Customer training workshop: Device Configurator_Pins

TRAVEO^{TM} T2G CYT4BF series Microcontroller Training
V1.0.0 2022-12

Please read the **Important notice and warnings** at the end of this document

Scope of work

- This document helps application developers understand how to use the Device Configurator pins as part of creating a ModusToolbox™ (MTB) application
  - The Device Configurator pins are part of a collection of tools included with the MTB software. It provides a GUI to configure the pin-related resources.

- ModusToolbox™ tools package version: 3.0.0
- Device Configurator version: 4.0
- Device
  - The TRAVEO^{TM} T2G CYT4BFBCH device is used in this code example.
- Board
  - The TRAVEO^{TM} T2G KIT_T2G-B-H_EVK board is used for testing.

Introduction

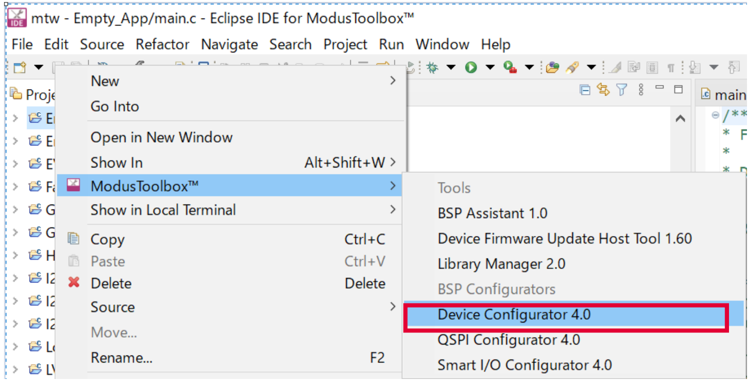
- **GPIO has the following features:**
  - Analog and digital input and output capabilities
  - Eight drive strength modes
  - Separate port read and write registers
  - Edge-triggered interrupts on rising edge, falling edge, or on both the edges, on all GPIO
  - Slew rate control
  - Hold mode for latching previous state (used to retain the I/O state in DeepSleep mode)
  - Selectable CMOS, TTL, and automotive input buffer mode
  - Smart I/O provides the ability to perform Boolean functions in the I/O signal path
  - See the Architecture technical reference manual for GPIO details

Launch the Device Configurator

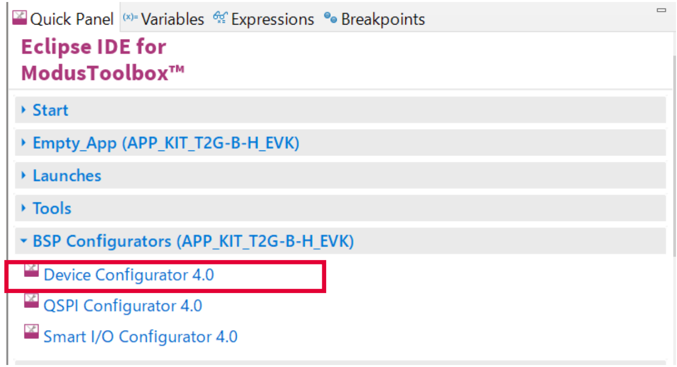
› From Eclipse IDE

– You can launch the Device Configurator by either of the following methods:

a) Right-click on the project in the Project Explorer and select **ModusToolbox™** > **Device Configurator <version>**



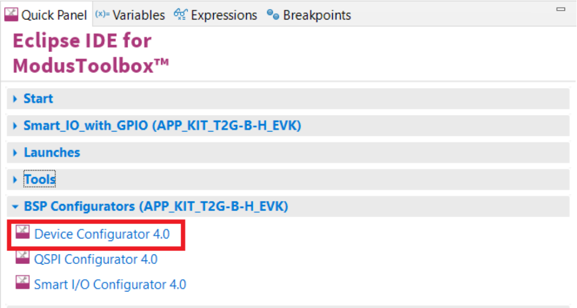
b) Click the Device Configurator link in the Quick Panel


Quick Panel
Variables
Expressions
Breakpoints
Eclipse IDE for ModusToolbox™
Start
Empty_App (APP_KIT_T2G-B-H_EVK)
Launches
Tools
BSP Configurators (APP_KIT_T2G-B-H_EVK)
Device Configurator 4.0
QSPI Configurator 4.0
Smart I/O Configurator 4.0

Launch the Device Configurator (contd.)

› From Device Configurator

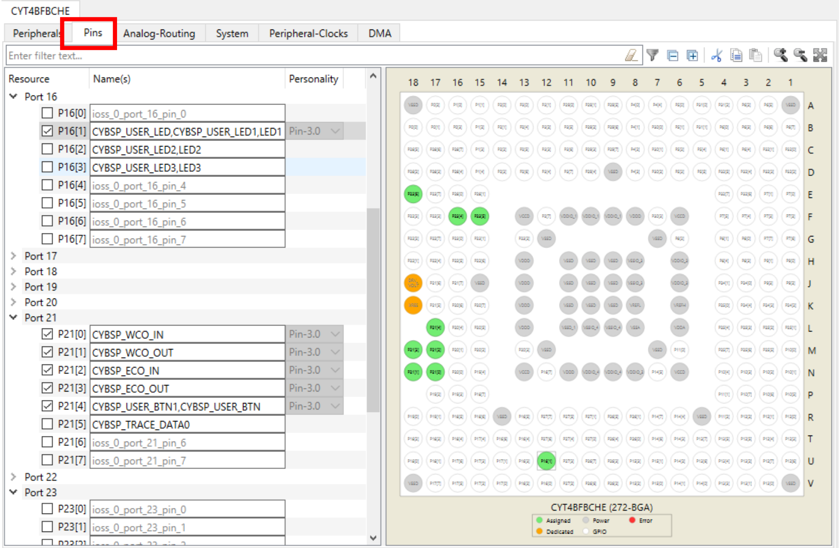
1) Open the Device configurator



Eclipse IDE for ModusToolbox™

- Start
- Smart_IO_with_GPIO (APP_KIT_T2G-B-H_EVK)
- Launches
- Tools
- BSP Configurators (APP_KIT_T2G-B-H_EVK)
  - Device Configurator 4.0
  - QSPI Configurator 4.0
  - Smart I/O Configurator 4.0

2) On the Pins tab, check each port resource



Resource	Name(s)	Personality
⌄ Port 16		
□ P16[0]	ioss_0_port_16_pin_0	
☑ P16[1]	CYBSP_USER_LED,CYBSP_USER_LED1,LED1	Pin-3.0
□ P16[2]	CYBSP_USER_LED2,LED2	
□ P16[3]	CYBSP_USER_LED3,LED3	
□ P16[4]	ioss_0_port_16_pin_4	
□ P16[5]	ioss_0_port_16_pin_5	
□ P16[6]	ioss_0_port_16_pin_6	
□ P16[7]	ioss_0_port_16_pin_7	
› Port 17		
› Port 18		
› Port 19		
› Port 20		
⌄ Port 21		
☑ P21[0]	CYBSP_WCO_IN	Pin-3.0
☑ P21[1]	CYBSP_WCO_OUT	Pin-3.0
☑ P21[2]	CYBSP_ECO_IN	Pin-3.0
☑ P21[3]	CYBSP_ECO_OUT	Pin-3.0
☑ P21[4]	CYBSP_USER_BTN1,CYBSP_USER_BTN	Pin-3.0
□ P21[5]	CYBSP_TRACE_DATA0	
□ P21[6]	ioss_0_port_21_pin_6	
□ P21[7]	ioss_0_port_21_pin_7	
› Port 22		
⌄ Port 23		
□ P23[0]	ioss_0_port_23_pin_0	
□ P23[1]	ioss_0_port_23_pin_1	

Device Configurator Pin config view

› Device Configurator pins

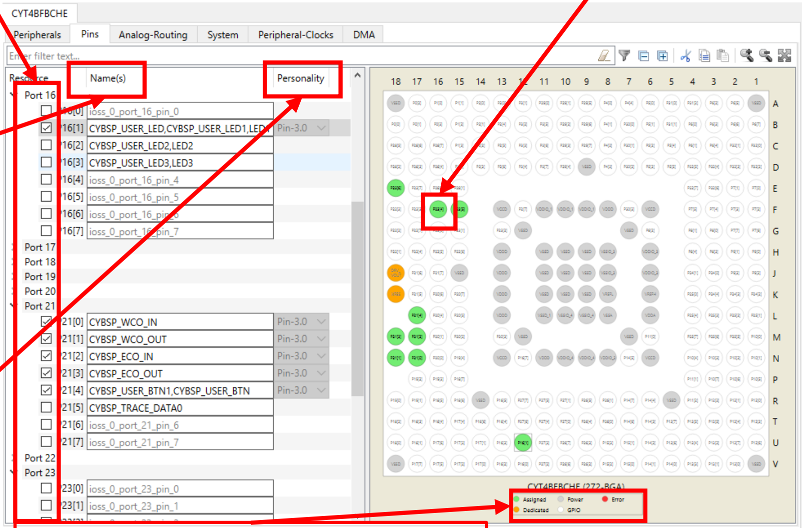
– The Pins tab/tree is where you enable all the pin related resources

All available pins are shown in an expandable tree, arranged by the port number.
You can check the port numbers that are set to enabled.

Name(s)
- This displays the current resource name(s). This is an editable field where you can specify optional, alternate names for this resource. This is also used in generated code.
- Enter any string in this field. The tool converts the name into a legal C identifier and replaces nonlegal characters with underscores.
- If entering more than one name, use a comma-separated list.

