GigaDevice Semiconductor Inc.

GD32VW553H-EVAL

User Guide

Rev1.2

Tables of Contents

List of Figures

List of Tables

1.   Summary

GD32VW553H-EVAL uses GD32VW553H as the main controller. It uses GD-Link Mini USB interface to supply 5V power. Reset, Boot, Tamper/Wakeup KEY, LED, ADC, I2C, SPI_LCD, IFRP and USART to USB interface are also included. For more details, please refer to GD32VW553H-EVAL-V1.2 schematic.

2. Function Pin Assign

Table 2-1. Function pin assignment

Function	Pin	Description
LED	PA4	LED1
	PA5	LED2
	PA6	LED3
RESET		K1-Reset
KEY	PA0	K2-Wakeup
ADC	PA1	ADC_IN0
USART	PB15	USART_TX
	PA8	USART_RX
I2C	PA2	I2C0_SCL
	PA3	I2C0_SDA
IFRP	PB15	IR_OUT
	PB11	TIMER_CH2
SPI_LCD	PA12	SPI_NSS
	PB12	LCD_RESET
	PB13	LCD_D/C
	PA9	SPI_MOSI
	PA11	SPI_SCK
	PA10	SPI_MISO
QSPI Flash	PA5	QSPI_NSS
	PA4	QSPI_SCK
	PA6	QSPI_IO0
	PA7	QSPI_IO1
	PB3	QSPI_IO2
	PB4	QSPI_IO3

3. Getting started

The EVAL board uses GD-Link Mini USB connecter to get power DC +5V, which is the hardware system normal work voltage. A I-jet tool, GD-Link tool on board or JLink tool is necessary in order to download and debug programs. Select the correct boot mode and then power on.

There are IAR version, eclipse version and SES version of all projects. IAR version of the projects are created based on IAR Embedded Workbench for RISC-V 3.10.1. Eclipse version is 4.8.0. Eclipse version is SEGGER Embedded Studio for RISC-V Release 7.30. During use, the following points should be noted:

1. If you use IAR to open the project, install IAR_GD32VW55x_ADDON.exe to load the associated files.

4. Hardware layout overview

4.1. Power supply

Figure 4-1. Schematic diagram of power supply

POWER

GDLINK_5V	P1 +5V	U1 AMS1117-3.3	+3V3 R1 LEDPWR
	SMD1210P050TF	Vin Vout	470Ω
	E1 C1		E2 C2 LED0603
	16V/10uF,AVX 50V/0.1uF	GND	50V/0.1uF 16V/10uF,AVX
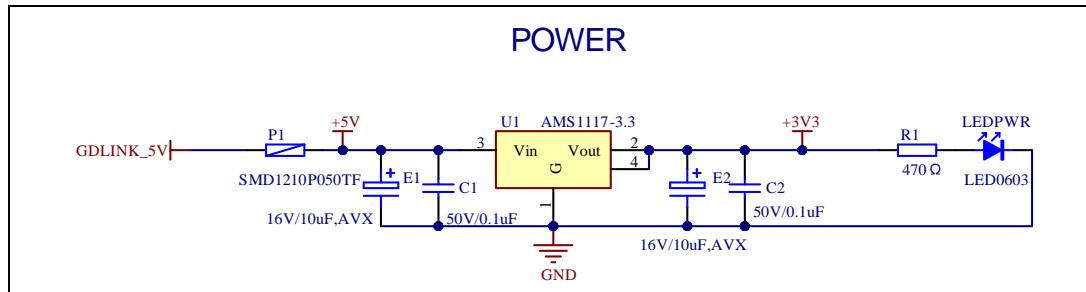
4.2. Boot option

