

# Readme

**APM32F445\_446 SDK**

**Rev: V1.0**

# 1 Introduction

The Geehy Semiconductor APM32F445\_446 EVAL board software development kit includes a series driver library, a group of example applications that demonstrate key peripheral functionality, and other development files.

Software development kit have a hierarchy as follows:

- SDK directory
  - \* [Boards](#)
  - \* [Documents](#)
  - \* [Examples](#)
  - \* [Libraries](#)
  - \* [Middlewares](#)
  - \* [Package](#)

## Table of Contents

<b>1</b>	<b>Introduction .....</b>	<b>1</b>
<b>2</b>	<b>About Boards.....</b>	<b>5</b>
<b>3</b>	<b>About Documents .....</b>	<b>6</b>
<b>4</b>	<b>About Examples .....</b>	<b>7</b>
4.1	ADC_HardwareCompare .....	11
4.2	ADC_HardwareConversion .....	11
4.3	ADC_ReadConfig.....	11
4.4	ADC_SoftwareConversion.....	12
4.5	CAN_ClassicFrames .....	12
4.6	CAN_FdFrames .....	12
4.7	CAN_PretendedNetworking .....	13
4.8	CAN_RxFifo .....	13
4.9	CLOCK_AllClocksArrayConfiguration.....	13
4.10	CLOCK_AllClocksConfiguration .....	14
4.11	CLOCK_SeparateConfig.....	14
4.12	COMP_DAC.....	15
4.13	CRC_Checksum .....	15
4.14	CSEC_Security .....	15
4.15	EINJ_Injection .....	16
4.16	EREP_Report.....	16
4.17	EWDT_Interrupt .....	17
4.18	FLASH_EraseSector .....	17
4.19	FLASH_Partition.....	18
4.20	FLASH_ProgramOnce.....	18
4.21	FLASH_ProgramSection .....	18
4.22	FLASH_Protection .....	19

4.23	CFGIO_I2C_Master .....	19
4.24	CFGIO_SPI_Dual.....	20
4.25	CFGIO_SPI_Irq .....	20
4.26	CFGIO_UART_Echo .....	21
4.27	CFGTMR_InputCapture.....	22
4.28	CFGTMR_PeriodicInterrupt .....	22
4.29	CFGTMR_PWMOutput .....	23
4.30	CFGTMR0_CenterAlignedPWMOutput .....	23
4.31	CFGTMR0_CombinedPWMOutput.....	24
4.32	CFGTMR0_DeadtimeInsertPWMOutput .....	24
4.33	CFGTMR0_EdgeAlignedPWMOutput.....	25
4.34	CFGTMR0_FaultControl .....	26
4.35	CFGTMR1_OCMatchSet.....	26
4.36	CFGTMR1_OCToggle .....	26
4.37	CFGTMR2_QDPhaseEncoding.....	27
4.38	LPI2C_TwoBoardsDMA_Master.....	28
4.39	LPI2C_TwoBoardsDMA_Slave.....	28
4.40	LPI2C_TwoBoardsInterrupt_Master .....	29
4.41	LPI2C_TwoBoardsInterrupt_Slave .....	30
4.42	LPITMR_Counter.....	30
4.43	LPITMR_Timer .....	30
4.44	LPSPI_DMA .....	31
4.45	LPSPI_Isr .....	31
4.46	LPTMR_PeriodicInterrupt .....	32
4.47	LPTMR_PulseCounter .....	32
4.48	LIN_Master .....	33
4.49	LIN_Slave .....	34
4.50	LPUART_DMA .....	34

4.51	LPUART_SendReceive .....	35
4.52	MPU_MemoryProtection .....	35
4.53	PDU_PeriodicInterrupt .....	36
4.54	PINS_Isr .....	36
4.55	PINS_Led.....	36
4.56	PINS_Read .....	37
4.57	Power_SwitchMode .....	37
4.58	RTC_Alarm .....	38
4.59	RTC_ClockOut .....	38
4.60	RTC_Second .....	38
4.61	FreeRTOS .....	39
4.62	RT-Thread .....	39
4.63	Template .....	39
4.64	WDT_FastTest .....	39
4.65	WDT_Interrupt .....	40
<b>5</b>	<b>About Libraries.....</b>	<b>42</b>
<b>6</b>	<b>About Middlewares .....</b>	<b>43</b>
<b>7</b>	<b>About Package .....</b>	<b>44</b>
<b>8</b>	<b>Revision History .....</b>	<b>45</b>

## 2 About Boards

The boards folder includes a board support package for APM32F445\_446 EVAL board. It can help drive the peripheral circuit or components on the board quickly. The BSP can be found in the [~/Boards](#) directory.

The BSP provided are built for APM32F445\_446 EVAL board compatibility. For other user development board use, some minor modifications may be required.

Boards have a hierarchy as follows:

- Boards folder
  - \* Board\_APM32F445\_EVAL folder
    - inc
    - src
  - \* Board\_APM32F446\_EVAL folder
    - inc
    - src
  - \* board.c
  - \* board.h

### 3 **About Documents**

The documents folder includes a link file that can be redirected to the technical support center of Geehy semiconductor. The BSP can be found in the [~/Documents](#) directory.

## 4 About Examples

The example applications can be found in the [~/Examples](#) directory.

The examples provided are built for APM32F445\_446 EVAL board compatibility. For other user development board use, some minor modifications may be required.

Example projects have a hierarchy as follows:

- Example folder
  - BOARD\_APM32F445\_EVAL folder
    - Include
    - Project
      - IAR
      - MDK
      - Eclipse
    - Source
  - BOARD\_APM32F446\_EVAL folder
    - Include
    - Project
      - IAR
      - MDK
      - Eclipse
    - Source

All example applications tested with: **APM32F445\_446 StdPeriphDriver v1.0.1**, include the following examples for APM32F445, and APM32F446 as so:

- Examples
  - \* ADC
    - [ADC\\_HardwareCompare](#)
    - [ADC\\_HardwareConversion](#)
    - [ADC\\_ReadConfig](#)
    - [ADC\\_SoftwareConversion](#)



- \* CAN
  - [CAN\\_ClassicFrames](#)
  - [CAN\\_FdFrames](#)
  - [CAN\\_PretendedNetworking](#)
  - [CAN\\_RxFifo](#)
- \* CFGIO
  - [CFGIO\\_I2C\\_Master](#)
  - [CFGIO\\_SPI\\_Dual](#)
  - [CFGIO\\_SPI\\_Irq](#)
  - [CFGIO\\_UART\\_Echo](#)
- \* CLOCK
  - [CLOCK\\_AllClocksArrayConfiguration](#)
  - [CLOCK\\_AllClocksConfiguration](#)
  - [CLOCK\\_SeparateConfig](#)
- \* COMP
  - [COMP\\_DAC](#)
- \* CRC
  - [CRC\\_Checksum](#)
- \* CSEC
  - [CSEC\\_Security](#)
- \* EINJ
  - [EINJ\\_Injection](#)
- \* EREP
  - [EREP\\_Report](#)
- \* EWDT
  - [EWDT\\_Interrupt](#)
- \* FLASH
  - [FLASH\\_EraseSector](#)

