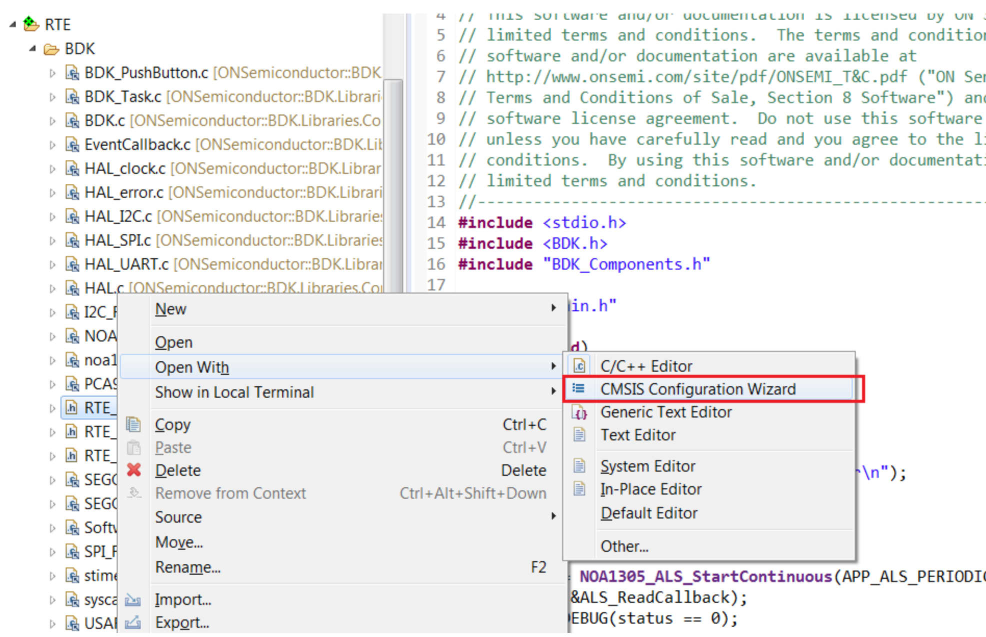
- Peripheral Server Support



CONFIGURATION SETUP

System settings can be configured directly from within the CMSIS pack. Each example is equipped with basic system configuration that covers three main categories. These are accessible in the RTE/BDK folder within the project. Each system configuration starts with “RTE_”. As shown below, opening the RTE_... header files using the CMSIS configuration wizard (right click on the header file), displays the configuration table. Various application specific parameters can be set. This allows pre−configuration of RSL10 without the need for explicit programming.

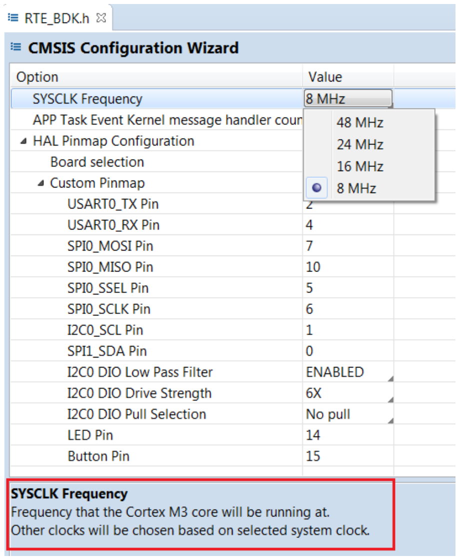




A brief description on the header files is given below.

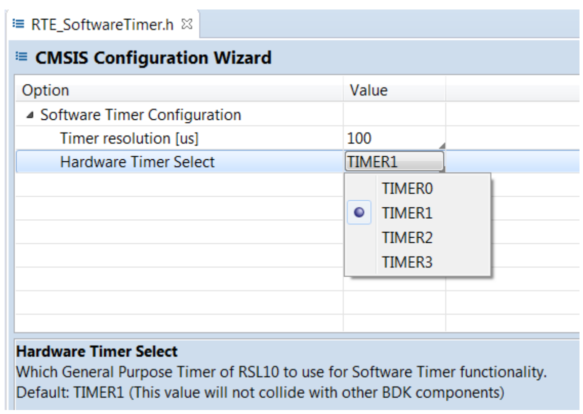
RTE_BDK.h

Parameters such as system clock frequency and the board that feature RSL10 (default set to BDK−GEVK), etc. can be set. Descriptions of each of these parameters are also provided.