Figure 4-2. Schematic diagram of boot option

BOOT

JP4

PB1	BOOT1	R2 +3V3	1	2	+3V3 R3	BOOT0	PC8
		10KΩ	3	4			
			5	6	GND 10KΩ		

HEADER 3X2
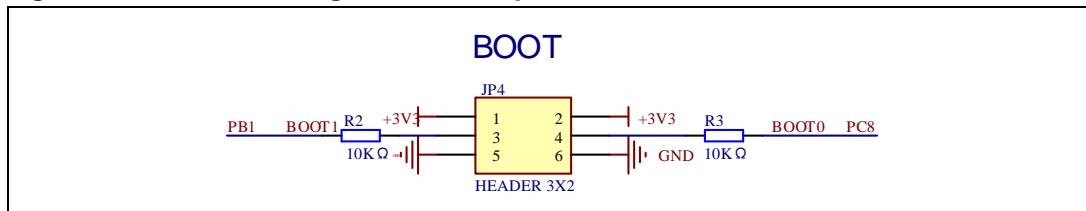
4.3. LED

Figure 4-3. Schematic diagram of LED function

LED

JP3

QSPI_SCK	1	2	QSPI_NSS
PA4	3	4	PA5
LED1	5	6	LED2

HEADER 3X2

LED1	R4	LED1	GND
	470Ω	LED0603	

LED2	R5	LED2	GND
	470Ω	LED0603	

JP5

QSPI_IO0	1
PA6	2
LED3	3

HEADER 3

LED3	R6	LED3	GND
	470Ω	LED0603	

Short JP3(1,3)(2,4) for QSPI function
Short JP5(1,2)for QSPI function
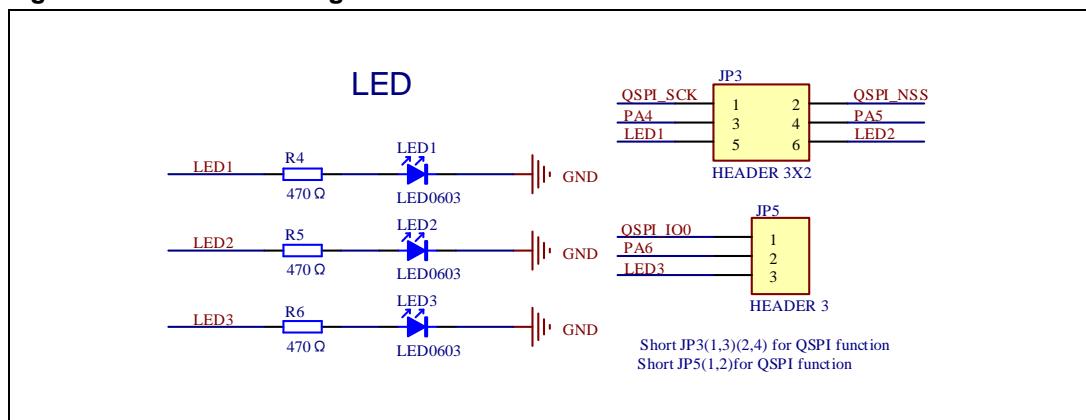
4.4. KEY

Figure 4-4. Schematic diagram of Key function
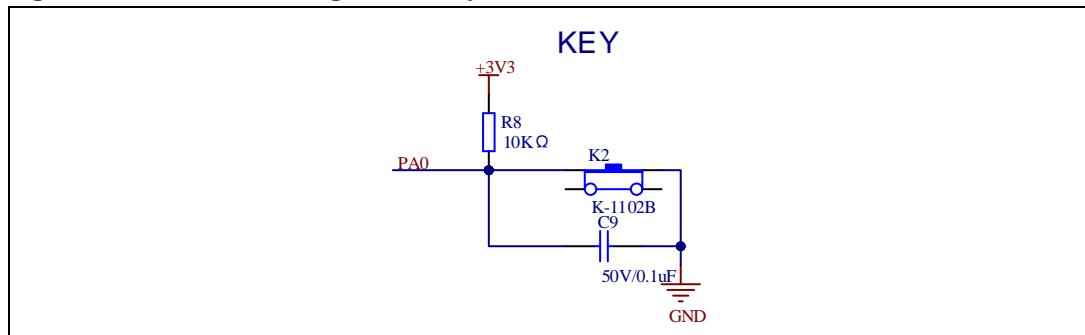
4.5. ADC

Figure 4-5. Schematic diagram of ADC
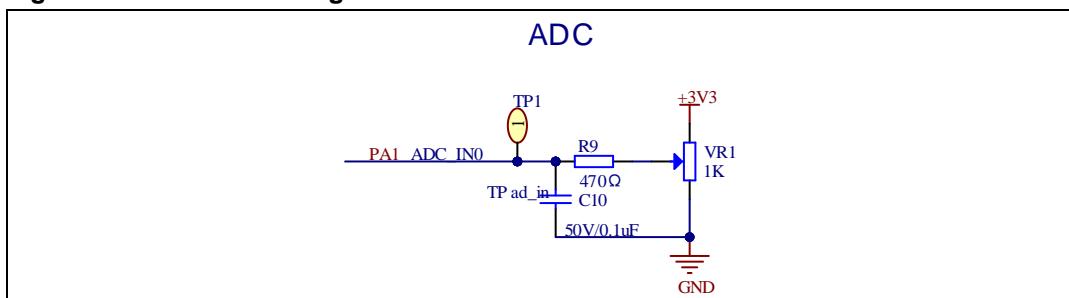
4.6. USART