Personality
- Some peripherals, such as Serial Communication Block (SCB) and Timer, Counter, Pulse Width Modulator (TCPWM), have a pull-down menu to select a specific personality, such as UART, SPI, or I2C
- Some peripherals have multiple personality versions from which you can select
- Some peripherals have a read-only field that only shows the name of this resource's personality file

You can also enable/disable a pin by double-clicking it in the diagram



Resource	Name(s)	Personality
Port 16		
□ 16[0]	ioss_0_port_16_pin_0	
☑ 16[1]	CYBSP_USER_LED,CYBSP_USER_LED1,LED1	Pin-3.0
□ 16[2]	CYBSP_USER_LED2,LED2	
□ 16[3]	CYBSP_USER_LED3,LED3	
□ 16[4]	ioss_0_port_16_pin_4	
□ 16[5]	ioss_0_port_16_pin_5	
□ 16[6]	ioss_0_port_16_pin_6	
□ 16[7]	ioss_0_port_16_pin_7	
Port 17		
Port 18		
Port 19		
Port 20		
Port 21		
☑ 21[0]	CYBSP_WCO_IN	Pin-3.0
☑ 21[1]	CYBSP_WCO_OUT	Pin-3.0
☑ 21[2]	CYBSP_ECO_IN	Pin-3.0
☑ 21[3]	CYBSP_ECO_OUT	Pin-3.0
☑ 21[4]	CYBSP_USER_BTN1,CYBSP_USER_BTN	Pin-3.0
□ 21[5]	CYBSP_TRACE_DATA0	
□ 21[6]	ioss_0_port_21_pin_6	
□ 21[7]	ioss_0_port_21_pin_7	
Port 22		
Port 23		
□ 23[0]	ioss_0_port_23_pin_0	
□ 23[1]	ioss_0_port_23_pin_1	

Pin states are shown in different colors:
Green – Assigned, Orange – Dedicated
Grey – Power, White – GPIO
Red – Error

Device Configurator pin config view (contd.)

› Device Configurator pin configuration

You can select a port

You can enable/disable a pin by double-clicking it in the diagram

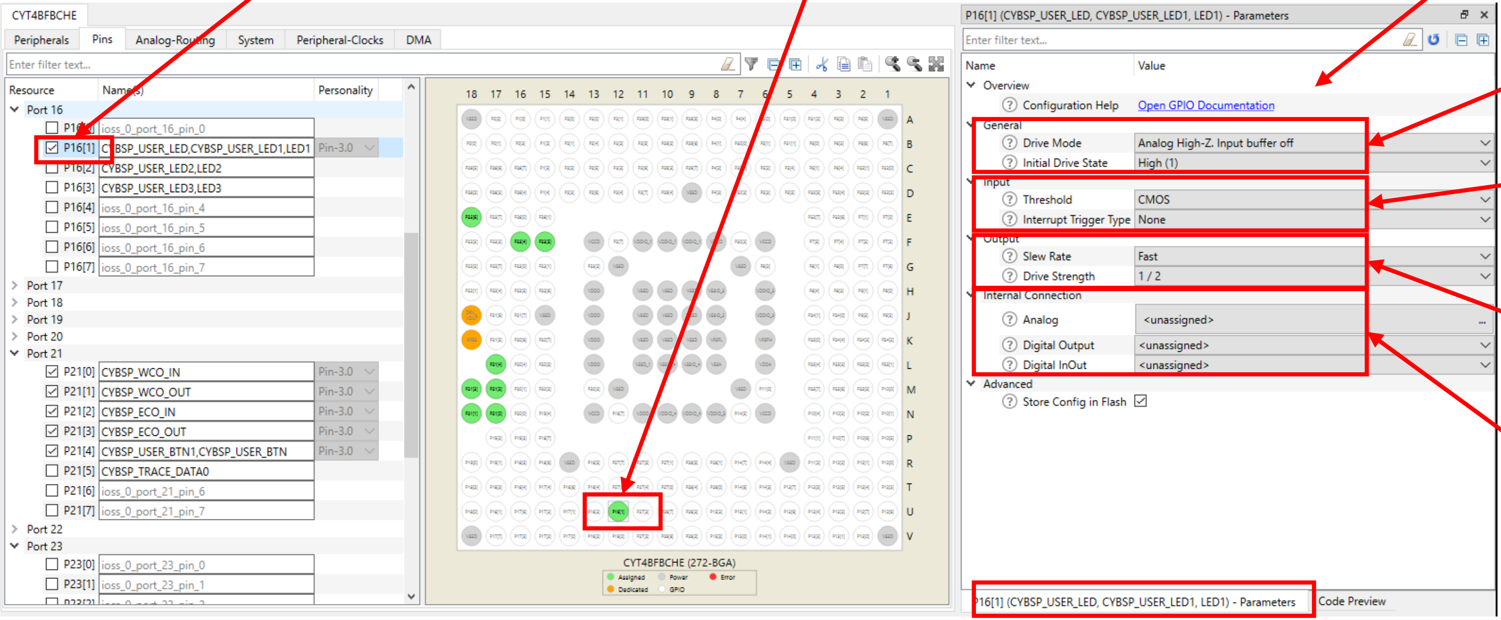
You can select the parameters

Configure Drive mode and initial Drive state

Configure Threshold and Interrupt Trigger Type for input

Configure Slew Rate and Drive Strength for output

Configure internal function Input/Output

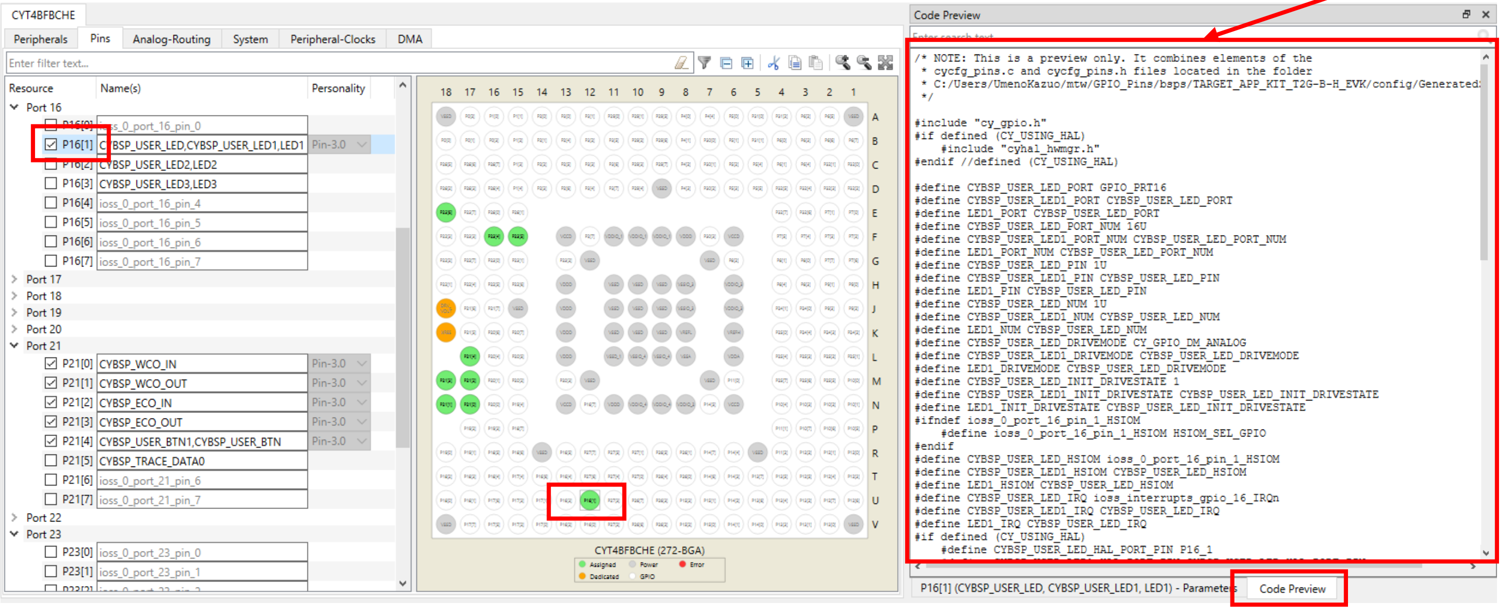


Resource	Name(s)	Personality
⌄ Port 16		
□ P16[0]	ioss_0_port_16_pin_0	
☑ P16[1]	CYBSP_USER_LED,CYBSP_USER_LED1,LED1	Pin-3.0
□ P16[2]	CYBSP_USER_LED2,LED2	
□ P16[3]	CYBSP_USER_LED3,LED3	
□ P16[4]	ioss_0_port_16_pin_4	
□ P16[5]	ioss_0_port_16_pin_5	
□ P16[6]	ioss_0_port_16_pin_6	
□ P16[7]	ioss_0_port_16_pin_7	
› Port 17		
› Port 18		
› Port 19		
› Port 20		
⌄ Port 21		
☑ P21[0]	CYBSP_WCO_IN	Pin-3.0
☑ P21[1]	CYBSP_WCO_OUT	Pin-3.0
☑ P21[2]	CYBSP_ECO_IN	Pin-3.0
☑ P21[3]	CYBSP_ECO_OUT	Pin-3.0
☑ P21[4]	CYBSP_USER_BTN1,CYBSP_USER_BTN	Pin-3.0
□ P21[5]	CYBSP_TRACE_DATA0	
□ P21[6]	ioss_0_port_21_pin_6	
□ P21[7]	ioss_0_port_21_pin_7	
› Port 22		
⌄ Port 23		
□ P23[0]	ioss_0_port_23_pin_0	
□ P23[1]	ioss_0_port_23_pin_1	

Name	Value
⌄ Overview	
Configuration Help	Open GPIO Documentation
⌄ General	
Drive Mode	Analog High-Z. Input buffer off
Initial Drive State	High (1)
⌄ Input	
Threshold	CMOS
Interrupt Trigger Type	None
⌄ Output	
Slew Rate	Fast
Drive Strength	1 / 2
⌄ Internal Connection	
Analog	<unassigned>
Digital Output	<unassigned>
Digital InOut	<unassigned>
⌄ Advanced	
Store Config in Flash	☑

Device Configurator Pin config view (contd.)