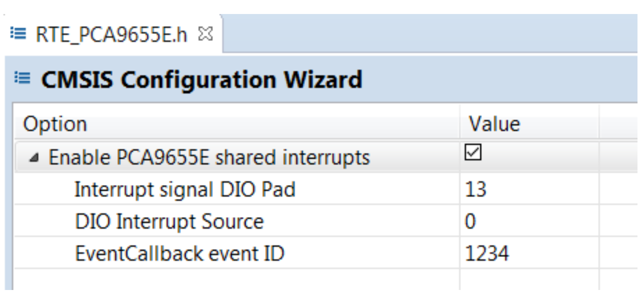
RTE_Software_Timer.h

Various timers (4) supported by RSL10 can be configured by invoking the CMSIS configuration wizard on this header file. Timer 1 is used for B−IDK components.



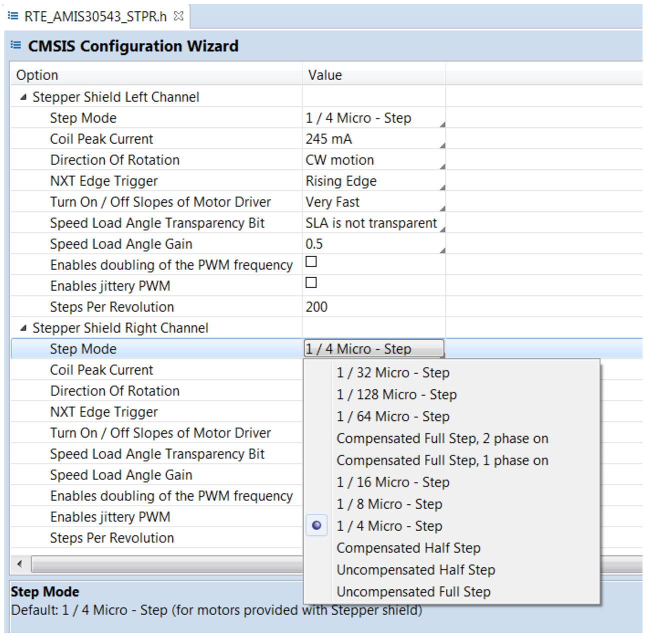
RTE_PCA9655.h

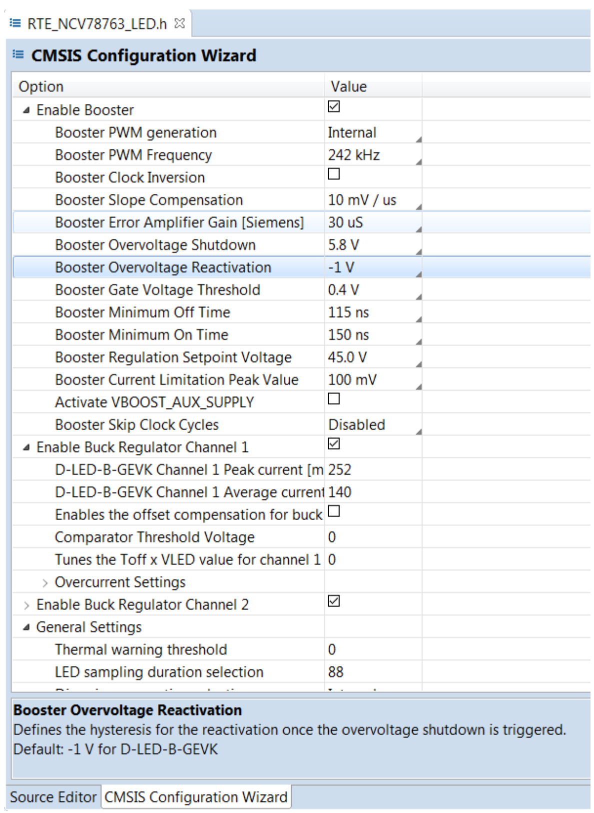
PCA9655 is the GPIO expander chip assembled on most daughter cards to expand interface functionality. Parameters related to this chip can be set here.