Figure 4-6. Schematic diagram of USART
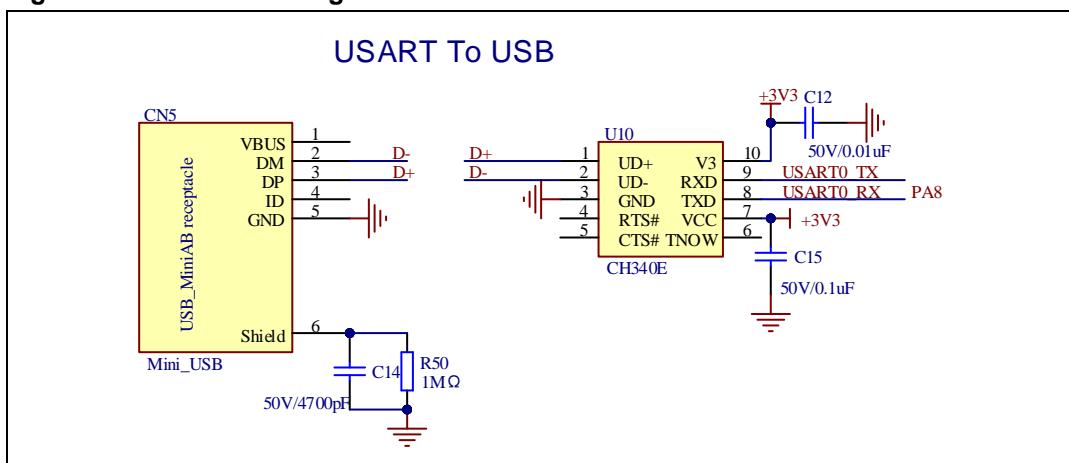
4.7. I2C

Figure 4-7. Schematic diagram of I2C
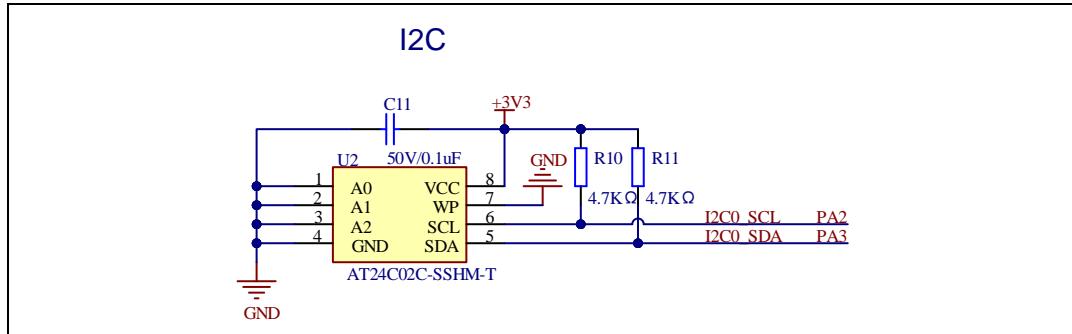
4.8. IFRP

Figure 4-8. Schematic diagram of IFRP
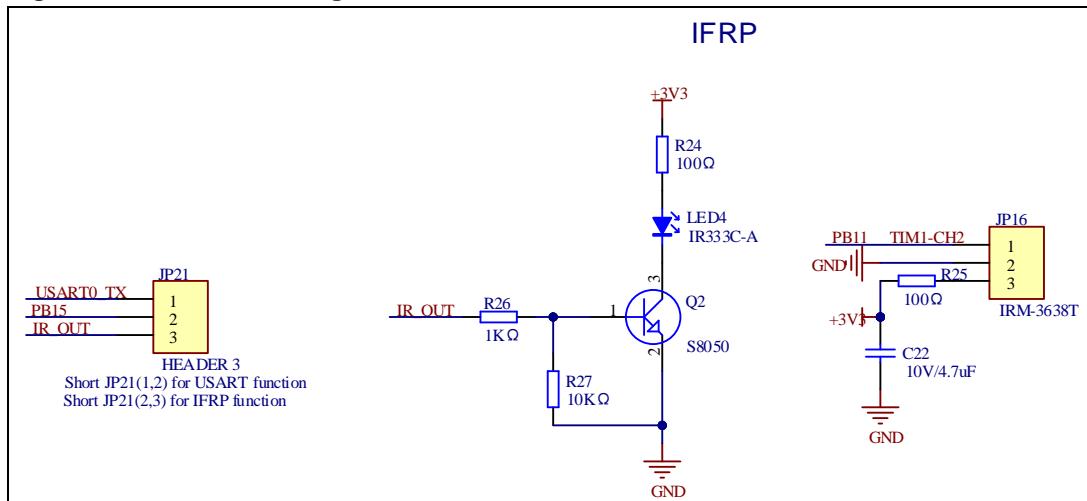
4.9. SPI_LCD