- [FLASH\\_Partition](#)
- [FLASH\\_ProgramOnce](#)
- [FLASH\\_ProgramSection](#)
- [FLASH\\_Protection](#)
- \* CFGTMR
  - [CFGTMR\\_InputCapture](#)
  - [CFGTMR\\_PeriodicInterrupt](#)
  - [CFGTMR\\_PWMOutput](#)
  - [CFGTMR0\\_CenterAlignedPWMOutput](#)
  - [CFGTMR0\\_CombinedPWMOutput](#)
  - [CFGTMR0\\_DeadtimeInsertPWMOutput](#)
  - [CFGTMR0\\_EdgeAlignedPWMOutput](#)
  - [CFGTMR0\\_FaultControl](#)
  - [CFGTMR1\\_OCMatchSet](#)
  - [CFGTMR1\\_OCToggle](#)
  - [CFGTMR2\\_QDPhaseEncoding](#)
- \* LPI2C
  - [LPI2C\\_TwoBoardsDMA\\_Master](#)
  - [LPI2C\\_TwoBoardsDMA\\_Slave](#)
  - [LPI2C\\_TwoBoardsInterrupt\\_Master](#)
  - [LPI2C\\_TwoBoardsInterrupt\\_Slave](#)
- \* LPITMR
  - [LPITMR\\_Counter](#)
  - [LPITMR\\_Timer](#)
- \* LPSPI
  - [LPSPI\\_DMA](#)
  - [LPSPI\\_Isr](#)
- \* LPTMR

- [LPTMR\\_PeriodicInterrupt](#)
- [LPTMR\\_PulseCounter](#)
- \* LPUART\_LIN
  - [LIN\\_Master](#)
  - [LIN\\_Slave](#)
  - [LPUART\\_DMA](#)
  - [LPUART\\_SendReceive](#)
- \* MPU
  - [MPU\\_MemoryProtection](#)
- \* PDU
  - [PDU\\_PeriodicInterrupt](#)
- \* PINS
  - [PINS\\_Isr](#)
  - [PINS\\_Led](#)
  - [PINS\\_Read](#)
- \* POWER
  - [Power\\_SwitchMode](#)
- \* RTC
  - [RTC\\_Alarm](#)
  - [RTC\\_ClockOut](#)
  - [RTC\\_Second](#)
- \* RTOS
  - [FreeRTOS](#)
  - [RT-Thread](#)
- \* Template
  - [Template](#)
- \* WDT
  - [WDT\\_FastTest](#)

---

- [WDT Interrupt](#)

## 4.1 **ADC\_HardwareCompare**

### 4.1.1 **Example Description**

After enabling the hardware comparison function, use a sliding rheostat to adjust the conversion voltage. If the limited voltage is exceeded, the LED light will turn red, and vice versa, it will turn green. The serial port outputs the conversion voltage value and flag bit status.

### 4.1.2 **Directory contents**

This example can be found in the [~/Examples/BOARD\\_APM32F445\\_EVAL/ADC/ADC\\_HardwareCompare](#) directory.

## 4.2 **ADC\_HardwareConversion**

### 4.2.1 **Example Description**

By adjusting the potentiometer on the development board, and the voltage value converted by the ADC channel can be printed in real-time through the serial port.

### 4.2.2 **Directory contents**

This example can be found in the [~/Examples/BOARD\\_APM32F445\\_EVAL/ADC/ADC\\_HardwareConversion](#) directory.

## 4.3 **ADC\_ReadConfig**

### 4.3.1 **Example Description**

This process is used to read various configuration information of the ADC module, which is defined by the user.

### 4.3.2 **Directory contents**

This example can be found in the [~/Examples/BOARD\\_APM32F445\\_EVAL/ADC/ADC\\_ReadConfig](#) directory.

## 4.4 **ADC\_SoftwareConversion**

### 4.4.1 **Example Description**

By adjusting the potentiometer on the development board, and the voltage value converted by the ADC channel can be printed in real-time through the serial port.

### 4.4.2 **Directory contents**

This example can be found in the [~ /Examples/BOARD\\_APM32F445\\_EVAL/ADC/ADC\\_SoftwareConversion](#) directory.

## 4.5 **CAN\_ClassicFrames**

### 4.5.1 **Example Description**

This example shows the usage of the CAN module with classic CAN frames. Using the two buttons it will send data to the CAN bus with standard CAN ID 1, or extended CAN ID 2. Two message buffers are configured to receive CAN frames with standard CAN ID 1 and extended CAN ID 1. It will toggle the green LED in case CAN frames are received.

### 4.5.2 **Directory contents**

This example can be found in the [~ /Examples/BOARD\\_APM32F445\\_EVAL/CAN/CAN\\_ClassicFrames](#) directory.

## 4.6 **CAN\_FdFrames**

### 4.6.1 **Example Description**

This example shows the usage of the CAN module with CAN FD frames. Using the two buttons it will send data to the CAN bus with standard CAN ID 1, or extended CAN ID 2. One message buffer is configured to receive CAN frames with standard CAN ID 1. It will toggle the green LED in case CAN frames are received.

### 4.6.2 **Directory contents**

This example can be found in the [~ /Examples/BOARD\\_APM32F445\\_EVAL/CAN/CAN\\_FdFrames](#) directory.

## 4.7 CAN\_PretendedNetworking

### 4.7.1 Example Description

This example shows the usage of the CAN Pretended Networking functionality. It enables Pretended Networking and then go to sleep mode. Once a matching CAN frame is received the device will be woke up, and two seconds later it will go to sleep mode again.

Blue LED On: The device is in SLEEP mode.

Green LED On: The device is in RUN mode.

The device will be woke up when the following CAN frame is received:

- Standard CAN ID = 1
- Payload = 0x11, 0x22, 0x33, 0x44, 0x55, 0x66, 0x77, 0x88.

### 4.7.2 Directory contents

This example can be found in the [~ /Examples/BOARD\\_APM32F445\\_EVAL/CAN/CAN\\_PretendedNetworking](#) directory.

## 4.8 CAN\_RxFifo

### 4.8.1 Example Description

This example shows the usage of the CAN Rx FIFO.

The Rx FIFO is configured with 8 ID filters:

- Standard CAN ID: 1, 2, 3, 4
- Extended CAN ID: 1, 2, 3, 4

It will toggle the green LED in case CAN frames are received.

Using the two buttons it will send data to the CAN bus with standard CAN ID 1, or extended CAN ID 2.

### 4.8.2 Directory contents

This example can be found in the [~ /Examples/BOARD\\_APM32F445\\_EVAL/CAN/CAN\\_RxFifo](#) directory.

## 4.9 CLOCK\_AllClocksArrayConfiguration

### 4.9.1 Example Description

This example describes how to config serval clocks\_configuration of MCU device.

All the clocks of user configuration are displayed on serial assistant through LPUART1.

LED\_BLUE blinks.

### 4.9.2 Directory contents

This example can be found in the [~/Examples/BOARD\\_APM32F445\\_EVAL/CLOCK /CLOCK\\_ArrayConfiguration](#) directory.

## 4.10 CLOCK\_AllClocksConfiguration

### 4.10.1 Example Description

This example describes how to config all the clocks of MCU device at the same time.

All the clocks of user configuration are displayed on serial assistant through LPUART1.

LED\_BLUE blinks.

### 4.10.2 Directory contents

This example can be found in the [~/Examples/BOARD\\_APM32F445\\_EVAL/CLOCK /CLOCK\\_AllClocksConfiguration](#) directory.

## 4.11 CLOCK\_SeparateConfig

### 4.11.1 Example Description

This example describes how to config some clocks which are needed for use.