RTE_x.h

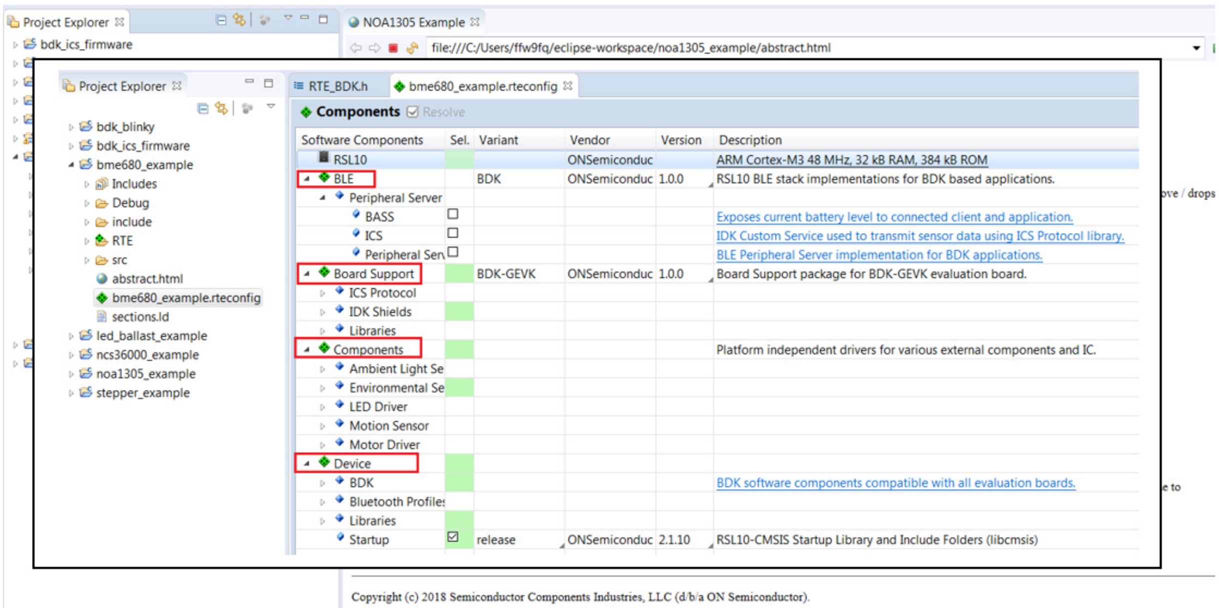
In addition to configuring system settings, all the supported daughter cards' parameters can be configured directly using the configuration wizard, without the need for programming. Once the parameters are changed per the application requirements, saving, rebuilding and flashing the project will let the new parameters take effect. Examples for the stepper and LED ballast daughter cards are shown below. Other daughter cards can be configured in a similar fashion.





