Customer training workshop: Device Configurator_Pins

TRAVEO™ 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™ T2G CYT4BFBCCH device is used in this code example.
› Board
– The TRAVEO™ 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
Launch the Device Configurator (contd.)

› **From Device Configurator**

1) Open the Device configurator

2) On the Pins tab, check each port resource
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

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
Device Configurator pin configuration (contd.)

- Device Configurator pin configuration

Code preview automatically generates the selected parameter configuration.
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)

- 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

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
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
Configure GPIO

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

Enable the P16.1 for LED output
Enable the P21.4 for button input

Drive mode:
- P16: Strong Drive, input buffer off
- P21: Digital High-Z, input buffer on

Initial Drive state:
- P16: High
- P21: High

Internal connection:
- GPIO (unassigned)

Threshold: CMOS
Interrupt Trigger Type: Not use

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

P21.4: CYBSP_USER_BTN1
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

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

Click
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:
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
Implementation (contd.)

Code modification

- Add include file
- Add variable

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

```
int main(void)
{
    cyrst_t result;
    volatile bool read_val = false;
    result = cybsp_init();
    // Initialize the device and board peripherals
    if (result != CYBSP_SUCCESS)
    {
        return -1;
    }
    // Enable global interrupts
    for (; ;)
    {
        // Read current button state from the user button on pin PL4
        read_val = cy_GPIO_Read(CYBSP_USER_BT1_PORT, CYBSP_USER_BT1_PIN);
        // If button released, LED OFF
        if (read_val == true)
        {
            cy_GPIO_Write(port, value, CYBSP_USER_BT1_PORT, CYBSP_USER_BT1_PIN, CYBSP_USER_BT1_PIN, CYBSP_USER_BT1_PIN);
            // If button pressed, LED ON
        }
    }
}
```

*CYBSP_USER_BT1* signal
*CYBSP_USER_LED* signal

Added GPIO read/write functions
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_USERBTN1_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)
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
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

<table>
<thead>
<tr>
<th>Revision</th>
<th>ECN</th>
<th>Submission Date</th>
<th>Description of Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>**</td>
<td>7846970</td>
<td>2022/12/12</td>
<td>Initial release</td>
</tr>
</tbody>
</table>
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

© 2022 Infineon Technologies AG.
All Rights Reserved.

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.