Figure 4-9. Schematic diagram of SPI_LCD
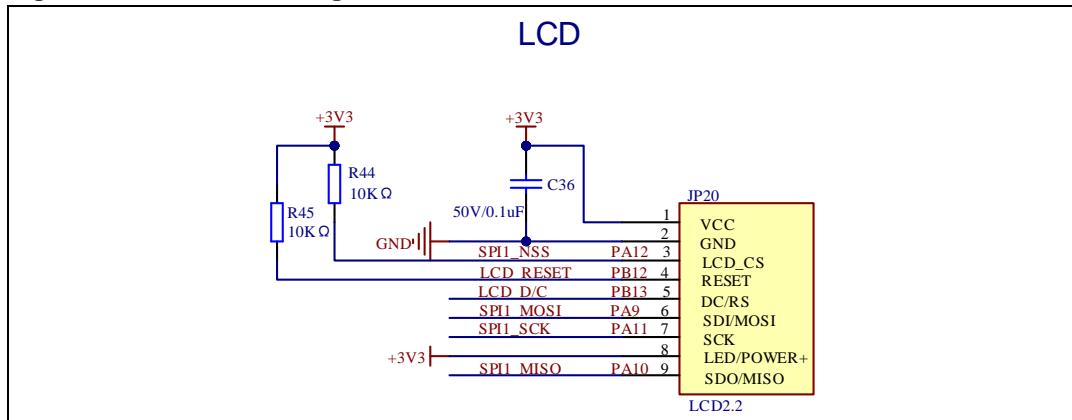
4.10. QSPI_Flash

Figure 4-10. Schematic diagram of QSPI_FLASH
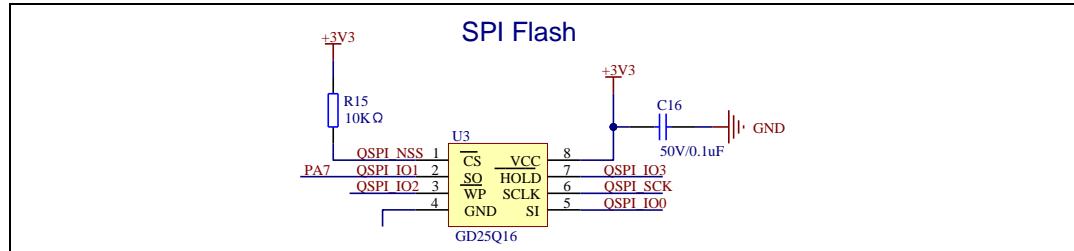
4.11. Extension

Figure 4-11. Schematic diagram of Extension
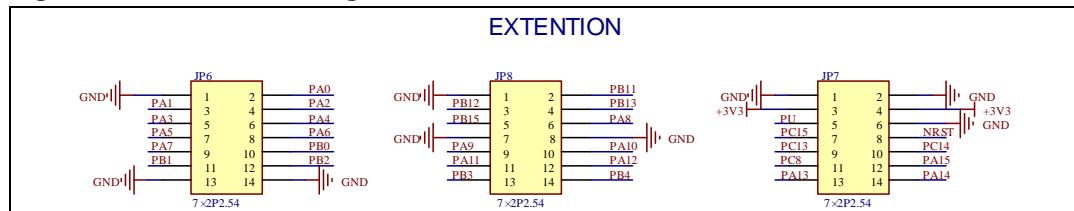
4.12. GD-Link