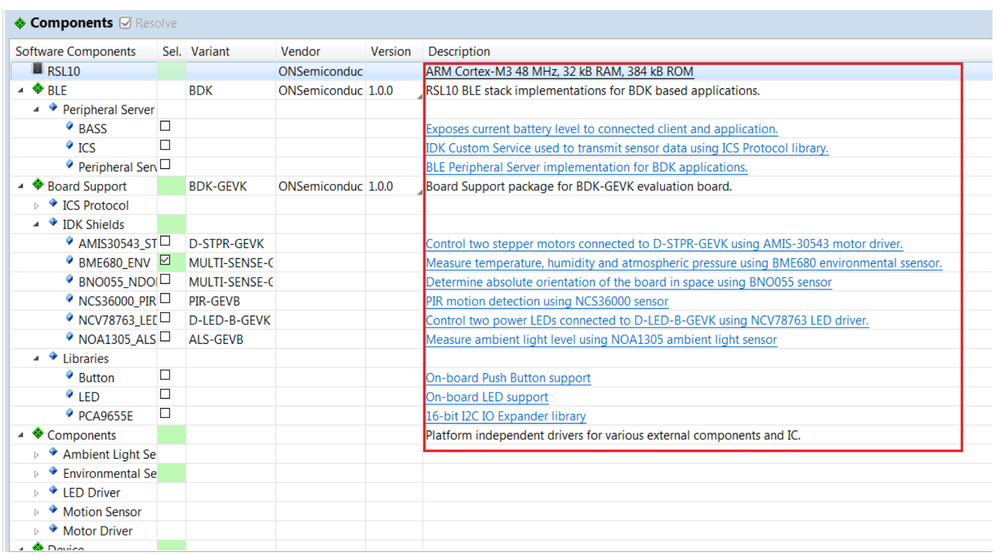
DOCUMENTATION

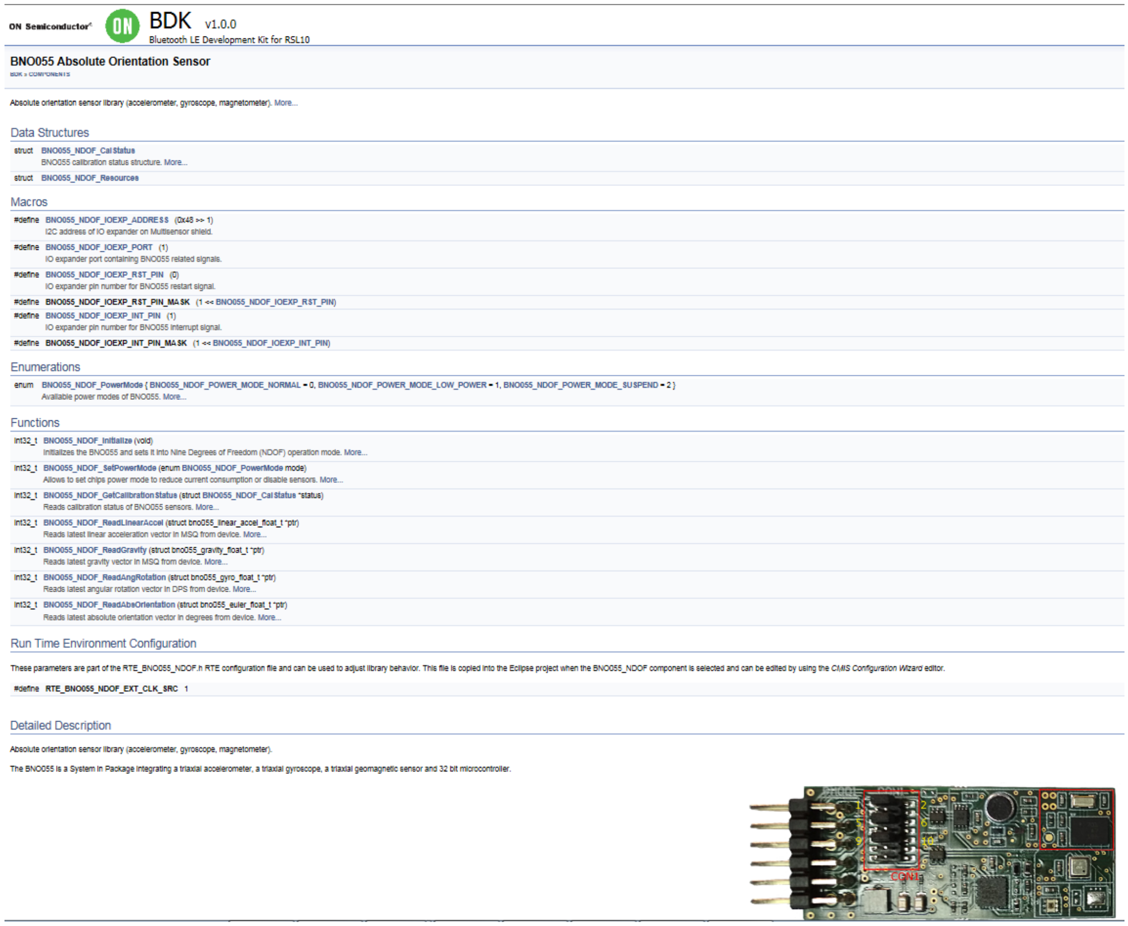
Detailed documentation of all functions, code, APIs, HALs is part of the CMSIS package. Every use case (for a particular daughter card, service, etc.) copied into the workspace has its own manual with key description in the abstract.html page. URL Information and orderable part numbers are also provided as shown below.