Thoes clocks frequency are displayed on serial assistant through LPUART1.

LED\_BLUE blinks.

### 4.11.2 Directory contents

This example can be found in the [~/Examples/BOARD\\_APM32F445\\_EVAL/CLOCK /CLOCK\\_SeparateConfig](#) directory.

## 4.12 COMP\_DAC

### 4.12.1 Example Description

The purpose of this demo application is to show you how to use the Analog Comparator of the APM32F445 MCU using the SDK API.

The Comparator is configured to compare analog input 0 (AIN0) with 1/3 of the reference voltage generated by the internal DAC. Based on the input voltage, the LEDs light according the following rules:

- $V_{in} < \text{DAC voltage}$  : RED on, GREEN off
- $V_{in} > \text{DAC voltage}$  : RED off, GREEN on
- Unknown state : RED on, GREEN on.

### 4.12.2 Directory contents

This example can be found in the [~ /Examples/BOARD\\_APM32F445\\_EVAL/COMP /COMP \\_DAC](#) directory.

## 4.13 CRC\_Checksum

### 4.13.1 Example Description

This example application showing the usage of the CRC module.

After reset, both GREEN and RED LEDs on the EVAL board are off. Initializes CRC module with CCITT 16 bits standard configuration.

Pressing the KEY1 button, CCITT 16 CRC calculation is executed.

If the result is correct, CCITT 32 CRC calculation is executed.

If both operation results are correct, GREEN LED is turned on.

Otherwise the RED LED will be turned on.

### 4.13.2 Directory contents

This example can be found in the [~ /Examples/BOARD\\_APM32F445\\_EVAL/CRC /CRC C hecksum](#) directory.

## 4.14 CSEC\_Security



#### 4.14.1 Example Description

This example shows the usage of the CSEc module. It contains three parts which can be enabled in main.c:

- TEST\_SECURE\_BOOT: Shows Secure Boot functionalities.
- TEST\_ENCRYPT\_DECRYPT: Shows AES-128 encryption/decryption and CMAC generation/verification functionalities.
- FLASH\_FACTORY\_RESET: Reset the flash to factory state and erase all keys.

#### 4.14.2 Directory contents

This example can be found in the [~ /Examples/BOARD\\_APM32F445\\_EVAL/CSEC /CSEC\\_Security](#) directory.

### 4.15 EINJ\_Injection

#### 4.15.1 Example Description

This example describes that how to use the EINJ module to inject error, and use EREP to detect error. If KEY1 is pressed, then EINJ will inject a single bit error to SRAML, and LED\_BLUE On, if KEY2 is pressed, then EINJ will inject a non correctable error to SRAML, and LED\_RED On.

The information will export to upper computer by LPUART1.

#### 4.15.2 Directory contents

This example can be found in the [~ /Examples/BOARD\\_APM32F445\\_EVAL/EINJ /EINJ\\_Injection](#) directory.

### 4.16 EREP\_Report

#### 4.16.1 Example Description

This example describes that how to use the interrupt of EREP module.

If KEY1 is pressed, the EINJ will inject a single bit error to SRAML, EREP will catch the error and enter the single\_fault interrupt, LED\_BLUE On.

If KEY2 is pressed, the EINJ will inject a non correctable error to SRAML, EREP will catch the error and enter the double\_fault interrupt, LED\_RED On.

---

The error detail will export to upper computer by LPUART1.

#### **4.16.2 Directory contents**

This example can be found in the [~ /Examples/BOARD\\_APM32F445\\_EVAL/EREP/EREP\\_Report\\_](#) directory.

### **4.17 EWDT\_Interrupt**

#### **4.17.1 Example Description**

This example shows that how to use the EWDT\_out signal and interrupt of EWDT.

After reset, the EWDT and SysTick will be enable, the EWDT is refreshed every 250 milliseconds, LED\_BLUE keeps toggle. When press KEY, system will stop refresh EWDT, and then EWDT time out, LED\_RED On and EWDT\_out signal output low-level signal by PE4.

The information will export to upper computer by LPUART1.

#### **4.17.2 Directory contents**

This example can be found in the [~ /Examples/BOARD\\_APM32F445\\_EVAL/EWDT/EWDT\\_Interrupt\\_](#) directory.

### **4.18 FLASH\_EraseSector**

#### **4.18.1 Example Description**

This demo provides the usage of the flash erase sector and program.

Erase sector: the last sector of Pflash. If success, LED\_BLUE on.

Program address: the last sector of Pflash.

Program size: 100 bytes. If success, LED\_RED on.

The operation successful or failed is displayed on serial assistant through LPUART1.

#### **4.18.2 Directory contents**

This example can be found in the [~ /Examples/BOARD\\_APM32F445\\_EVAL/FLASH /FLASH\\_EraseSector\\_](#) directory.

## 4.19 FLASH\_Partition

### 4.19.1 Example Description

This demo provides the usage of the CFGNVM partition.

And write data to EERAM.

If CFGRam available for EEPROM successful, turn on LED\_GREEN. Or not, turn on LED\_RED.

If write data to EERAM successful, turn on LED\_BLUE. Or not, turn on LED\_RED.

Partition:

EEPROM size = 4 Kbytes

EEPROM backup size = 32 Kbytes.

### 4.19.2 Directory contents

This example can be found in the [~ /Examples/BOARD\\_APM32F445\\_EVAL/FLASH/FLASH\\_Partition](#) directory.

## 4.20 FLASH\_ProgramOnce

### 4.20.1 Example Description

This demo shows how to Program Once to the Program Once Field with record index 0x07 .

The operation successful or failed is displayed on serial assistant through LPUART1.

And the data of Program Once Field record index 0x07 is displayed on serial assistant through LPUART1.

### 4.20.2 Directory contents

This example can be found in the [~ /Examples/BOARD\\_APM32F445\\_EVAL/FLASH /FLASH\\_ProgramOnce](#) directory.

## 4.21 FLASH\_ProgramSection

### 4.21.1 Example Description

This demo provides the usage of the flash program section function.

Program the last sector of pflash.

If Program Section Successful, turn on LED\_GREEN. Or not, turn on LED\_RED.

The operation successful or failed is displayed on serial assistant through LPUART1.

#### **4.21.2 Directory contents**

This example can be found in the [~ /Examples/BOARD\\_APM32F445\\_EVAL/FLASH /FLASH\\_ProgramSection](#) directory.

### **4.22 FLASH\_Protection**

#### **4.22.1 Example Description**

This demo shows how to Configure the flash protection(PFlash, DFlash, EERam).

And try to write data to the protected region.

The operation successful or failed is displayed on serial assistant through LPUART1.

And the value of flash protection is displayed on serial assistant through LPUART1.

#### **4.22.2 Directory contents**

This example can be found in the [~ /Examples/BOARD\\_APM32F445\\_EVAL/FLASH /FLASH\\_Protection](#) directory.

### **4.23 CFGIO\_I2C\_Master**

#### **4.23.1 Example Description**

This example shows the usage of the CFGIO\_I2C driver. It uses the CFGIO\_I2C as I2C master, and the LPI2C as I2C slave. The following connections must be done for this example to work:

CFGIO\_I2C:

- SDA: PTA1 (CFGIO\_D3)
- SCL: PTA0 (CFGIO\_D2)

LPI2C:

- SDA: PTA2
- SCL: PTA3.