› **Device Configurator pin configuration**



Resource	Name(s)	Personality
Port 16		
□ P16[0]	ioss_0_port_16_pin_0	
☑ P16[1]	CYBSP_USER_LED,CYBSP_USER_LED1,LED1	Pin-3.0
□ P16[2]	CYBSP_USER_LED2,LED2	
□ P16[3]	CYBSP_USER_LED3,LED3	
□ P16[4]	ioss_0_port_16_pin_4	
□ P16[5]	ioss_0_port_16_pin_5	
□ P16[6]	ioss_0_port_16_pin_6	
□ P16[7]	ioss_0_port_16_pin_7	
Port 17		
Port 18		
Port 19		
Port 20		
Port 21		
☑ P21[0]	CYBSP_WCO_IN	Pin-3.0
☑ P21[1]	CYBSP_WCO_OUT	Pin-3.0
☑ P21[2]	CYBSP_ECO_IN	Pin-3.0
☑ P21[3]	CYBSP_ECO_OUT	Pin-3.0
☑ P21[4]	CYBSP_USER_BTN1,CYBSP_USER_BTN	Pin-3.0
□ P21[5]	CYBSP_TRACE_DATA0	
□ P21[6]	ioss_0_port_21_pin_6	
□ P21[7]	ioss_0_port_21_pin_7	
Port 22		
Port 23		
□ P23[0]	ioss_0_port_23_pin_0	
□ P23[1]	ioss_0_port_23_pin_1	
□ P23[2]	ioss_0_port_23_pin_2	

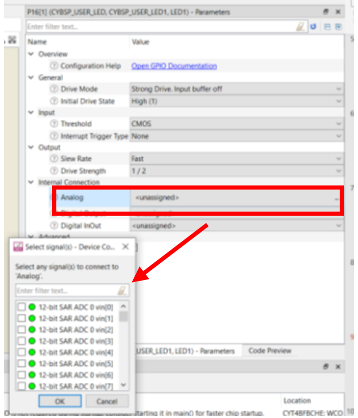
```
/* NOTE: This is a preview only. It combines elements of the
 * cycfg_pins.c and cycfg_pins.h files located in the folder
 * C:/Users/UmenoKazuo/mtw/GPIO_Pins/bsps/TARGET_APP_KIT_T2G-B-H_EVK/config/Generated
 */

#include "cy_gpio.h"
#if defined (CY_USING_HAL)
    #include "cyhal_hwmgr.h"
#endif //defined (CY_USING_HAL)

#define CYBSP_USER_LED_PORT GPIO_PRT16
#define CYBSP_USER_LED1_PORT CYBSP_USER_LED_PORT
#define LED1_PORT CYBSP_USER_LED_PORT
#define CYBSP_USER_LED_PORT_NUM 16U
#define CYBSP_USER_LED1_PORT_NUM CYBSP_USER_LED_PORT_NUM
#define LED1_PORT_NUM CYBSP_USER_LED_PORT_NUM
#define CYBSP_USER_LED_PIN 1U
#define CYBSP_USER_LED1_PIN CYBSP_USER_LED_PIN
#define LED1_PIN CYBSP_USER_LED_PIN
#define CYBSP_USER_LED_NUM 1U
#define CYBSP_USER_LED1_NUM CYBSP_USER_LED_NUM
#define LED1_NUM CYBSP_USER_LED_NUM
#define CYBSP_USER_LED_DRIVEMODE CY_GPIO_DM_ANALOG
#define CYBSP_USER_LED1_DRIVEMODE CYBSP_USER_LED_DRIVEMODE
#define LED1_DRIVEMODE CYBSP_USER_LED_DRIVEMODE
#define CYBSP_USER_LED_INIT_DRIVESTATE 1
#define CYBSP_USER_LED1_INIT_DRIVESTATE CYBSP_USER_LED_INIT_DRIVESTATE
#define LED1_INIT_DRIVESTATE CYBSP_USER_LED_INIT_DRIVESTATE
#ifndef ioss_0_port_16_pin_1_HSIOM
    #define ioss_0_port_16_pin_1_HSIOM HSIOM_SEL_GPIO
#endif
#define CYBSP_USER_LED_HSIOM ioss_0_port_16_pin_1_HSIOM
#define CYBSP_USER_LED1_HSIOM CYBSP_USER_LED_HSIOM
#define LED1_HSIOM CYBSP_USER_LED_HSIOM
#define CYBSP_USER_LED_IRQ ioss_interrupts_gpio_16_IRQn
#define CYBSP_USER_LED1_IRQ CYBSP_USER_LED_IRQ
#define LED1_IRQ CYBSP_USER_LED_IRQ
#if defined (CY_USING_HAL)
    #define CYBSP_USER_LED_HAL_PORT_PIN P16_1
```

Code preview automatically generates the selected parameter configuration

Device Configurator pin config view (contd.)

› Internal connection

- Device Configurator can configure inputs and outputs for internal functions
- You can connect to Analog, Digital Output, and Digital Input
- Click the value box to select the connectable function input/output in the pull-down menu
- The following is an example (see the device datasheet for connectable function input and output)



P16[1] (CYBSP_USER_LED, CYBSP_USER_LED1, LED1) - Parameters

Enter filter text...

Name	Value
Overview	
Configuration Help	Open GPIO Documentation
General	
Drive Mode	Strong Drive. Input buffer off
Initial Drive State	High (1)
Input	
Threshold	CMOS
Interrupt Trigger Type	None
Output	
Slew Rate	Fast
Drive Strength	1 / 2
Internal Connection	
Analog	<unassigned>
Digital InOut	<unassigned>
Advanced	

Select signal(s) - Device Co...

Select any signal(s) to connect to 'Analog'.

Enter filter text...

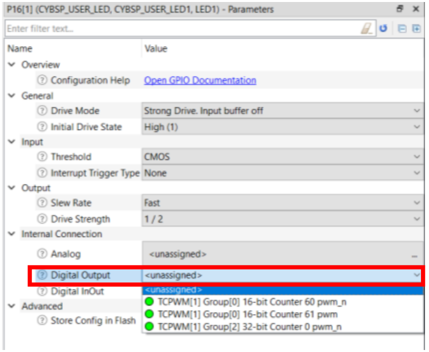
- □ ● 12-bit SAR ADC 0 vin[0]
- □ ● 12-bit SAR ADC 0 vin[1]
- □ ● 12-bit SAR ADC 0 vin[2]
- □ ● 12-bit SAR ADC 0 vin[3]
- □ ● 12-bit SAR ADC 0 vin[4]
- □ ● 12-bit SAR ADC 0 vin[5]
- □ ● 12-bit SAR ADC 0 vin[6]
- □ ● 12-bit SAR ADC 0 vin[7]

OK Cancel

USER_LED1, LED1) - Parameters Code Preview

Location

tarting it in main() for faster chip startup. CYT4BFBCHE: WCO

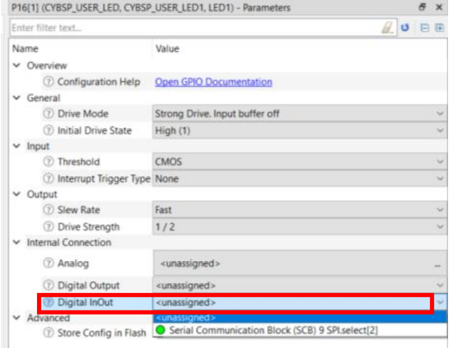


P16[1] (CYBSP_USER_LED, CYBSP_USER_LED1, LED1) - Parameters

Enter filter text...

Name	Value
Overview	
Configuration Help	Open GPIO Documentation
General	
Drive Mode	Strong Drive. Input buffer off
Initial Drive State	High (1)
Input	
Threshold	CMOS
Interrupt Trigger Type	None
Output	
Slew Rate	Fast
Drive Strength	1 / 2
Internal Connection	
Analog	<unassigned>
Digital Output	<unassigned>
Digital InOut	
Advanced	
Store Config in Flash	

- <unassigned>
- ● TCPWM[1] Group[0] 16-bit Counter 60 pwm_n
- ● TCPWM[1] Group[0] 16-bit Counter 61 pwm
- ● TCPWM[1] Group[2] 32-bit Counter 0 pwm_n



P16[1] (CYBSP_USER_LED, CYBSP_USER_LED1, LED1) - Parameters

Enter filter text...

Name	Value
Overview	
Configuration Help	Open GPIO Documentation
General	
Drive Mode	Strong Drive. Input buffer off
Initial Drive State	High (1)
Input	
Threshold	CMOS
Interrupt Trigger Type	None
Output	
Slew Rate	Fast
Drive Strength	1 / 2
Internal Connection	
Analog	<unassigned>
Digital Output	<unassigned>
Digital InOut	<unassigned>
Advanced	
Store Config in Flash	

- <unassigned>
- ● Serial Communication Block (SCB) 9 SPI.select[2]

- If you configured function inputs and outputs, you must also configure the selected functions

Quick start

- **To use the Device Configurator for Pins setting**
  - Launch the Device Configurator.
  - Use the various menus to configure signals.
  - The Device Configurator generates code into a "GeneratedSource" directory in your Eclipse IDE application, or in the same location you saved the *.modus file for non-IDE applications. That directory contains the necessary source (.c) and header (.h) files for the generated firmware, which uses the relevant driver APIs to configure the hardware.
  - Use the generated structures as input parameters for pin configuration in your application.
  - The generated structures are automatically configured in the *cybsp_init()* function. Therefore, the user does not need any specific action for pin configuration.