*.rteconfig

The *.rteconfig file lists the software components within the CMSIS pack as described in the B_IDK CMSIS Software Organization section. To access the components, double click *.rteconfig file. Extensive help is provided under the description tab.





ON Semiconductor® BDK v1.0.0
Bluetooth LE Development Kit for RSL10

BNO055 Absolute Orientation Sensor
BDK » COMPONENTS

Absolute orientation sensor library (accelerometer, gyroscope, magnetometer). More...

Data Structures

struct BNO055_NDOF_CalStatus
BNO055 calibration status structure. More...
struct BNO055_NDOF_Resources

Macros

#define BNO055_NDOF_IOEXP_ADDRESS (0x48 >> 1)
I2C address of IO expander on Multisensor shield.
#define BNO055_NDOF_IOEXP_PORT (1)
IO expander port containing BNO055 related signals.
#define BNO055_NDOF_IOEXP_RST_PIN (0)
IO expander pin number for BNO055 restart signal.
#define BNO055_NDOF_IOEXP_RST_PIN_MASK (1 << BNO055_NDOF_IOEXP_RST_PIN)
#define BNO055_NDOF_IOEXP_INT_PIN (1)
IO expander pin number for BNO055 interrupt signal.
#define BNO055_NDOF_IOEXP_INT_PIN_MASK (1 << BNO055_NDOF_IOEXP_INT_PIN)

Enumerations

enum BNO055_NDOF_PowerMode { BNO055_NDOF_POWER_MODE_NORMAL = 0, BNO055_NDOF_POWER_MODE_LOW_POWER = 1, BNO055_NDOF_POWER_MODE_SUSPEND = 2 }
Available power modes of BNO055. More...

Functions

int32_t BNO055_NDOF_Initialize (void)
Initializes the BNO055 and sets it into Nine Degrees of Freedom (NDOF) operation mode. More...
int32_t BNO055_NDOF_SetPowerMode (enum BNO055_NDOF_PowerMode mode)
Allows to set chips power mode to reduce current consumption or disable sensors. More...
int32_t BNO055_NDOF_GetCalibrationStatus (struct BNO055_NDOF_CalStatus *status)
Reads calibration status of BNO055 sensors. More...
int32_t BNO055_NDOF_ReadLinearAccel (struct bno055_linear_accel_float_t *ptr)
Reads latest linear acceleration vector in MSQ from device. More...
int32_t BNO055_NDOF_ReadGravity (struct bno055_gravity_float_t *ptr)
Reads latest gravity vector in MSQ from device. More...
int32_t BNO055_NDOF_ReadAngRotation (struct bno055_gyro_float_t *ptr)
Reads latest angular rotation vector in DPS from device. More...
int32_t BNO055_NDOF_ReadAbsOrientation (struct bno055_euler_float_t *ptr)
Reads latest absolute orientation vector in degrees from device. More...

Run Time Environment Configuration

These parameters are part of the RTE_BNO055_NDOF.h RTE configuration file and can be used to adjust library behavior. This file is copied into the Eclipse project when the BNO055_NDOF component is selected and can be edited by using the *CMIS Configuration Wizard* editor.