Figure 4-12. Schematic diagram of GD-Link
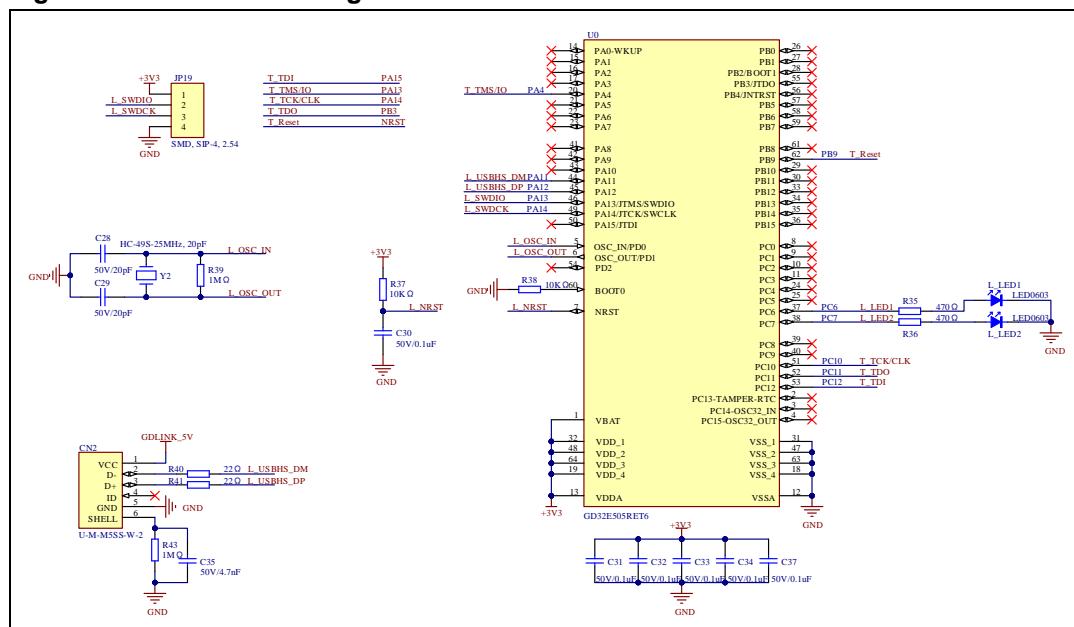
4.13. MCU

Figure 4-13. Schematic diagram of MCU
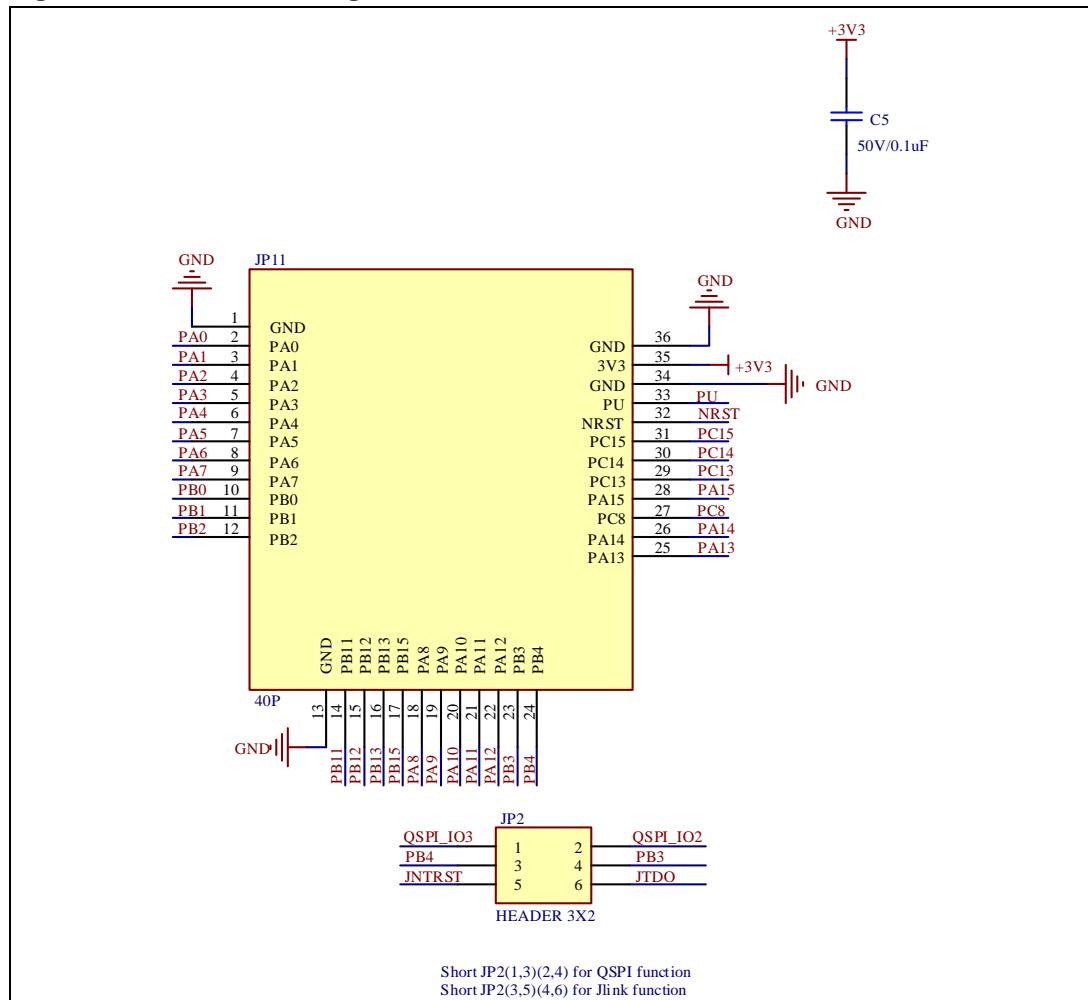
5. Routine use guide