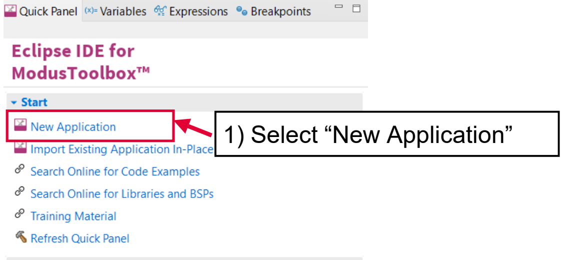
Use case

- Configure P16.1 as an output pin. It is assigned as user LED output.
- Configure P21.4 as an input pin. It is assigned as user button input.
- The LED turns on by pressing the user button, and the LED turns off by releasing the user button.

Device configurator configuration

› Create project

1) Click “New Application” in Quick Panel and open the Choose Board Support Package (BSP) window



Eclipse IDE for ModusToolbox™

▾ Start

- New Application
- Import Existing Application In-Place
- Search Online for Code Examples
- Search Online for Libraries and BSPs
- Training Material
- Refresh Quick Panel

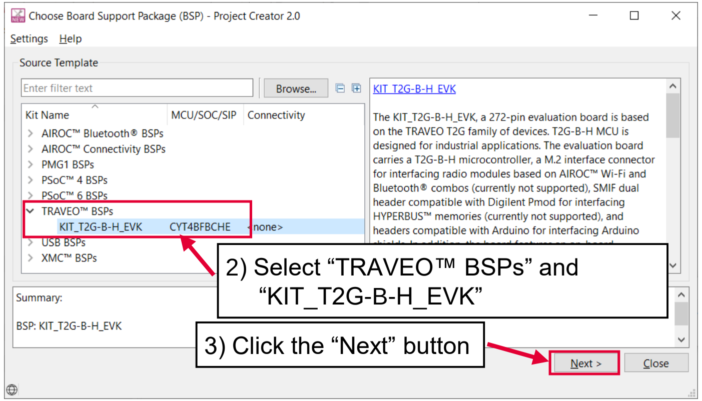
1) Select “New Application”

2) Select “TRAVEO™ BSPs” and “KIT_T2G-B-H_EVK”

3) Click the **Next** button and open the Application window

4) Check the “Empty App” option.
In this use case, change the application name to “PortPin_training”.

5) Click the **Create** button to start application creation



Choose Board Support Package (BSP) - Project Creator 2.0

Kit Name	MCU/SOC/SIP	Connectivity
› AIROC™ Bluetooth® BSPs		
› AIROC™ Connectivity BSPs		
› PMG1 BSPs		
› PSoC™ 4 BSPs		
› PSoC™ 6 BSPs		
⌄ TRAVEO™ BSPs		
KIT_T2G-B-H_EVK	CYT4BFBCHE	<none>
› USB BSPs		
› XMC™ BSPs		

KIT_T2G-B-H_EVK

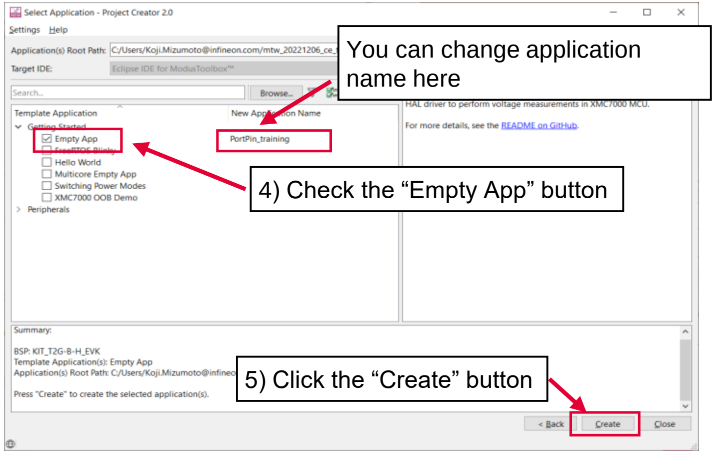
The KIT_T2G-B-H_EVK, a 272-pin evaluation board is based on the TRAVEO T2G family of devices. T2G-B-H MCU is designed for industrial applications. The evaluation board carries a T2G-B-H microcontroller, a M.2 interface connector for interfacing radio modules based on AIROC™ Wi-Fi and Bluetooth® combos (currently not supported), SMIF dual header compatible with Digilent Pmod for interfacing HYPERBUS™ memories (currently not supported), and headers compatible with Arduino for interfacing Arduino

Summary:

BSP: KIT_T2G-B-H_EVK

2) Select “TRAVEO™ BSPs” and “KIT_T2G-B-H_EVK”

3) Click the “Next” button



Select Application - Project Creator 2.0

Application(s) Root Path: C:/Users/Koji.Mizumoto@infineon.com/mtw_20221206_ce_

Target IDE: Eclipse IDE for ModusToolbox™

Template Application	New Application Name
⌄ Getting Started	
☑ Empty App	PortPin_training
☐ FreeRTOS Blinky	
☐ Hello World	
☐ Multicore Empty App	
☐ Switching Power Modes	
☐ XMC7000 OOB Demo	
› Peripherals	

HAL driver to perform voltage measurements in XMC7000 MCU.

For more details, see the README on GitHub.

Summary:

BSP: KIT_T2G-B-H_EVK
Template Application(s): Empty App
Application(s) Root Path: C:/Users/Koji.Mizumoto@infineo

Press "Create" to create the selected application(s).

You can change application name here

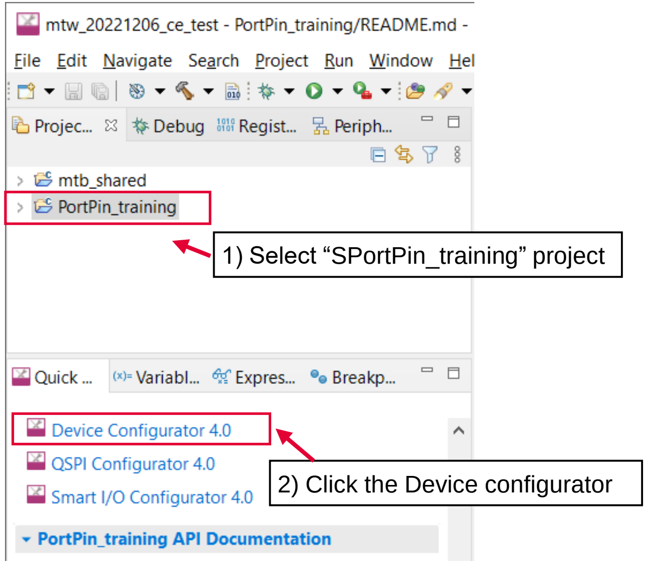
4) Check the “Empty App” button

5) Click the “Create” button

Device configurator configuration (contd.)

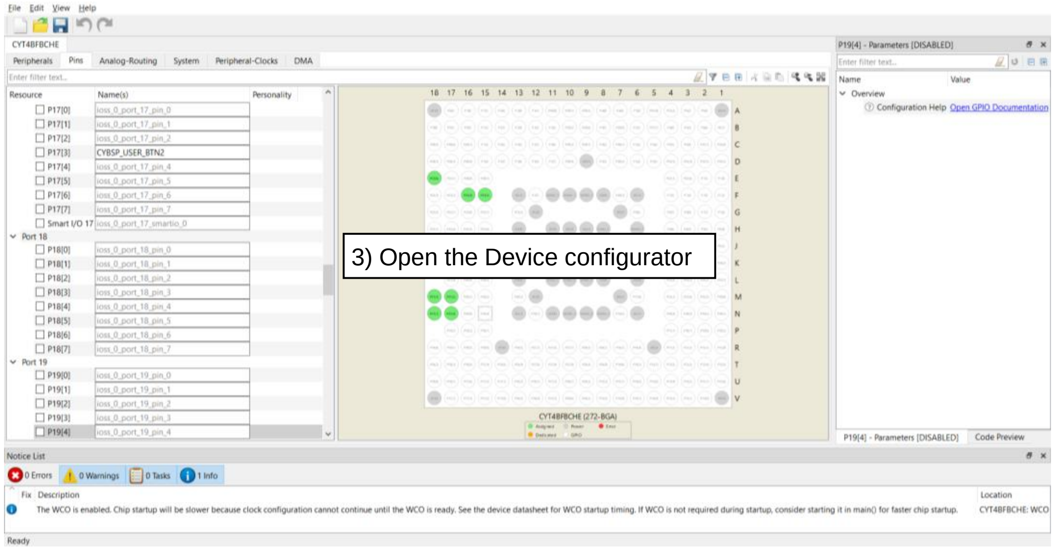
› Launch the Device Configurator

1) Select the “PortPin_training” project.
2) Click the Device Configurator in the Quick Panel
3) Then, open the Device configurator window


mtw_20221206_ce_test - PortPin_training/README.md -
File Edit Navigate Search Project Run Window Hel
Projec... Debug Regist... Periph...
mtb_shared
PortPin_training
Quick ... Variabl... Expres... Breakp...
Device Configurator 4.0
QSPI Configurator 4.0
Smart I/O Configurator 4.0
PortPin_training API Documentation

1) Select “SPortPin_training” project

2) Click the Device configurator


File Edit View Help
CYT4BFBCHE
Peripherals Pins Analog-Routing System Peripheral-Clocks DMA
Enter filter text...
CYT4BFBCHE (272-BGA)
P19[4] - Parameters [DISABLED]
Name Value
Overview
Configuration Help Open GPIO Documentation
Code Preview
Notice List
0 Errors 0 Warnings 0 Tasks 1 Info
Ready