#define RTE_BNO055_NDOF_EXT_CLK_SRC 1

Detailed Description

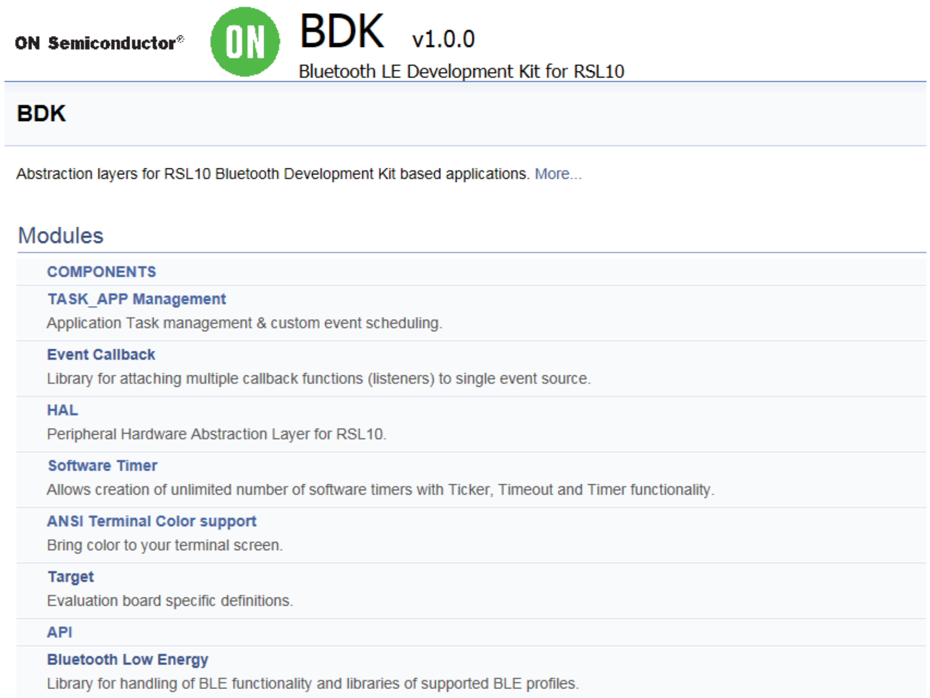
Absolute orientation sensor library (accelerometer, gyroscope, magnetometer).

The BNO055 is a System in Package integrating a triaxial accelerometer, a triaxial gyroscope, a triaxial geomagnetic sensor and 32 bit microcontroller.

Main Help Page

The main help page is accessible via Device/BDK, visible for all use cases in *.rteconfig file. It's further divided into various modules as shown below.

Software Components	Sel.	Variant	Vendor	Version	Description
RSL10			ONSemiconduc		ARM Cortex-M3 48 MHz, 32 kB RAM, 384 kB ROM
BLE		BDK	ONSemiconduc	1.0.0	RSL10 BLE stack implementations for BDK based applications.
Board Support		BDK-GEVK	ONSemiconduc	1.0.0	Board Support package for BDK-GEVK evaluation board.
Components					Platform independent drivers for various external components and IC.
Device					
BDK					BDK software components compatible with all evaluation boards.
AES	☐		ONSemiconduc	1.0.0	AES module from mbedTLS
Event Callback	☑		ONSemiconduc	1.0.0	Library for assigning of multiple callbacks to events.
HAL	☑		ONSemiconduc	1.0.0	Peripheral HAL drivers and RSL10 configuration
Output Redirection	☑	SEGGER RTT	ONSemiconduc	1.0.0	Redirects standart output calls using SEGGER RTT
Scheduling	☑		ONSemiconduc	1.0.0	Management layer for Event Kernel Application Task
Software Timer	☑		ONSemiconduc	1.0.0	Allows to create multiple timer events while using only single hardware timer.
Bluetooth Profiles					
Libraries					
Startup	☑	release	ONSemiconduc	2.1.10	RSL10-CMSIS Startup Library and Include Folders (libcmsis)



ON Semiconductor®

BDK v1.0.0

Bluetooth LE Development Kit for RSL10

BDK

Abstraction layers for RSL10 Bluetooth Development Kit based applications. More...

Modules

COMPONENTS

TASK_APP Management
Application Task management & custom event scheduling.

Event Callback
Library for attaching multiple callback functions (listeners) to single event source.

HAL
Peripheral Hardware Abstraction Layer for RSL10.

Software Timer
Allows creation of unlimited number of software timers with Ticker, Timeout and Timer functionality.