### 4.23.2 Directory contents

This example can be found in the [~ /Examples/BOARD\\_APM32F445\\_EVAL/CFGIO/CFGIO\\_I2C\\_Master](#) directory.

## 4.24 CFGIO\_SPI\_Dual

### 4.24.1 Example Description

This use case uses the CFGIO module to simulate SPI communication, You need to prepare two development boards, SPI\_MASTER macro definition exists in the code, and when it is written as 1, the SPI is master, and when it is written as 0, SPI is slave. The specific wiring is as follows:

master		slave	
PD9	==>	PD9	mosi
PA11	==>	PA11	miso
PA0	==>	PA0	sck
PA1	==>	PA1	ss

If communication is successful, The green LED will light up, and if communication fails, the red LED will light up. The corresponding data will be output through the serial port.

### 4.24.2 Directory contents

This example can be found in the [~ /Examples/BOARD\\_APM32F445\\_EVAL/CFGIO/CFGIO\\_SPI\\_Dual](#) directory.

## 4.25 CFGIO\_SPI\_Irq

### 4.25.1 Example Description

This use case uses the CFGIO module to simulate SPI communication, using the following pin configuration:

master:

CFGIO\_D0 ==> PTD9 ==> MOSI

CFGIO\_D1 ==> PTA11 ==> MISO

CFGIO\_D2 ==> PTA0 ==> SCK

CFGIO\_D3 ==> PTA1 ==> SS

slave:

CFGIO\_D4 ==> PTA2 ==> MOSI

CFGIO\_D5 ==> PTA3 ==> MISO

CFGIO\_D6 ==> PTA8 ==> SCK

CFGIO\_D7 ==> PTA9 ==> SS

You only need one development board, the wiring is as follows:

PTD9 ==> PTA2

PTA11 ==> PTA3

PTA0 ==> PTA8

PTA1 ==> PTA9

If the communication is successful, the green LED will be lit, and if it fails, the red LED will be lit. The corresponding data will be output through the serial port.

## 4.25.2 Directory contents

This example can be found in the [~ /Examples/BOARD\\_APM32F445\\_EVAL/CFGIO/CFGIO\\_SPI\\_Irq](#) directory.

## 4.26 CFGIO\_UART\_Echo

### 4.26.1 Example Description

This example shows the usage of the CFGIO\_UART driver.

It uses CFGIO\_D2 and CFGIO\_D3 as UART port (115200, 8N1).

It prints a welcome message on startup, and accepts user input commands to switch on the LEDs. User can enter "blue", "green", "red" commands, then the blue, green, or red LEDs will be switched on.

The following connections must be done for this example to work:

- RX: PTA1 (CFGIO\_D3)

- TX: PTA0 (CFGIO\_D2)

- GND.

#### 4.26.2 Directory contents

This example can be found in the [~ /Examples/BOARD\\_APM32F445\\_EVAL/CFGIO/CFGIO\\_UART\\_Echo](#) directory.

### 4.27 CFGTMR\_InputCapture

#### 4.27.1 Example Description

This example shows how to configure the CFGTMR0 peripheral to generate a PWM signal, and then CFGTMR1 peripheral to measure the frequency of the PWM signal in input capture mode.

After Reset, the initial frequency of PWM output is 1KHz. Then you can press KEY1 to change PWM signal frequency. The frequency will range from 1KHz to 50KHz, each time the button is pressed, the frequency increases by 1KHz. You can view the measurable frequency value through serial terminal.

CFGTMR count clock configuration:

- CFGTMR0 counter clock:  $\text{CFGTMRClockSource} / \text{CFGTMRPrescaler} = 48\text{MHz} / 1 = 48\text{MHz}$ .

- CFGTMR1 counter clock:  $\text{CFGTMRClockSource} / \text{CFGTMRPrescaler} = 48\text{MHz} / 1 = 48\text{MHz}$ .

#### 4.27.2 Directory contents

This example can be found in the [~ /Examples/BOARD\\_APM32F445\\_EVAL/CFGTMR /CFGTMR\\_InputCapture](#) directory.

### 4.28 CFGTMR\_PeriodicInterrupt

#### 4.28.1 Example Description

This example shows how to configure the CFGTMR0 peripheral in timer mode.

Then generate periodic timer interrupts with the corresponding interrupt request.

The CFGTMR\_CLOCK frequency is set to system clock (48MHz), the prescaler is 1.

So the CFGTMR counter clock is  $48\text{MHz} / 1 = 48\text{MHz}$ . CFGTMR final counter value is configured 48000, so the interrupt period is  $48000 / 48\text{MHz} = 1\text{ms}$ .

The phenomenon:

In the interrupt handler routine, when the counter value reaches 1000, the LED status is toggled.

#### 4.28.2 Directory contents

This example can be found in the [~ /Examples/BOARD\\_APM32F445\\_EVAL/CFGTM0 /CFGTM0\\_PeriodicInterrupt](#) directory.

### 4.29 CFGTM0\_PWMOutput

#### 4.29.1 Example Description

This example shows how to configure the CFGTM0 peripheral to generate a PWM signal with different frequency and duty cycles.

After Reset, the initial frequency of the PWM signal is 10KHz, and the duty cycle is 25%. You can press KEY1 to change the PWM signal frequency, and press KEY2 to change the PWM signal duty cycle. The waveform can be displayed using an oscilloscope. You can also view the frequency and duty cycle through serial terminal.

The CFGTM0\_CLOCK frequency is set to system clock (48MHz), the prescaler is 1.

So the CFGTM0 counter clock is  $48\text{MHz} / 1 = 48\text{MHz}$ .

#### 4.29.2 Directory contents

This example can be found in the [~ /Examples/BOARD\\_APM32F445\\_EVAL/CFGTM0 /CFGTM0\\_PWMOutput](#) directory.

### 4.30 CFGTM0\_CenterAlignedPWMOutput

#### 4.30.1 Example Description

This example shows how to configure the CFGTM0 peripheral in center aligned PWM mode. Then to generate two complementary PWM signals with center alignment.

The CFGTM0\_CLOCK frequency is set to system clock (48MHz), the prescaler is 1.

So the CFGTM0 counter clock is  $48\text{MHz} / 1 = 48\text{MHz}$ .

The phenomenon:

The channel output frequency is set to 10KHz.



$\text{CFGTMRO\_CH0} = 0x4000 / 0x8000 = 50\%$  , so CFGTMRO\_CH1 is set to 50%.

$\text{CFGTMRO\_CH2} = 0x2000 / 0x8000 = 25\%$  , so CFGTMRO\_CH3 is set to 75%.

The CFGTMRO\_CH0/CFGTMRO\_CH1 is the first channel pair, the CFGTMRO\_CH2/CFGTMRO\_CH3 is the second channel pair. The second channel pair is center aligned compared to the first channel pair.

#### 4.30.2 Directory contents

This example can be found in the [~ /Examples/BOARD\\_APM32F445\\_EVAL/CFGTMRO /CFGTMRO\\_CenterAlignedPWMOutput](#) directory.

### 4.31 CFGTMRO\_CombinedPWMOutput

#### 4.31.1 Example Description

This example shows how to configure the CFGTMRO peripheral in Combined PWM mode, and then generate 4 different PWM waveform.

The CFGTMRO\_CLOCK frequency is set to system clock (48MHz), the prescaler is 2.

So the CFGTMRO counter clock is  $48\text{MHz} / 2 = 24\text{MHz}$ .