Resource	Name(s)	Personality
□ P17[0]	ioss_0_port_17_pin_0	
□ P17[1]	ioss_0_port_17_pin_1	
□ P17[2]	ioss_0_port_17_pin_2	
□ P17[3]	CYBSP_USER_BTN2	
□ P17[4]	ioss_0_port_17_pin_4	
□ P17[5]	ioss_0_port_17_pin_5	
□ P17[6]	ioss_0_port_17_pin_6	
□ P17[7]	ioss_0_port_17_pin_7	
□ Smart I/O 17	ioss_0_port_17_smartio_0	
Port 18		
□ P18[0]	ioss_0_port_18_pin_0	
□ P18[1]	ioss_0_port_18_pin_1	
□ P18[2]	ioss_0_port_18_pin_2	
□ P18[3]	ioss_0_port_18_pin_3	
□ P18[4]	ioss_0_port_18_pin_4	
□ P18[5]	ioss_0_port_18_pin_5	
□ P18[6]	ioss_0_port_18_pin_6	
□ P18[7]	ioss_0_port_18_pin_7	
Port 19		
□ P19[0]	ioss_0_port_19_pin_0	
□ P19[1]	ioss_0_port_19_pin_1	
□ P19[2]	ioss_0_port_19_pin_2	
□ P19[3]	ioss_0_port_19_pin_3	
□ P19[4]	ioss_0_port_19_pin_4	

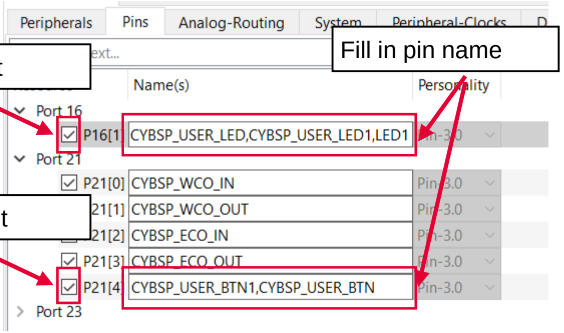
Fix	Description	Location
	The WCO is enabled. Chip startup will be slower because clock configuration cannot continue until the WCO is ready. See the device datasheet for WCO startup timing. If WCO is not required during startup, consider starting it in main() for faster chip startup.	CYT4BFBCHE: WCO

3) Open the Device configurator

Device configurator configuration (contd.)

› Configure GPIO

- Pin assignment is as follows:
  - P16[1] is used for “CYBSP_USER_LED”
  - P21[4] is used for “CYBSP_USER_BTN1”



Peripherals | Pins | Analog-Routing | System | Peripheral-Clocks | D

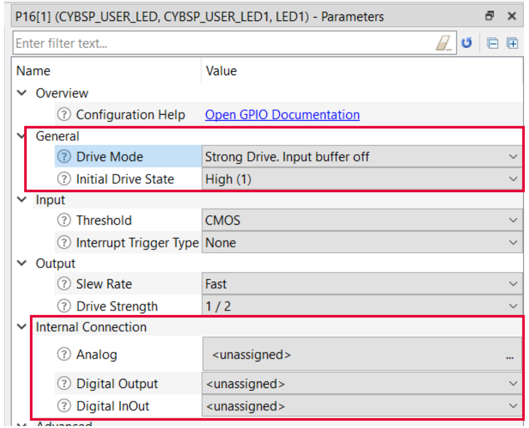
ext...

	Name(s)	Personality
Port 16		
☑ P16[1]	CYBSP_USER_LED,CYBSP_USER_LED1,LED1	Pin-3.0
Port 21		
☑ P21[0]	CYBSP_WCO_IN	Pin-3.0
☑ P21[1]	CYBSP_WCO_OUT	Pin-3.0
☑ P21[2]	CYBSP_ECO_IN	Pin-3.0
☑ P21[3]	CYBSP_ECO_OUT	Pin-3.0
☑ P21[4]	CYBSP_USER_BTN1,CYBSP_USER_BTN	Pin-3.0
Port 23		

Enable the P16.1 for LED output

Fill in pin name

Enable the P21.4 for button input



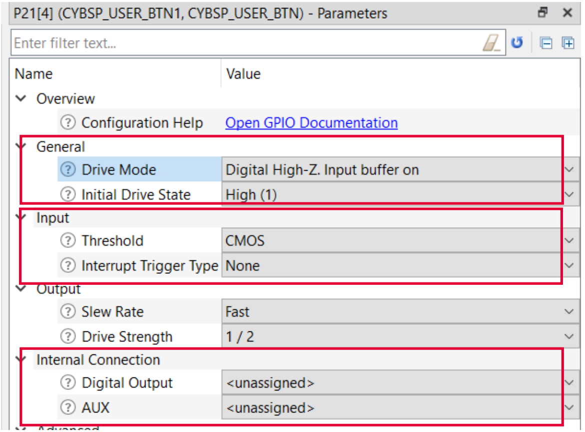
P16[1] (CYBSP_USER_LED, CYBSP_USER_LED1, LED1) - Parameters

Enter filter text...

Name	Value
Overview	
Configuration Help	Open GPIO Documentation
General	
Drive Mode	Strong Drive. Input buffer off
Initial Drive State	High (1)
Input	
Threshold	CMOS
Interrupt Trigger Type	None
Output	
Slew Rate	Fast
Drive Strength	1 / 2
Internal Connection	
Analog	<unassigned>
Digital Output	<unassigned>
Digital InOut	<unassigned>
Advanced	

Drive mode:
Strong Drive, input buffer off
Initial Drive state:
High

Internal connection:
GPIO (unassigned)



P21[4] (CYBSP_USER_BTN1, CYBSP_USER_BTN) - Parameters

Enter filter text...

Name	Value
Overview	
Configuration Help	Open GPIO Documentation
General	
Drive Mode	Digital High-Z. Input buffer on
Initial Drive State	High (1)
Input	
Threshold	CMOS
Interrupt Trigger Type	None
Output	
Slew Rate	Fast
Drive Strength	1 / 2
Internal Connection	
Digital Output	<unassigned>
AUX	<unassigned>
Advanced	

Drive mode:
Digital High-Z, input buffer on
Initial Drive state:
High

Threshold: CMOS
Interrupt Trigger Type:
Not use

Internal connection:
GPIO (unassigned)

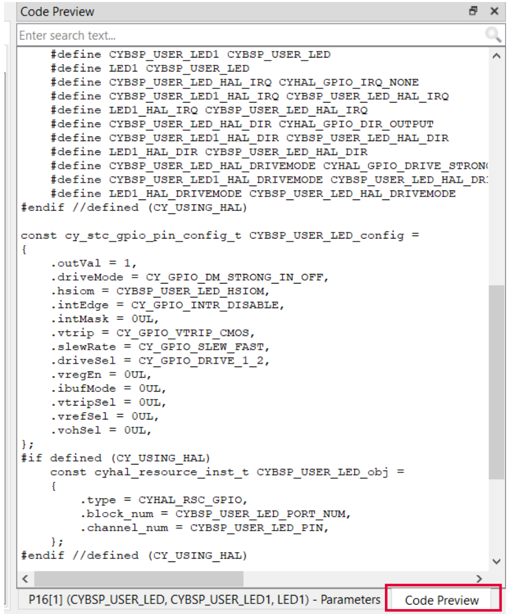
Note: To limit noise, when Drive Strength is configured to “Full”. Device configurator displays a warning.

Device configurator configuration (contd.)

› Confirm configuration result

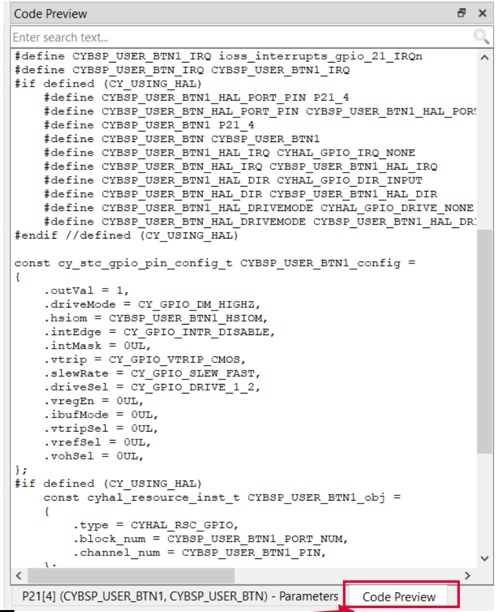
- You can check the configuration result in the “Code Preview” tab of the Device Configurator

P16.1: CYBSP_USER_LED



```
    #define CYBSP_USER_LED1 CYBSP_USER_LED
    #define LED1 CYBSP_USER_LED
    #define CYBSP_USER_LED_HAL_IRQ CYHAL_GPIO_IRQ_NONE
    #define CYBSP_USER_LED1_HAL_IRQ CYBSP_USER_LED_HAL_IRQ
    #define LED1_HAL_IRQ CYBSP_USER_LED_HAL_IRQ
    #define CYBSP_USER_LED_HAL_DIR CYHAL_GPIO_DIR_OUTPUT
    #define CYBSP_USER_LED1_HAL_DIR CYBSP_USER_LED_HAL_DIR
    #define LED1_HAL_DIR CYBSP_USER_LED_HAL_DIR
    #define CYBSP_USER_LED_HAL_DRIVEMODE CYHAL_GPIO_DRIVE_STRON
    #define CYBSP_USER_LED1_HAL_DRIVEMODE CYBSP_USER_LED_HAL_DR
    #define LED1_HAL_DRIVEMODE CYBSP_USER_LED_HAL_DRIVEMODE
#endif //defined (CY_USING_HAL)

const cy_stc_gpio_pin_config_t CYBSP_USER_LED_config =
{
    .outVal = 1,
    .driveMode = CY_GPIO_DM_STRONG_IN_OFF,
    .hsiom = CYBSP_USER_LED_HSIOM,
    .intEdge = CY_GPIO_INTR_DISABLE,
    .intMask = 0UL,
    .vtrip = CY_GPIO_VTRIP_CMOS,
    .slewRate = CY_GPIO_SLEW_FAST,
    .driveSel = CY_GPIO_DRIVE_1_2,
    .vregEn = 0UL,
    .ibufMode = 0UL,
    .vtripSel = 0UL,
    .vrefSel = 0UL,
    .vohSel = 0UL,
};
#if defined (CY_USING_HAL)
    const cyhal_resource_inst_t CYBSP_USER_LED_obj =
    {
        .type = CYHAL_RSC_GPIO,
        .block_num = CYBSP_USER_LED_PORT_NUM,
        .channel_num = CYBSP_USER_LED_PIN,
    };
#endif //defined (CY_USING_HAL)
```