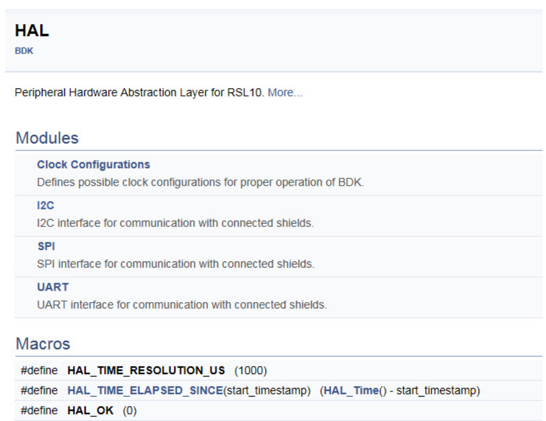
ANSI Terminal Color support
Bring color to your terminal screen.

Target
Evaluation board specific definitions.

API

Bluetooth Low Energy
Library for handling of BLE functionality and libraries of supported BLE profiles.

Sub−sections may be expanded for further information (Ex: HAL interfaces shown below)



HAL
BDK

Peripheral Hardware Abstraction Layer for RSL10. More...

Modules

Clock Configurations
Defines possible clock configurations for proper operation of BDK.

I2C
I2C interface for communication with connected shields.

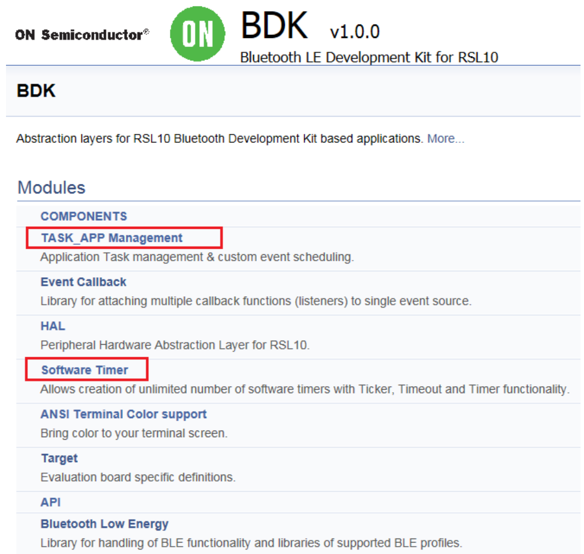
SPI
SPI interface for communication with connected shields.

UART
UART interface for communication with connected shields.

Macros

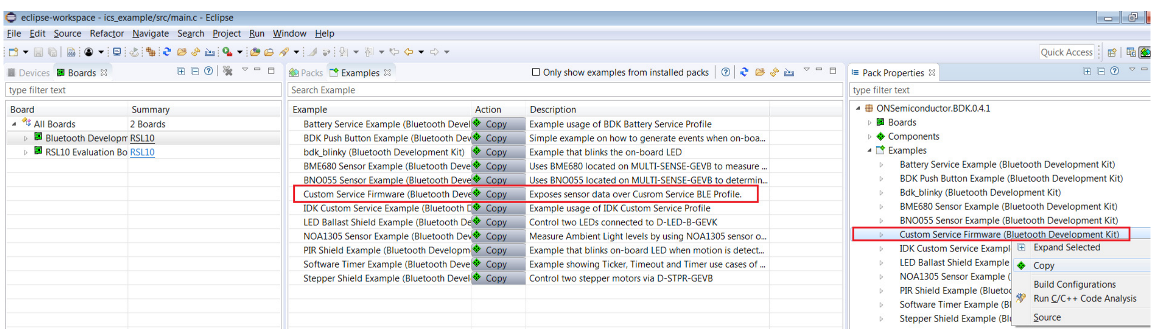
```
#define HAL_TIME_RESOLUTION_US (1000)
#define HAL_TIME_ELAPSED_SINCE(start_timestamp) (HAL_Time() - start_timestamp)
#define HAL_OK (0)
```