5.1. GPIO_Running_LED

5.1.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use GPIO control the LED.
- Learn to use SysTick to generate 1ms delay.

GD32VW553H-EVAL board has two keys and three LEDs, the LEDs are controlled by GPIO.

This demo will show how to light the LEDs.

5.1.2. DEMO running result

Download the program < 01_GPIO_Running_LED > to the EVAL board, three LEDs can light cycles.

5.2. GPIO_Key_Polling_mode

5.2.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use GPIO control the LED and the Key.
- Learn to use SysTick to generate 1ms delay.

GD32VW553H-EVAL board has two keys and three LEDs. The two keys are Reset key, Tamper/Wakeup key. The LEDs are controlled by GPIO.

This demo will show how to use the Tamper/Wakeup to control the LED2. When press down the Tamper/Wakeup, it will check the input value of the IO port. If the value is 0 and will wait for 100ms. Check the input value of the IO port again. If the value still is 0, it indicates that the button is pressed successfully and toggle LED2.

5.2.2. DEMO running result

Download the program < 02_GPIO_Key_Polling_mode > to the EVAL board, press down the Tamper/Wakeup, LED2 will be turned on. Press down the Tamper/Wakeup again, LED2 will be turned off.

5.3. EXTI_Key_Interrupt_mode

5.3.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use GPIO control the LED and the KEY
- Learn to use EXTI to generate external interrupt

GD32VW553H-EVAL board has two keys and three LEDs. The two keys are Reset key, Tamper/Wakeup key. The LEDs are controlled by GPIO.

This demo will show how to use the EXTI interrupt line to control the LED2. When press down the Tamper/Wakeup, it will produce an interrupt. In the interrupt service function, the demo will toggle LED2.

5.3.2. DEMO running result

Download the program < 03_EXTI_Key_Interrupt_mode > to the EVAL board, LED2 is turned on and off for test. When press down the Tamper/Wakeup, LED2 will be turned on. Press down the Tamper/Wakeup again, LED2 will be turned off.

5.4. USART_Printf

5.4.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use GPIO control the LED
- Learn to user USART to transfer data

5.4.2. DEMO running result

Download the program < 04_USART_Printf > to the EVAL board, connect serial cable to USART. Firstly, all the LEDs flash 2 times for test. Then, this implementation outputs "please press the Tamper key" on the HyperTerminal using USART. Press the Tamper key, the serial port will output "USART printf example".