P21.4: CYBSP_USER_BTN1



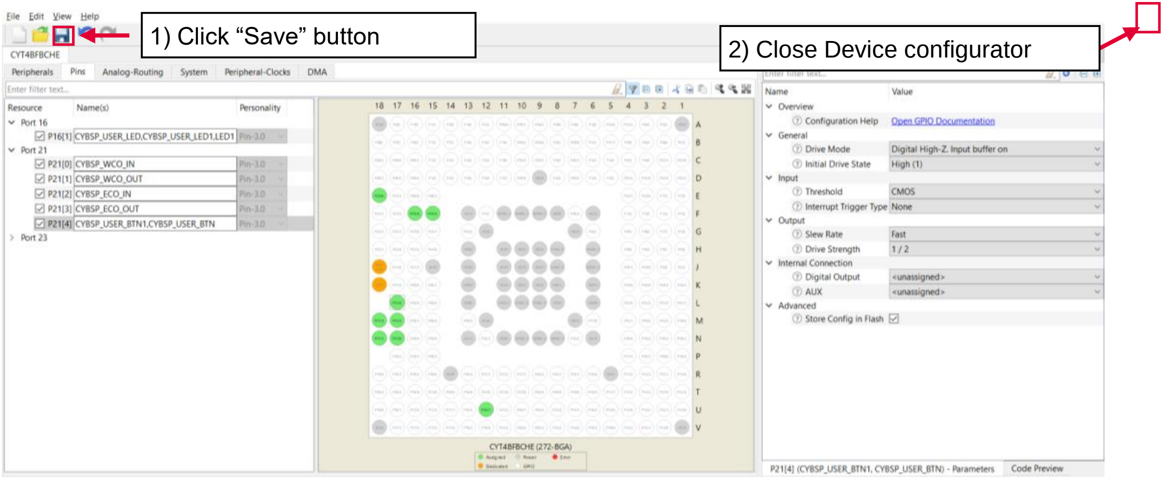
```
#define CYBSP_USER_BTN1_IRQ ioss_interrupts_gpio_21_IRQn
#define CYBSP_USER_BTN_IRQ CYBSP_USER_BTN1_IRQ
#if defined (CY_USING_HAL)
    #define CYBSP_USER_BTN1_HAL_PORT_PIN P21_4
    #define CYBSP_USER_BTN_HAL_PORT_PIN CYBSP_USER_BTN1_HAL_POR
    #define CYBSP_USER_BTN1 P21_4
    #define CYBSP_USER_BTN CYBSP_USER_BTN1
    #define CYBSP_USER_BTN1_HAL_IRQ CYHAL_GPIO_IRQ_NONE
    #define CYBSP_USER_BTN_HAL_IRQ CYBSP_USER_BTN1_HAL_IRQ
    #define CYBSP_USER_BTN1_HAL_DIR CYHAL_GPIO_DIR_INPUT
    #define CYBSP_USER_BTN_HAL_DIR CYBSP_USER_BTN1_HAL_DIR
    #define CYBSP_USER_BTN1_HAL_DRIVEMODE CYHAL_GPIO_DRIVE_NONE
    #define CYBSP_USER_BTN_HAL_DRIVEMODE CYBSP_USER_BTN1_HAL_DR
#endif //defined (CY_USING_HAL)

const cy_stc_gpio_pin_config_t CYBSP_USER_BTN1_config =
{
    .outVal = 1,
    .driveMode = CY_GPIO_DM_HIGHZ,
    .hsiom = CYBSP_USER_BTN1_HSIOM,
    .intEdge = CY_GPIO_INTR_DISABLE,
    .intMask = 0UL,
    .vtrip = CY_GPIO_VTRIP_CMOS,
    .slewRate = CY_GPIO_SLEW_FAST,
    .driveSel = CY_GPIO_DRIVE_1_2,
    .vregEn = 0UL,
    .ibufMode = 0UL,
    .vtripSel = 0UL,
    .vrefSel = 0UL,
    .vohSel = 0UL,
};
#if defined (CY_USING_HAL)
    const cyhal_resource_inst_t CYBSP_USER_BTN1_obj =
    {
        .type = CYHAL_RSC_GPIO,
        .block_num = CYBSP_USER_BTN1_PORT_NUM,
        .channel_num = CYBSP_USER_BTN1_PIN,
    };
```

Code preview tab

Device configurator configuration (contd.)

› Close Device configurator

- Click the **Save** button after completing all settings; then close the Device configurator



1) Click “Save” button

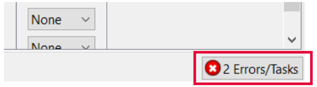
2) Close Device configurator

Resource	Name(s)	Personality
⌄ Port 16		
☑ P16[1]	CYBSP_USER_LED,CYBSP_USER_LED1,LED1	Pin-3.0
⌄ Port 21		
☑ P21[0]	CYBSP_WCO_IN	Pin-3.0
☑ P21[1]	CYBSP_WCO_OUT	Pin-3.0
☑ P21[2]	CYBSP_ECO_IN	Pin-3.0
☑ P21[3]	CYBSP_ECO_OUT	Pin-3.0
☑ P21[4]	CYBSP_USER_BTN1,CYBSP_USER_BTN	Pin-3.0
› Port 23		

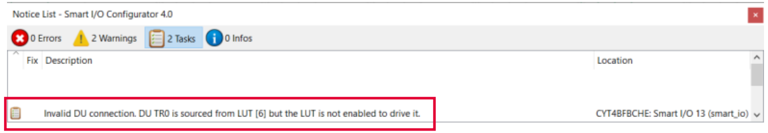
P21[4] (CYBSP_USER_BTN1, CYBSP_USER_BTN) - Parameters

Name	Value
⌄ Overview	
Configuration Help	Open GPIO Documentation
⌄ General	
Drive Mode	Digital High-Z. Input buffer on
Initial Drive State	High (1)
⌄ Input	
Threshold	CMOS
Interrupt Trigger Type	None
⌄ Output	
Slew Rate	Fast
Drive Strength	1 / 2
⌄ Internal Connection	
Digital Output	<unassigned>
AUX	<unassigned>
⌄ Advanced	
Store Config in Flash	☑

- If an **Errors/Tasks** message appears, it should be resolved according to the instructions



Click



Notice List - Smart I/O Configurator 4.0

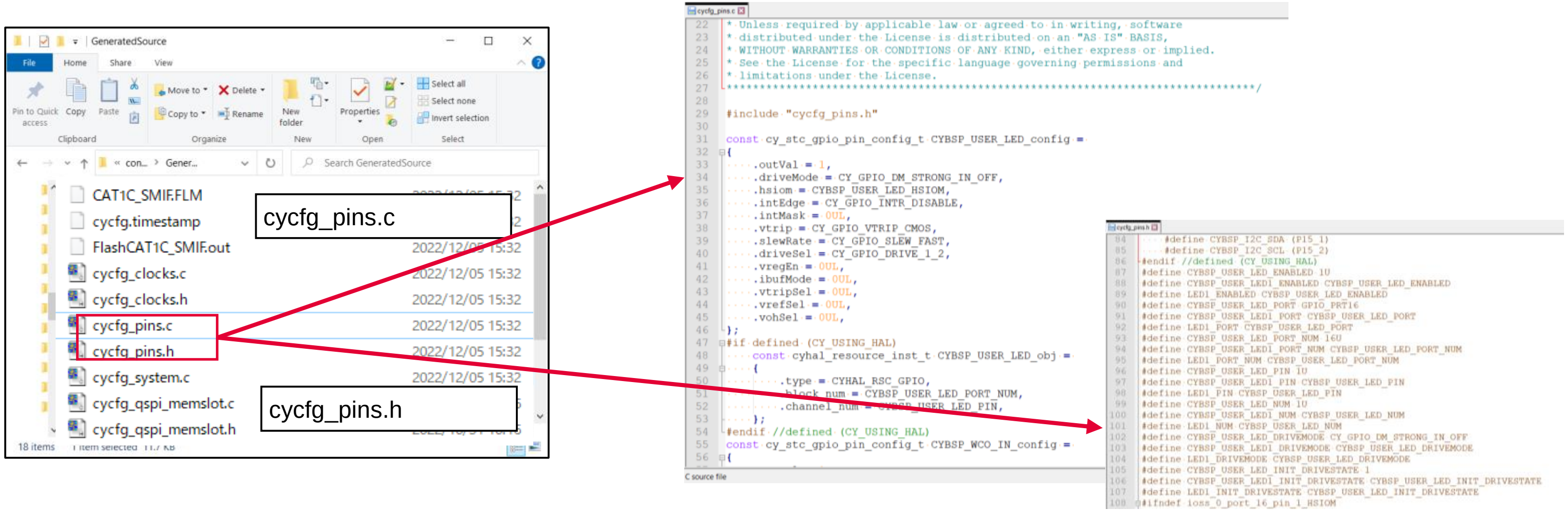
0 Errors 2 Warnings 2 Tasks 0 Infos

Fix	Description	Location
	Invalid DU connection. DU TR0 is sourced from LUT [6] but the LUT is not enabled to drive it.	CYT4BFBCHE: Smart I/O 13 (smart_io)

Device configurator configuration (contd.)