The phenomenon:

The CFGTMRO\_CH0/CFGTMRO\_CH1 is the first channel pair, the CFGTMRO\_CH2/CFGTMRO\_CH3 is the second channel pair, they are work in the complementary mode with a 50% duty cycle and 20KHz frequency.

The second channel pair (CFGTMRO\_CH2/CFGTMRO\_CH3) is phase-shifted by  $90^\circ$  to the first channel pair (CFGTMRO\_CH0/CFGTMRO\_CH1).

#### 4.31.2 Directory contents

This example can be found in the [~ /Examples/BOARD\\_APM32F445\\_EVAL/CFGTMRO /CFGTMRO\\_CombinedPWMOutput](#) directory.

### 4.32 CFGTMRO\_DeadtimeInsertPWMOutput

#### 4.32.1 Example Description

This example shows how to configure the CFGTMRO peripheral to generate complementary PWM signals, and to insert 4us dead time value.

The CFGTMR\_CLOCK frequency is set to system clock (48MHz), the prescaler is 1.

So the CFGTMR0 counter clock is  $48\text{MHz} / 1 = 48\text{MHz}$ .

The PWM signal parameter is computed as the following description:

- Frequency: Specified 10KHz
- Duty cycle:  $\text{CFGTMRO\_CH0} = 0x4000 / 0x8000 = 50\%$  , so CFGTMR0\_CH1 is set to 50%.
- Dead time =  $\text{deadTimeValue} / (\text{CFGTMR\_CLOCK} / \text{deadTimePsc}) = 12 / (48\text{MHz} / 16) = 4\mu\text{s}$ .

#### 4.32.2 Directory contents

This example can be found in the [~ /Examples/BOARD\\_APM32F445\\_EVAL/CFGTMR /CFGTMRO\\_DeadtimeInsertPWMOutput](#) directory.

### 4.33 CFGTMR0\_EdgeAlignedPWMOutput

#### 4.33.1 Example Description

This example shows how to configure the CFGTMR0 peripheral in edge aligned PWM mode. Then to generate two complementary PWM signals with edge alignment.

The CFGTMR\_CLOCK frequency is set to system clock (48MHz), the prescaler is 1.

So the CFGTMR0 counter clock is  $48\text{MHz} / 1 = 48\text{MHz}$ .

The phenomenon:

The channel output frequency is set to 10KHz.

$\text{CFGTMRO\_CH0} = 0x4000 / 0x8000 = 50\%$  , so CFGTMR0\_CH1 is set to 50%.

$\text{CFGTMRO\_CH2} = 0x2000 / 0x8000 = 25\%$  , so CFGTMR0\_CH3 is set to 75%.

The CFGTMR0\_CH0/CFGTMR0\_CH1 is the first channel pair, the CFGTMR0\_CH2/CFGTMR0\_CH3 is the second channel pair. The second channel pair is edge aligned compared to the first channel pair.

#### 4.33.2 Directory contents

This example can be found in the [~ /Examples/BOARD\\_APM32F445\\_EVAL/CFGTMR /CFGTMRO\\_EdgeAlignedPWMOutput](#) directory.

## 4.34 CFGTMR0\_FaultControl

### 4.34.1 Example Description

This example shows how to configure the fault-control feature of the CFGTMR.

In this example, two CFGTMR modules are initiated, while CFGTMR0 is configured to generate the edge-align PWM signal. The CFGTMR1 is used to generate the fault PWM signal, and then to affects the PWM output of the CFGTMR0 channels externally through the fault input pin PE3.

### 4.34.2 Directory contents

This example can be found in the [~ /Examples/BOARD\\_APM32F445\\_EVAL/CFGTMR /CFGTMR0\\_FaultControl](#) directory.

## 4.35 CFGTMR1\_OCMatchSet

### 4.35.1 Example Description

This example shows how to configure the CFGTMR1 peripheral in output compare mode.

When the counter reaches the matching value, the CFGTMR1\_CH0(PB2)/CFGTMR1\_CH1(PB3)/CFGTMR1\_CH2(PA15) output channel is set with different delays.

The CFGTMR\_CLOCK frequency is set to system clock (48MHz), the prescaler is 64.

So the CFGTMR1 counter clock is  $48\text{MHz} / 64 = 750\text{KHz}$ .

Generate 3 signal with 3 different delays:

CFGTMR1\_CH0(PB2) delay: Compare value / CFGTMR1 counter clock = 0 / 750KHz = 0us.

CFGTMR1\_CH1(PB3) delay: Compare value / CFGTMR1 counter clock = 7500 / 750KHz = 10ms.

CFGTMR1\_CH2(PA15) delay: Compare value / CFGTMR1 counter clock = 15000 / 750KHz = 20ms.

### 4.35.2 Directory contents

This example can be found in the [~ /Examples/BOARD\\_APM32F445\\_EVAL/CFGTMR /CFGTMR1\\_OCMatchSet](#) directory.

## 4.36 CFGTMR1\_OCToggle

### 4.36.1 Example Description

This example shows how to configure the CFGTMR1 peripheral in output compare toggle mode, and then to generate the toggle signal.

The CFGTMR\_CLOCK frequency is set to system clock (48MHz), the prescaler is 1.

So the CFGTMR1 counter clock is  $48\text{MHz} / 1 = 48\text{MHz}$ .

The phenomenon:

Channel update rate = CFGTMR1 counter clock / maxCountValue / 2 =  $48\text{MHz} / 24000 / 2 = 1\text{KHz}$ .

CFGTMR1\_CH0(PB2) delay: Compare value / CFGTMR1 counter clock =  $0 / 48\text{MHz} = 0\mu\text{s}$ .

CFGTMR1\_CH1(PB3) delay: Compare value / CFGTMR1 counter clock =  $1200 / 48\text{MHz} = 25\mu\text{s}$ .

CFGTMR1\_CH2(PA15) delay: Compare value / CFGTMR1 counter clock =  $2400 / 48\text{MHz} = 50\mu\text{s}$ .

### 4.36.2 Directory contents

This example can be found in the [~ /Examples/BOARD\\_APM32F445\\_EVAL/CFGTMR /CFGTMR1 OCToggle](#) directory.

## 4.37 CFGTMR2\_QDPhaseEncoding

### 4.37.1 Example Description

This example shows how to configure the CFGTMR2 peripheral in the quadrature decoder mode of the Phase A and Phase B encoding sub-mode.

Configuration step:

- Configure CFGTMR0 to generate the quadrature signals, as CFGTMR2 A/B phase input signals.
- Configure CFGTMR2 in the phase encoding mode of the quadrature decoder mode.
- CFGTMR2 Counter initial value 0, maximum value is 10.
- Every time 10 rising/falling edges of the Phase A and Phase B signals are detected, the CFGTMR2 counter generate overflow interrupt. Then PD1 PIN output toggle.

The phenomenon:

You can use an oscilloscope to observe the relationship between the A/B phase signal and the PD1 PIN output signal.

#### 4.37.2 Directory contents

This example can be found in the [~ /Examples/BOARD\\_APM32F445\\_EVAL/CFGTMR /CFGTMR2\\_QDPhaseEncoding](#) directory.

### 4.38 LPI2C\_TwoBoardsDMA\_Master

#### 4.38.1 Example Description

This example shows how to configure LPI2C peripherals to communicate between two different boards in DMA mode. In this example, LPI2C is operated as the Master, then wait for the user to input information in the serial terminal, and choose whether to send or receive data based on user's input information.