The output information via the HyperTerminal is as following:
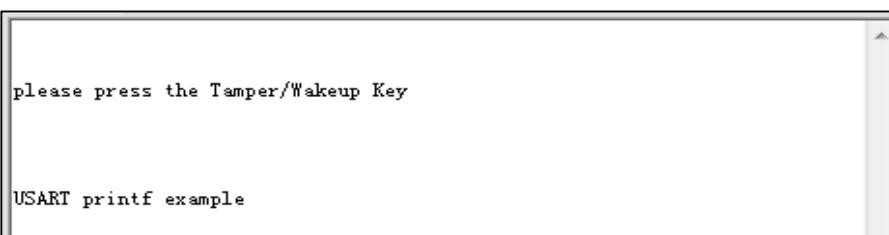

```
please press the Tamper/Wakeup Key



USART printf example
```

5.5. USART_Echo_Interrupt_mode

5.5.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

■ Learn to use the USART transmit and receive interrupts to communicate with the HyperTerminal

5.5.2. DEMO running result

Download the program <05_USART_Echo_Interrupt_mode> to the EVAL board, connect serial cable to USART. Firstly, all the LEDs are turned on and off for test. Then, the USART0 sends the tx_buffer array (from 0x00 to 0xFF) to the hyperterminal and waits for receiving data from the hyperterminal that you must send. The string that you have sent is stored in the rx_buffer array. The receive buffer have a BUFFER_SIZE bytes as maximum. After that, compare tx_buffer with rx_buffer. If tx_buffer is same with rx_buffer, LED1, LED2 are turn on, and LED3 is turn off. Otherwise, LED3 is turn on and LED1, LED2 are turn off.

```
00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 10 11 12 13 14 15 16 17 18 19 1A
1B 1C 1D 1E 1F 20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F 30 31 32 33 34 35
36 37 38 39 3A 3B 3C 3D 3E 3F 40 41 42 43 44 45 46 47 48 49 4A 4B 4C 4D 4E 4F 50
51 52 53 54 55 56 57 58 59 5A 5B 5C 5D 5E 5F 60 61 62 63 64 65 66 67 68 69 6A 6B
6C 6D 6E 6F 70 71 72 73 74 75 76 77 78 79 7A 7B 7C 7D 7E 7F 80 81 82 83 84 85 86
87 88 89 8A 8B 8C 8D 8E 8F 90 91 92 93 94 95 96 97 98 99 9A 9B 9C 9D 9E 9F A0 A1
A2 A3 A4 A5 A6 A7 A8 A9 AA AB AC AD AE AF B0 B1 B2 B3 B4 B5 B6 B7 B8 B9 BA BB BC
BD BE BF C0 C1 C2 C3 C4 C5 C6 C7 C8 C9 CA CB CC CD CE CF D0 D1 D2 D3 D4 D5 D6 D7
D8 D9 DA DB DC DD DE DF E0 E1 E2 E3 E4 E5 E6 E7 E8 E9 EA EB EC ED EE EF F0 F1 F2
F3 F4 F5 F6 F7 F8 F9 FA FB FC FD FE FF
```

5.6. USART_DMA

5.6.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

■ Learn to use the USART transmit and receive data using DMA

5.6.2. DEMO running result

Download the program <06_USART_DMA> to the EVAL board, connect serial cable to USART. Firstly, all the LEDs are flash 3 times for test. Then, the USART sends the tx_buffer array to the hyperterminal and waits for receiving data from the hyperterminal that you must send. The string that you have sent is stored in the rx_buffer array. The receive buffer have a BUFFER_SIZE bytes as maximum. After that, compare tx_buffer with rx_buffer. If tx_buffer is same with rx_buffer, LED1, LED2 are turn on, and LED3 is turn off. Otherwise, LED3 is

turn on and LED1, LED2 are turn off.

The output information via the HyperTerminal is as following:

```
00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 10 11 12 13 14 15 16 17
18 19 1A 1B 1C 1D 1E 1F 20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F
30 31 32 33 34 35 36 37 38 39 3A 3B 3C 3D 3E 3F 40 41 42 43 44 45 46 47
48 49 4A 4B 4C 4D 4E 4F 50 51 52 53 54 55 56 57 58 59 5A 5B 5C 5D 5E 5F
60 61 62 63 64 65 66 67 68 69 6A 6B 6C 6D 6E 6F 70 71 72 73 74 75 76 77
78 79 7A 7B 7C 7D 7E 7F 80 81 82 83 84 85 86 87 88 89 8A 8B 8C 8D 8E 8F
90 91 92 93 94 95 96 97 98 99 9A 9B 9C 9D 9E 9F A0 A1 A2 A3 A4 A5 A6 A7
A8 A9 AA AB AC AD AE AF B0 B1 B2 B3 B4 B5 B6 B7 B8 B9 BA BB BC BD BE BF
C0 C1 C2 C3 C4 C5 C6 C7 C8 C9 CA CB CC CD CE CF D0 D1 D2 D3 D4 D5 D6 D7
D8 D9 DA DB DC DD DE DF E0 E1 E2 E3 E4 E5 E6 E7 E8 E9 EA EB EC ED EE EF
F0 F1 F2 F3 F4 F5 F6 F7 F8 F9 FA FB FC FD FE FF
```

5.7.   ADC_conversion_triggered_by_timer

5.7.1.   DEMO purpose

This demo includes the following functions of GD32 MCU:

■   Learn to use the ADC to convert analog signal to digital data
■   Learn to use TIMER to generate a channel compare event

TIMER0 CH0 event triggers ADC conversion, the value corresponds to the ADC analog input, and changes with it. The converted data are moved to SRAM through DMA continuously, and printed by USART.