› Configuration file

- The Pins Configurator generates code into a "GeneratedSource" directory in your Eclipse IDE application, or in the same location you saved the *.modus file for non-IDE applications.
- In this example, the code is as follows:



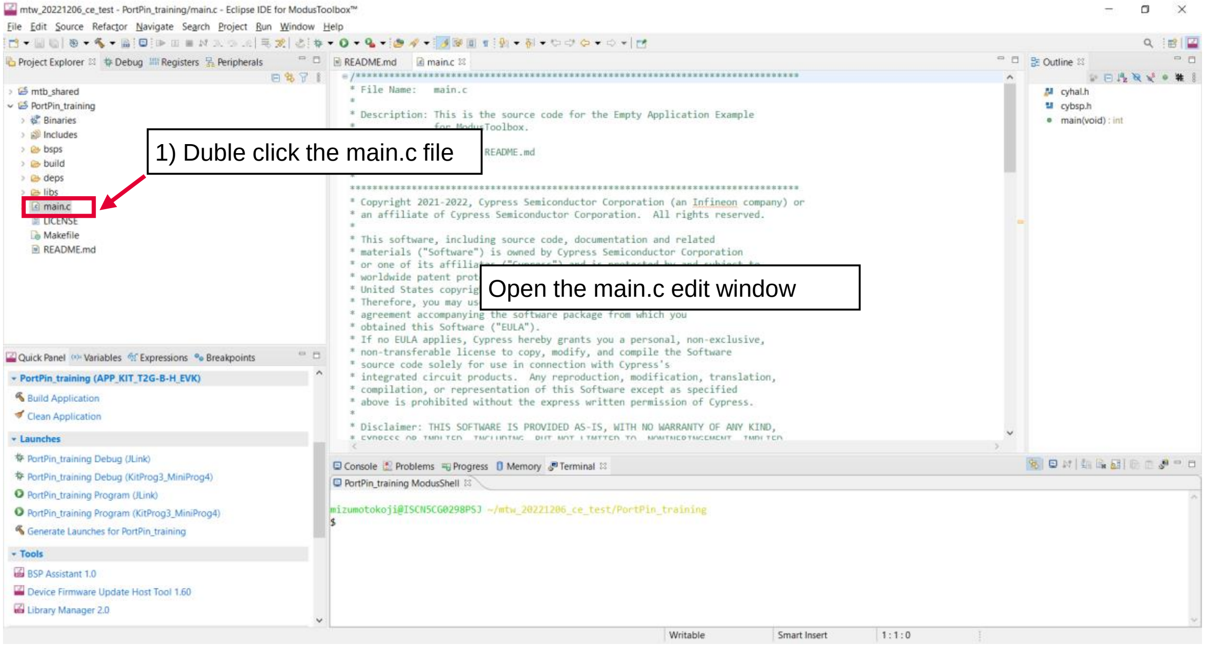
CAT1C_SMIF.FLM
cycfg.timestamp
FlashCAT1C_SMIF.out 2022/12/05 15:32
cycfg_clocks.c 2022/12/05 15:32
cycfg_clocks.h 2022/12/05 15:32
cycfg_pins.c 2022/12/05 15:32
cycfg_pins.h 2022/12/05 15:32
cycfg_system.c 2022/12/05 15:32
cycfg_qspi_memslot.c
cycfg_qspi_memslot.h

```c
* Unless required by applicable law or agreed to in writing, software
* distributed under the License is distributed on an "AS IS" BASIS,
* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
* See the License for the specific language governing permissions and
* limitations under the License.
********************************************************************************/

#include "cycfg_pins.h"

const cy_stc_gpio_pin_config_t CYBSP_USER_LED_config =
{
    .outVal = 1,
    .driveMode = CY_GPIO_DM_STRONG_IN_OFF,
    .hsiom = CYBSP_USER_LED_HSIOM,
    .intEdge = CY_GPIO_INTR_DISABLE,
    .intMask = 0UL,
    .vtrip = CY_GPIO_VTRIP_CMOS,
    .slewRate = CY_GPIO_SLEW_FAST,
    .driveSel = CY_GPIO_DRIVE_1_2,
    .vregEn = 0UL,
    .ibufMode = 0UL,
    .vtripSel = 0UL,
    .vrefSel = 0UL,
    .vohSel = 0UL,
};
#if defined (CY_USING_HAL)
    const cyhal_resource_inst_t CYBSP_USER_LED_obj =
    {
        .type = CYHAL_RSC_GPIO,
        .block_num = CYBSP_USER_LED_PORT_NUM,
        .channel_num = CYBSP_USER_LED_PIN,
    };
#endif //defined (CY_USING_HAL)
const cy_stc_gpio_pin_config_t CYBSP_WCO_IN_config =
{
```

```c
    #define CYBSP_I2C_SDA (P15_1)
    #define CYBSP_I2C_SCL (P15_2)
#endif //defined (CY_USING_HAL)
#define CYBSP_USER_LED_ENABLED 1U
#define CYBSP_USER_LED1_ENABLED CYBSP_USER_LED_ENABLED
#define LED1_ENABLED CYBSP_USER_LED_ENABLED
#define CYBSP_USER_LED_PORT GPIO_PRT16
#define CYBSP_USER_LED1_PORT CYBSP_USER_LED_PORT
#define LED1_PORT CYBSP_USER_LED_PORT
#define CYBSP_USER_LED_PORT_NUM 16U
#define CYBSP_USER_LED1_PORT_NUM CYBSP_USER_LED_PORT_NUM
#define LED1_PORT_NUM CYBSP_USER_LED_PORT_NUM
#define CYBSP_USER_LED_PIN 1U
#define CYBSP_USER_LED1_PIN CYBSP_USER_LED_PIN
#define LED1_PIN CYBSP_USER_LED_PIN
#define CYBSP_USER_LED_NUM 1U
#define CYBSP_USER_LED1_NUM CYBSP_USER_LED_NUM
#define LED1_NUM CYBSP_USER_LED_NUM
#define CYBSP_USER_LED_DRIVEMODE CY_GPIO_DM_STRONG_IN_OFF
#define CYBSP_USER_LED1_DRIVEMODE CYBSP_USER_LED_DRIVEMODE
#define LED1_DRIVEMODE CYBSP_USER_LED_DRIVEMODE
#define CYBSP_USER_LED_INIT_DRIVESTATE 1
#define CYBSP_USER_LED1_INIT_DRIVESTATE CYBSP_USER_LED_INIT_DRIVESTATE
#define LED1_INIT_DRIVESTATE CYBSP_USER_LED_INIT_DRIVESTATE
#ifndef ioss_0_port_16_pin_1_HSIOM
```

Implementation

- This section describes how to implement the configured pin setting. This example will implement the pin setting configuration in the PortPin_training project.
  - Open main.c in PortPin_training project


mtw_20221206_ce_test - PortPin_training/main.c - Eclipse IDE for ModusToolbox™
File Edit Source Refactor Navigate Search Project Run Window Help
Project Explorer
Debug
Registers
Peripherals
mtb_shared
PortPin_training
Binaries
Includes
bsps
build
deps
libs
main.c
LICENSE
Makefile
README.md
README.md
main.c
Outline
cyhal.h
cybsp.h
main(void) : int
Quick Panel
Variables
Expressions
Breakpoints
PortPin_training (APP_KIT_T2G-B-H_EVK)
Build Application
Clean Application
Launches
PortPin_training Debug (JLink)
PortPin_training Debug (KitProg3_MiniProg4)
PortPin_training Program (JLink)
PortPin_training Program (KitProg3_MiniProg4)
Generate Launches for PortPin_training
Tools
BSP Assistant 1.0
Device Firmware Update Host Tool 1.60
Library Manager 2.0
Console
Problems
Progress
Memory
Terminal
PortPin_training ModusShell
Writable
Smart Insert
1 : 1 : 0

1) Duble click the main.c file

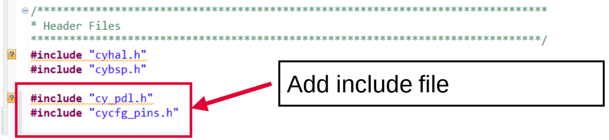
Open the main.c edit window

```
/*******************************************************************************
* File Name:   main.c
*
* Description: This is the source code for the Empty Application Example
*              for ModusToolbox.
*
                 README.md
*
*******************************************************************************
* Copyright 2021-2022, Cypress Semiconductor Corporation (an Infineon company) or
* an affiliate of Cypress Semiconductor Corporation.  All rights reserved.
*
* This software, including source code, documentation and related
* materials ("Software") is owned by Cypress Semiconductor Corporation
* or one of its affilia
* worldwide patent prot
* United States copyrig
* Therefore, you may us
* agreement accompanying the software package from which you
* obtained this Software ("EULA").
* If no EULA applies, Cypress hereby grants you a personal, non-exclusive,
* non-transferable license to copy, modify, and compile the Software
* source code solely for use in connection with Cypress's
* integrated circuit products.  Any reproduction, modification, translation,
* compilation, or representation of this Software except as specified
* above is prohibited without the express written permission of Cypress.
*
* Disclaimer: THIS SOFTWARE IS PROVIDED AS-IS, WITH NO WARRANTY OF ANY KIND,
```

```
mizumotokoji@ISCN5CG0298PSJ ~/mtw_20221206_ce_test/PortPin_training
$
```

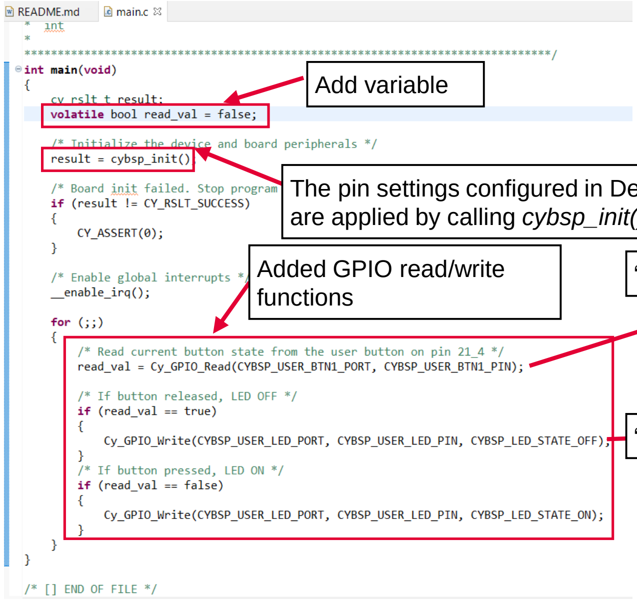
Implementation (contd.)