For Master board:

Please enter 1 or 2:

1) LPI2C Master write data.

2) LPI2C Master read data.

Enter your input:

You must first select Send/Receive on Slave board, after that select Receive/Send on Master board.

If master and slave transfer is successfully completed, slave buffer and master buffer will be checked.

If the data buffer is equal, turn on the green LED, otherwise turn on the red LED.

You can view the transfer information through the serial terminal.

#### 4.38.2 Directory contents

This example can be found in the [~ /Examples/BOARD\\_APM32F445\\_EVAL/LPI2C /LPI2C\\_TwoBoardsDMA\\_Master](#) directory.

### 4.39 LPI2C\_TwoBoardsDMA\_Slave

#### 4.39.1 Example Description

This example shows how to configure LPI2C peripherals to communicate between two different boards. In this example, LPI2C is operated as the Slave, then wait for the user to input information in the serial terminal, and choose whether to send or receive data based on user's input information. The slave is configured in interrupt mode when sending data and DMA mode when receiving data.

For Slave board:

Please enter 1 or 2:

1) LPI2C Slave write data.

2) LPI2C Slave read data.

Enter your input:

You must first select Send/Receive on Slave board, after that select Receive/Send on Master board.

If master and slave transfer is successfully completed, slave buffer and master buffer will be checked.

If the data buffer is equal, turn on the green LED, otherwise turn on the red LED.

You can view the transfer information through the serial terminal.

#### **4.39.2 Directory contents**

This example can be found in the [~/Examples/BOARD\\_APM32F445\\_EVAL/LPI2C /LPI2C\\_TwoBoardsDMA\\_Slave](#) directory.

### **4.40 LPI2C\_TwoBoardsInterrupt\_Master**

#### **4.40.1 Example Description**

This example shows how to configure LPI2C peripherals to communicate between two different boards in interrupt mode. Run Slave board first, after that Run Master board.

First, Master sends data to the slave, and then wait the ACK of the slave. Then Master receive data from the slave to the master buffer.

If the transfer is successful, Green LED will turn on, otherwise red LED will turn.

You can view the transmission information through the UART.

#### **4.40.2 Directory contents**

This example can be found in the [~ /Examples/BOARD\\_APM32F445\\_EVAL/LPI2C /LPI2C\\_TwoBoardsInterrupt\\_Master](#) directory.

## 4.41 LPI2C\_TwoBoardsInterrupt\_Slave

### 4.41.1 Example Description

This example shows how to configure LPI2C peripherals to communicate between two different boards in interrupt mode. Run Slave board first, after that Run Master board.

The LPI2C Slave is configured in listening mode, using interrupt callback functions to handle request events.

If the transfer is successful, Green LED will turn on, otherwise red LED will turn.

You can view the transmission information through the UART.

### 4.41.2 Directory contents

This example can be found in the [~ /Examples/BOARD\\_APM32F445\\_EVAL/LPI2C /LPI2C\\_TwoBoardsInterrupt\\_Slave](#) directory.

## 4.42 LPITMR\_Counter

### 4.42.1 Example Description

This example shows how to use the counter mode of the LPITMR module.

After reset, LPITMR works on counter mode, it will enter interrupt every 500 milliseconds and then toggle the LED\_BLUE status.

### 4.42.2 Directory contents

This example can be found in the [~ /Examples/BOARD\\_APM32F445\\_EVAL/LPITMR /LPITMR\\_Counter](#) directory.

## 4.43 LPITMR\_Timer

### 4.43.1 Example Description

This example shows that how to use the timer mode of the LPITMR module.

After reset, LPITMR work on timer mode, it will enter interrupt every 500 milliseconds and then toggle the LED\_BLUE status.

#### 4.43.2 Directory contents

This example can be found in the [~/Examples/BOARD\\_APM32F445\\_EVAL/LPITMR /LPITMR\\_Timer](#) directory.

### 4.44 LPSPI\_DMA

#### 4.44.1 Example Description

This use case uses hardware SPI communication, You need to prepare two development boards, SPI\_MASTER macro definition exists in the code, and when it is written as 1, the SPI is master, and when it is written as 0, SPI is slave. The specific wiring is as follows:

master		slave	
PD2	==>	PD1	mosi
PD1	==>	PD2	miso
PB14	==>	PB14	sck
PD3	==>	PD3	ss

If communication is successful, the green LED will light up, and if communication fails, the red LED will light up. The corresponding data will be output through the serial port.

#### 4.44.2 Directory contents

This example can be found in the [~/Examples/BOARD\\_APM32F445\\_EVAL/LPSPI/LPSPI\\_DMA](#) directory.

### 4.45 LPSPI\_Isr

#### 4.45.1 Example Description

This use case uses hardware SPI communication, You need to prepare two development boards, SPI\_MASTER macro definition exists in the code, and when it is written as 1, the SPI is master, and when it is written as 0, SPI is slave. The specific wiring is as follows:

master		slave	
PD2	==>	PD1	mosi



---

PD1	==>	PD2	miso
PB14	==>	PB14	sck
PD3	==>	PD3	ss

If communication is successful, the green LED will light up, and if communication fails, the red LED will light up. The corresponding data will be output through the serial port.

#### 4.45.2 Directory contents

This example can be found in the [~/Examples/BOARD\\_APM32F445\\_EVAL/LPSPI /LPSPI\\_lsr](#) directory.

### 4.46 LPTMR\_PeriodicInterrupt

#### 4.46.1 Example Description

This example shows how to configure the LPTMR peripheral in timer mode, and then generate periodic timer interrupts.

The phenomenon is that the green LED blinks alternately between 0.5s and 1s.

#### 4.46.2 Directory contents

This example can be found in the [~/Examples/BOARD\\_APM32F445\\_EVAL/LPTMR /LPTMR\\_PeriodicInterrupt](#) directory.

### 4.47 LPTMR\_PulseCounter

#### 4.47.1 Example Description

This example shows how to configure the LPTMR peripheral in pulse counter mode, and the detect the number of pulses, this pulse signal comes from the PD1 PIN.

When the KEY1(PC12) is pressed 5 times, an interrupt is triggered, then green LED is toggled, and changes the pulse counter compare value to 2. The press KEY1 3 times again, green LED is toggled again. Repeat the process.

If you press KEY2(PC13), it will change the pulse input source pin to PD5.

#### 4.47.2 Directory contents

This example can be found in the [~ /Examples/BOARD\\_APM32F445\\_EVAL/LPTMR /LPTMR\\_PulseCounter](#) directory.

## 4.48 LIN\_Master

### 4.48.1 Example Description

LIN MASTER

Example that shows the usage of the LIN driver in master mode.

Application description:

This example demonstrates the LIN communication between APM32F445 EVAL Master and Slave using LIN driver.

A frame contains header and data. The Master node can send header and data, but Slave node only can send data. Based on header, Master node or Slave node will take corresponding action. On Master node:

Press KEY1:

- For the first time, Master node sends FRAME\_MASTER\_RX\_DATA header and require slave node responds by sending data.
- For the second time, Master sends FRAME\_SLAVE\_RX\_DATA header, then continue sending data and slave node will receive the data.
- If node successful receives data, this node will turn on GREEN\_LED, otherwise turn on RED\_LED.