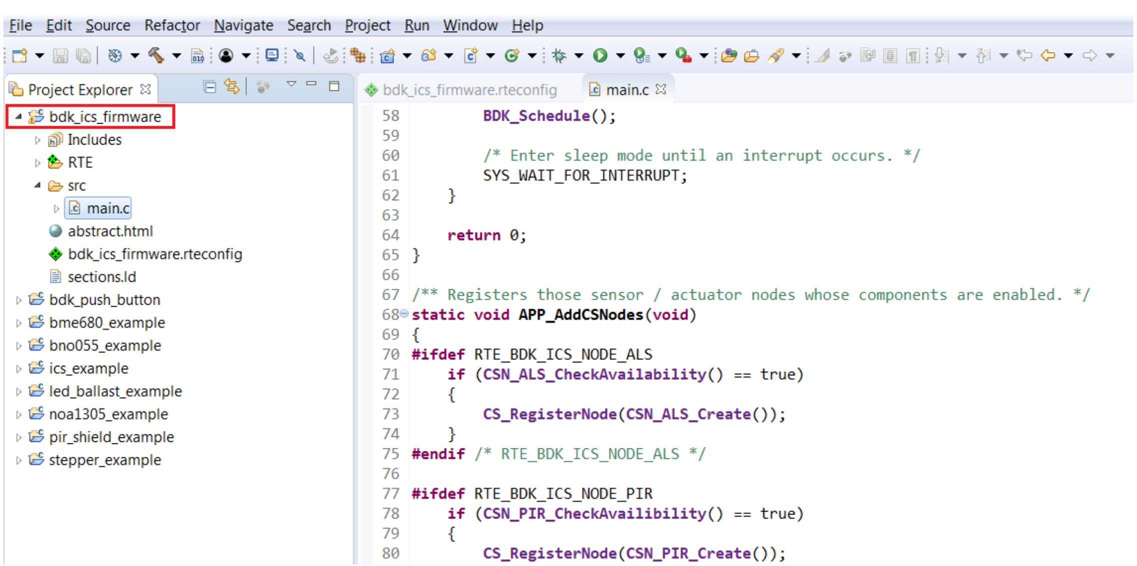
B−IDK also provides software timers and applications task manager abstraction layers to enable management of specific tasks and timing within the event kernel.



Custom Service Firmware

In order to read sensor data and control actuators connected to the BDK−GEVK from the RSL10 Sense and Control mobile app, the Custom Service Firmware must be downloaded onto the BDK−GEVK. This firmware can be found as Custom Service Firmware under examples in the CMSIS pack.





PUBLICATION ORDERING INFORMATION

LITERATURE FULFILLMENT:
Email Requests to: orderlit@onsemi.com

ON Semiconductor Website: www.onsemi.com

TECHNICAL SUPPORT
North American Technical Support:
Voice Mail: 1 800−282−9855 Toll Free USA/Canada
Phone: 011 421 33 790 2910

Europe, Middle East and Africa Technical Support:
Phone: 00421 33 790 2910
For additional information, please contact your local Sales Representative

Данный компонент на территории Российской Федерации

Вы можете приобрести в компании MosChip.

Для оперативного оформления запроса Вам необходимо перейти по данной ссылке:

http://moschip.ru/get-element

Вы можете разместить у нас заказ для любого Вашего проекта, будь то серийное производство или разработка единичного прибора.

В нашем ассортименте представлены ведущие мировые производители активных и пассивных электронных компонентов.

Нашей специализацией является поставка электронной компонентной базы двойного назначения, продукции таких производителей как XILINX, Intel (ex.ALTERA), Vicor, Microchip, Texas Instruments, Analog Devices, Mini-Circuits, Amphenol, Glenair.

Сотрудничество с глобальными дистрибьюторами электронных компонентов, предоставляет возможность заказывать и получать с международных складов практически любой перечень компонентов в оптимальные для Вас сроки.

На всех этапах разработки и производства наши партнеры могут получить квалифицированную поддержку опытных инженеров.

Система менеджмента качества компании отвечает требованиям в соответствии с ГОСТ Р ИСО 9001, ГОСТ РВ 0015-002 и ЭС РД 009

Офис по работе с юридическими лицами:

105318, г.Москва, ул.Щербаковская д.3, офис 1107, 1118, ДЦ «Щербаковский»

Телефон: +7 495 668-12-70 (многоканальный)

Факс: +7 495 668-12-70 (доб.304)

E-mail: info@moschip.ru

Skype отдела продаж:
moschip.ru
moschip.ru_4
moschip.ru_6
moschip.ru_9