› Code modification



```c
/*******************************************************************************
 * Header Files
 *******************************************************************************/
#include "cyhal.h"
#include "cybsp.h"

#include "cy_pdl.h"
#include "cycfg_pins.h"
```

Add include file



```c
 *   int
 *
 *******************************************************************************/
int main(void)
{
    cy_rslt_t result;
    volatile bool read_val = false;

    /* Initialize the device and board peripherals */
    result = cybsp_init();

    /* Board init failed. Stop program
    if (result != CY_RSLT_SUCCESS)
    {
        CY_ASSERT(0);
    }

    /* Enable global interrupts */
    __enable_irq();

    for (;;)
    {
        /* Read current button state from the user button on pin 21_4 */
        read_val = Cy_GPIO_Read(CYBSP_USER_BTN1_PORT, CYBSP_USER_BTN1_PIN);

        /* If button released, LED OFF */
        if (read_val == true)
        {
            Cy_GPIO_Write(CYBSP_USER_LED_PORT, CYBSP_USER_LED_PIN, CYBSP_LED_STATE_OFF);
        }
        /* If button pressed, LED ON */
        if (read_val == false)
        {
            Cy_GPIO_Write(CYBSP_USER_LED_PORT, CYBSP_USER_LED_PIN, CYBSP_LED_STATE_ON);
        }
    }
}

/* [] END OF FILE */
```

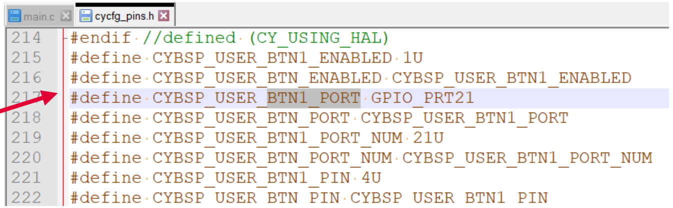
Add variable

The pin settings configured in Device Configurator are applied by calling *cybsp_init()* function

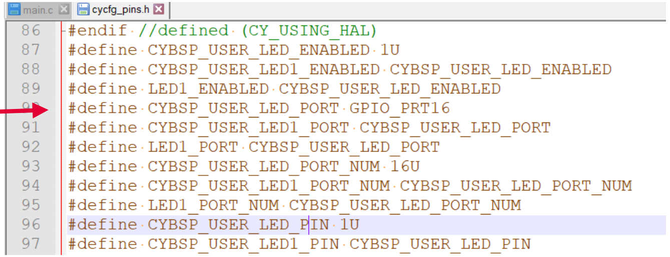
Added GPIO read/write functions

“CYBSP_USER_BTN1” signal

“CYBSP_USER_LED” signal



```c
#endif //defined (CY_USING_HAL)
#define CYBSP_USER_BTN1_ENABLED 1U
#define CYBSP_USER_BTN_ENABLED CYBSP_USER_BTN1_ENABLED
#define CYBSP_USER_BTN1_PORT GPIO_PRT21
#define CYBSP_USER_BTN_PORT CYBSP_USER_BTN1_PORT
#define CYBSP_USER_BTN1_PORT_NUM 21U
#define CYBSP_USER_BTN_PORT_NUM CYBSP_USER_BTN1_PORT_NUM
#define CYBSP_USER_BTN1_PIN 4U
#define CYBSP_USER_BTN_PIN CYBSP_USER_BTN1_PIN
```



```c
#endif //defined (CY_USING_HAL)
#define CYBSP_USER_LED_ENABLED 1U
#define CYBSP_USER_LED1_ENABLED CYBSP_USER_LED_ENABLED
#define LED1_ENABLED CYBSP_USER_LED_ENABLED
#define CYBSP_USER_LED_PORT GPIO_PRT16
#define CYBSP_USER_LED1_PORT CYBSP_USER_LED_PORT
#define LED1_PORT CYBSP_USER_LED_PORT
#define CYBSP_USER_LED_PORT_NUM 16U
#define CYBSP_USER_LED1_PORT_NUM CYBSP_USER_LED_PORT_NUM
#define LED1_PORT_NUM CYBSP_USER_LED_PORT_NUM
#define CYBSP_USER_LED_PIN 1U
#define CYBSP_USER_LED1_PIN CYBSP_USER_LED_PIN
```

Implementation (contd.)

Pin configuration

- Call the ***Cybsp_init()*** function to configure pins
  - Initialize all hardware on the board
  - Pin settings that are configured in the Device Configurator are set in this function

GPIO port read

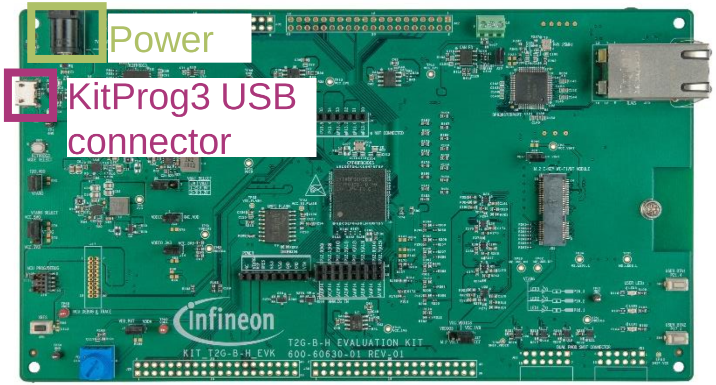
- Call the ***Cy_GPIO_Read()*** function to read GPIO
  - It is used to read the user button state
  - "CYBSP_USER_BTN1" is configured as "***CYBSP_USER_BTN1_PORT***(= Port 21)" and "***CYBSP_USER_BTN1_PIN*** (= 4 pin)" in ***cycfg_pins.h*** file

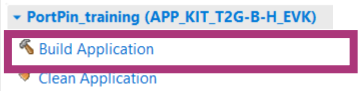
GPIO port write

- Call the ***Cy_GPIO_Write()*** function to set GPIO
  - It is used to control the user LED
  - "CYBSP_USER_LED" is configured as "***GPIO_USER_LED_PORT*** (= Port 16)" and "***GPIO_RST_LED_PIN*** (= 1 pin)" in ***cycfg_pins.h*** file

Compiling and programming

1. Connect to power and USB cable
2. Use Eclipse IDE for ModusToolbox™ software for compiling and programming
3. Compile
   a) Select the target application project in the Project Explorer
   b) In the Quick Panel, scroll down, and click "Build Application" in PortPin_training (APP KIT_T2G-B-H_EVK)



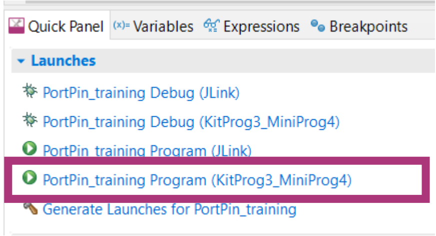


PortPin_training (APP_KIT_T2G-B-H_EVK)

Build Application

Clean Application

4. Programming
   a) Select the target application project in the Project Explorer
   b) In the Quick Panel, scroll down, and click "PortPin_training Program (KitProg3_MiniProg4)" in Launches



Quick Panel | Variables | Expressions | Breakpoints

Launches

PortPin_training Debug (JLink)

PortPin_training Debug (KitProg3_MiniProg4)

PortPin_training Program (JLink)

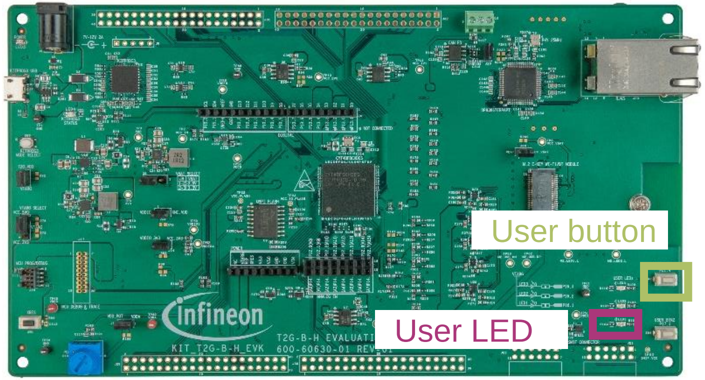
PortPin_training Program (KitProg3_MiniProg4)

Generate Launches for PortPin_training

Run and test

1. After successful programming, press the user button (P21.4), and observe that the user LED (P16.1) turns ON demonstrating the GPIO read and write function.

2. Release the user button, observe that the user LED turns OFF



References

Datasheet

- CYT4BF datasheet 32-bit Arm® Cortex®-M7 microcontroller TRAVEO™ T2G family

Architecture Technical reference manual

- TRAVEO™ T2G automotive body controller high family architecture technical reference manual

Registers Technical reference manual

- TRAVEO™ T2G Automotive body controller high registers technical reference manual

PDL/HAL

- PDL
- HAL

Training

- TRAVEO™ T2G Training

Revision History

Revision	ECN	Submission Date	Description of Change
**	7846970	2022/12/12	Initial release

Important notice and warnings

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

Edition 2022-12
Published by
Infineon Technologies AG
81726 Munich, Germany

Do you have a question about this document?
Go to:
www.infineon.com/support

Document reference
002-36711 Rev. **

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.