Press KEY2:

Master node will check current node state. If the state is LIN\_NODE\_SLP\_MODE,

Master node will send wakeup signal and BLUE\_LED will be turned on both nodes.

Otherwise Master node will send header to set Master node and Slave node to sleep mode and all LEDs will be turned off on both nodes.

### 4.48.2 Directory contents

This example can be found in the [~ /Examples/BOARD\\_APM32F445\\_EVAL/LPUART LIN/ LIN\\_Master](#) directory.

## 4.49 LIN\_Slave

### 4.49.1 Example Description

#### LIN SLAVE

Example shows the usage of the LIN driver in slave mode.

Application description:

This example demonstrates the LIN communication between APM32F445 EVAL Master and Slave using LIN driver.

A frame contains header and data. The Master node can send header and data, but Slave node only can send data. Base on header, Master node or Slave node will take corresponding action.

- If Slave node receives FRAME\_MASTER\_RX\_DATA header, Slave node will respond by sending data.
- If Slave node receives FRAME\_SLAVE\_RX\_DATA header, Slave node will receive and check data. If data is success, Slave node will turn on GREEN\_LED, otherwise turn on RED\_LED.
- If Slave node receives FRAME\_ENTER\_SLP header, Slave node will go to sleep mode and turn off all led.
- If Slave node receives a wakeup signal, it will check current node state, if the node state is sleep mode, Slave node will wakeup and turn on BLUE\_LED, otherwise wakeup signal is aborted and keep the previous state.

### 4.49.2 Directory contents

This example can be found in the [~ /Examples/BOARD\\_APM32F445\\_EVAL/LPUART\\_LIN/LIN\\_Slave](#) directory.

## 4.50 LPUART\_DMA

### 4.50.1 Example Description

The purpose of this demo application is to show you how to use the LPUART from the APM32F445 MCU using the SDK with DMA.

You need to connect a serial console to UART1. The application will print a brief message describing the example behavior at startup. Then you can input a string and press ENTER. It will echo back any string that is sent from the console.

If 'Hi,Geehy\n' are sent, the board will reply with 'Welcome'.

#### **4.50.2 Directory contents**

This example can be found in the [~ /Examples/BOARD\\_APM32F445\\_EVAL/LPUART\\_LIN/LPUART\\_DMA](#) directory.

### **4.51 LPUART\_SendReceive**

#### **4.51.1 Example Description**

The purpose of this demo application is to show you how to use the LPUART from the APM32F445 MCU using the SDK.

You need to connect a serial console to UART1. The application will print a brief message describing the example behavior at startup. Then you can input a string and press ENTER. It will echo back any string that is sent from the console.

If 'Hi,Geehy\n' are sent, the board will reply with 'Welcome'.

#### **4.51.2 Directory contents**

This example can be found in the [~ /Examples/BOARD\\_APM32F445\\_EVAL/LPUART\\_LIN/LPUART\\_SendReceive](#) directory.

### **4.52 MPU\_MemoryProtection**

#### **4.52.1 Example Description**

This example describes how to use the memory protection function of the MPU.

Press the KEY1 will read data in the protected memory, it will trigger a memory error, if the error information is the same as the expected, the LED will change from green to red.

#### **4.52.2 Directory contents**

---

This example can be found in the [~/Examples/BOARD\\_APM32F445\\_EVAL/MPU /MPU MemoryProtection](#) directory.

## 4.53 PDU\_PeriodicInterrupt

### 4.53.1 Example Description

This example shows how to configure the PDU peripheral, and then generate periodic timer interrupts.

The phenomenon is that the green LED blinks continuously for 500ms.

### 4.53.2 Directory contents

This example can be found in the [~/Examples/BOARD\\_APM32F445\\_EVAL/PDU /PDU PeriodicInterrupt](#) directory.

## 4.54 PINS\_Isr

### 4.54.1 Example Description

This use case is used to detect external interrupts, press the key1 button to turn on LED\_RED, and press the key2 button to turn off LED\_RED.

### 4.54.2 Directory contents

This example can be found in the [~/Examples/BOARD\\_APM32F445\\_EVAL/PINS /PINS\\_Isr](#) directory.

## 4.55 PINS\_Led

### 4.55.1 Example Description

This use case is used to test the basic functions of the pins module.

At the beginning, the all LEDs are always on, press the key1 button to turn on the LED\_RED, and press the key2 button to turn off all LEDs.

### 4.55.2 Directory contents

This example can be found in the [~/Examples/BOARD\\_APM32F445\\_EVAL/PINS /PINS\\_L ed](#) directory.

## 4.56 PINS\_Read

### 4.56.1 Example Description

This use case is used to view the status of pins after they are disabled.

### 4.56.2 Directory contents

This example can be found in the [~/Examples/BOARD\\_APM32F445\\_EVAL/PINS /PINS\\_R ead](#) directory.

## 4.57 Power\_SwitchMode

### 4.57.1 Example Description

This demo shows how to switch to other power mode by LPUART1.

And press KEY1/KEY2 to wake up the CPU from STOP1,STOP2 or VLPS mode.

1 for HSRUN

2 for RUN

3 for VLPR

4 for STOP1

5 for STOP2

6 for VLPS

Phenomenon:

If switch to STOP1, STOP2 or VLPS mode, turn on LED\_RED.

And exit STOP1, STOP2 or VLPS mode, turn on LED\_GREEN.

### 4.57.2 Directory contents

This example can be found in the [~/Examples/BOARD\\_APM32F445\\_EVAL/POWER /Power\\_SwitchMode](#) directory.

## 4.58 RTC\_Alarm

### 4.58.1 Example Description

This example shows how to use the alarm of RTC module. Press key1 or key2 to create an alarm event, when the alarm clock in active state, LED\_GREEN will keep on, when the alarm clock has arrived, LED\_GREEN turn off.

LPUART1 will export information.

### 4.58.2 Directory contents

This example can be found in the [~ /Examples/BOARD\\_APM32F445\\_EVAL/RTC/RTC\\_Alarm](#) directory.

## 4.59 RTC\_ClockOut

### 4.59.1 Example Description

This example shows how to use the clock output function of RTC module.

After reset, RTC will general square wave from clock\_out pin.

### 4.59.2 Directory contents

This example can be found in the [~ /Examples/BOARD\\_APM32F445\\_EVAL/RTC/RTC\\_ClockOut](#) directory.

## 4.60 RTC\_Second

### 4.60.1 Example Description

This example shows how to use the seconds interrupt of RTC module to implement a clock with calendar. when the RTC working, the LED flash per second, and LPUART1 will export the current time to upper computer.

### 4.60.2 Directory contents

This example can be found in the [~ /Examples/BOARD\\_APM32F445\\_EVAL/RTC/RTC\\_Second](#) directory.

## 4.61 FreeRTOS

### 4.61.1 Example Description

This example describes how to use FreeRTOS for APM32F445.

For this example, You can see that LED\_GREEN is constantly blinking, and printing information through the serial assistant.

### 4.61.2 Directory contents

This example can be found in the [~/Examples/BOARD\\_APM32F445\\_EVAL/RTOS/FreeRTOS](#) directory.

## 4.62 RT-Thread

### 4.62.1 Example Description

This example describes how to use RT-Thread for APM32F445.

For this example, You can see that LED\_GREEN is constantly blinking, and printing information through the serial assistant.

### 4.62.2 Directory contents

This example can be found in the [~/Examples/BOARD\\_APM32F445\\_EVAL/RTOS/RT-Thread](#) directory.

## 4.63 Template

### 4.63.1 Example Description

This demo is based on the APM32F445 EVAL board. It provides a template project.

### 4.63.2 Directory contents

This example can be found in the [~/Examples/BOARD\\_APM32F445\\_EVAL/Template/Template](#) directory.

## 4.64 WDT\_FastTest

### 4.64.1 Example Description

[www.geehy.com](http://www.geehy.com)



This demo shows how to Configure the WDT with fast test.

When power on, begins WDT\_TEST\_LOW;

When reset by WDT\_TEST\_LOW, and then begins WDT\_TEST\_HIGH;

When reset by WDT\_TEST\_HIGH, and then begins WDT\_TEST\_USER, and feed dog by systick interrupt.

If put down KEY1, stop to feed dog, and it will reset the system by WDT\_TEST\_USER, and disable WDT.

Phenomenon:

If begins WDT\_TEST\_LOW, turn on LED\_BLUE.

If begins WDT\_TEST\_HIGH, turn on LED\_BLUE.

If begins WDT\_TEST\_USER, turn on LED\_GREEN.

If reset by WDT\_TEST\_USER, turn on LED\_RED.

#### 4.64.2 Directory contents

This example can be found in the [~/Examples/BOARD\\_APM32F445\\_EVAL/WDT/WDT\\_FastTest](#) directory.

### 4.65 WDT\_Interrupt

#### 4.65.1 Example Description

This demo shows how to Configure the WDT with Window mode, and feed dog by systick interrupt.

If put down KEY1, stop to feed dog, and it will reset the system by two ways as follows:

1. #define RCM\_WDT\_INT 0

Triggered on WDOG timeout, reset delayed by 128 BUS\_CLK cycles(2.6us, 48MHz).

2. #define RCM\_WDT\_INT 1

Triggered after WDOG interrupt delay, reset delayed by 514 LPO cycles(~4ms, 128kHz).

Phenomenon:

If Configure successful, turn on LED\_GREEN.

If reset by WDT, turn on LED\_RED.

PD16 will Toggles when CNT overflows, check it with analyzer.

#### **4.65.2 Directory contents**

This example can be found in the [~ /Examples/BOARD\\_APM32F445\\_EVAL/WDT/WDT\\_Interrupt](#) directory.

## 5 About Libraries

The libraries folder includes a series library. It can provide supports for APM32F445\_446 MCU such as device support and standard peripheral. The libraries can be found in the [~/Libraries](#) directory.

APM32F445\_446 MCU include following library:

- Libraries folder
  - \* CMSIS
  - \* Device
  - \* APM32F445\_446\_PeripheralLibrary

## 6 About Middlewares

The middlewares folder includes a series third-party middleware. The middlewares can be found in the [~/middlewares](#) directory.

The middlewares used by APM32F445\_446 EVAL include following:

- Middlewares folder
  - \* FreeRTOS
  - \* OSIF
  - \* RT-Thread

## 7 About Package

The package folder includes Geehy APM32F445\_446 DFP Package. The Package can be found in the [~/Package](#) directory.

The package used by APM32F445\_446 EVAL include following:

- Package folder
  - \* Geehy.APM32F445\_446\_DFP.1.0.0.pack
  - \* SVD

## 8 Revision History

Table 1 File Revision History

Date	Rev	Description
2026.01.31	1.0	First Release version of APM32F445_446 SDK

## Statement

This document is formulated and published by Geehy Semiconductor Co., Ltd. (hereinafter referred to as "Geehy"). The contents in this document are protected by laws and regulations of trademark, copyright and software copyright. Geehy reserves the right to make corrections and modifications to this document at any time. Please read this document carefully before using Geehy products. Once you use the Geehy product, it means that you (hereinafter referred to as the "users") have known and accepted all the contents of this document. Users shall use the Geehy product in accordance with relevant laws and regulations and the requirements of this document.

### 1. Ownership

This document can only be used in connection with the corresponding chip products or software products provided by Geehy. Without the prior permission of Geehy, no unit or individual may copy, transcribe, modify, edit or disseminate all or part of the contents of this document for any reason or in any form.

The “极海” or “Geehy” words or graphics with “®” or “TM” in this document are trademarks of Geehy. Other product or service names displayed on Geehy products are the property of their respective owners.

### 2. No Intellectual Property License

Geehy owns all rights, ownership and intellectual property rights involved in this document.

Geehy shall not be deemed to grant the license or right of any intellectual property to users explicitly or implicitly due to the sale or distribution of Geehy products or this document.

If any third party's products, services or intellectual property are involved in this document, it shall not be deemed that Geehy authorizes users to use the aforesaid third party's products, services or intellectual property, unless otherwise agreed in sales order or sales contract.

### 3. Version Update

Users can obtain the latest document of the corresponding models when ordering Geehy products.

If the contents in this document are inconsistent with Geehy products, the agreement in the sales order or the sales contract shall prevail.

### 4. Information Reliability

The relevant data in this document are obtained from batch test by Geehy Laboratory or cooperative third-party testing organization. However, clerical errors in correction or errors caused by differences in testing environment may occur inevitably. Therefore, users should understand that Geehy does not bear any responsibility for such errors that may occur in this document. The relevant data in this document are only used to guide users as performance parameter reference and do not constitute Geehy's guarantee for any product performance.

Users shall select appropriate Geehy products according to their own needs, and effectively verify and test the applicability of Geehy products to confirm that Geehy products meet their own needs, corresponding standards, safety or other reliability requirements. If losses are caused to users due to the user's failure to fully verify and test Geehy products, Geehy will not bear any responsibility.

## 5. Legality

USERS SHALL ABIDE BY ALL APPLICABLE LOCAL LAWS AND REGULATIONS WHEN USING THIS DOCUMENT AND THE MATCHING GEEHY PRODUCTS. USERS SHALL UNDERSTAND THAT THE PRODUCTS MAY BE RESTRICTED BY THE EXPORT, RE-EXPORT OR OTHER LAWS OF THE COUNTRIES OF THE PRODUCTS SUPPLIERS, GEEHY, GEEHY DISTRIBUTORS AND USERS. USERS (ON BEHALF OR ITSELF, SUBSIDIARIES AND AFFILIATED ENTERPRISES) SHALL AGREE AND PROMISE TO ABIDE BY ALL APPLICABLE LAWS AND REGULATIONS ON THE EXPORT AND RE-EXPORT OF GEEHY PRODUCTS AND/OR TECHNOLOGIES AND DIRECT PRODUCTS.

## 6. Disclaimer of Warranty

THIS DOCUMENT IS PROVIDED BY GEEHY "AS IS" AND THERE IS NO WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, TO THE EXTENT PERMITTED BY APPLICABLE LAW.

GEEHY WILL BEAR NO RESPONSIBILITY FOR ANY DISPUTES ARISING FROM THE SUBSEQUENT DESIGN OR USE BY USERS.

## 7. Limitation of Liability

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL GEEHY OR ANY OTHER PARTY WHO PROVIDE THE DOCUMENT "AS IS", BE LIABLE FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, DIRECT, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE DOCUMENT (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY USERS OR THIRD PARTIES).

## 8. Scope of Application



The information in this document replaces the information provided in all previous versions of the document.

© 2023 Geehy Semiconductor Co., Ltd. - All Rights Reserved