5.7.2.   DEMO running result

Download the program <07_ADC_conversion_triggered_by_timer> to the GD32VW553H-EVAL board and connect serial cable to USART, adjust the adjustable potentiometer knob to change the analog input. The ADC, which is triggered by TIMER0 CH0 event, will convert the analog input, and you will see the result by USART.

5.8.   I2C_EEPROM

5.8.1.   DEMO purpose

This demo includes the following functions of GD32 MCU:

■   Learn to use the master transmitting mode of I2C module
■   Learn to use the master receiving mode of I2C module
■   Learn to read and write the EEPROM with I2C interface

5.8.1. DEMO running result

Download the program <08_I2C_EEPROM> to the EVAL board and run. Connect serial cable to USART, open the HyperTerminal to show the print message.

Firstly, the data of 256 bytes will be written to the EEPROM from the address 0x00 and printed by the serial port. Then, reading the EEPROM from address 0x00 for 256 bytes and the result will be printed. Finally, compare the data that were written to the EEPROM and the data that were read from the EEPROM. If they are the same, the serial port will output "I2C-AT24C02 test passed!" and the LEDs flashing, otherwise the serial port will output "Err:data read and write aren't matching." and all the LEDs light.

The output information via the serial port is as following.

```
I2C-24C02 configured....

The I2C is hardware interface
The speed is 400000
AT24C02 writing...
0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x0A 0x0B 0x0C 0x0D 0x0E 0x0F
0x10 0x11 0x12 0x13 0x14 0x15 0x16 0x17 0x18 0x19 0x1A 0x1B 0x1C 0x1D 0x1E 0x1F
0x20 0x21 0x22 0x23 0x24 0x25 0x26 0x27 0x28 0x29 0x2A 0x2B 0x2C 0x2D 0x2E 0x2F
0x30 0x31 0x32 0x33 0x34 0x35 0x36 0x37 0x38 0x39 0x3A 0x3B 0x3C 0x3D 0x3E 0x3F
0x40 0x41 0x42 0x43 0x44 0x45 0x46 0x47 0x48 0x49 0x4A 0x4B 0x4C 0x4D 0x4E 0x4F
0x50 0x51 0x52 0x53 0x54 0x55 0x56 0x57 0x58 0x59 0x5A 0x5B 0x5C 0x5D 0x5E 0x5F
0x60 0x61 0x62 0x63 0x64 0x65 0x66 0x67 0x68 0x69 0x6A 0x6B 0x6C 0x6D 0x6E 0x6F
0x70 0x71 0x72 0x73 0x74 0x75 0x76 0x77 0x78 0x79 0x7A 0x7B 0x7C 0x7D 0x7E 0x7F
0x80 0x81 0x82 0x83 0x84 0x85 0x86 0x87 0x88 0x89 0x8A 0x8B 0x8C 0x8D 0x8E 0x8F
0x90 0x91 0x92 0x93 0x94 0x95 0x96 0x97 0x98 0x99 0x9A 0x9B 0x9C 0x9D 0x9E 0x9F
0xA0 0xA1 0xA2 0xA3 0xA4 0xA5 0xA6 0xA7 0xA8 0xA9 0xAA 0xAB 0xAC 0xAD 0xAE 0xAF
0xB0 0xB1 0xB2 0xB3 0xB4 0xB5 0xB6 0xB7 0xB8 0xB9 0xBA 0xBB 0xBC 0xBD 0xBE 0xBF
0xC0 0xC1 0xC2 0xC3 0xC4 0xC5 0xC6 0xC7 0xC8 0xC9 0xCA 0xCB 0xCC 0xCD 0xCE 0xCF
0xD0 0xD1 0xD2 0xD3 0xD4 0xD5 0xD6 0xD7 0xD8 0xD9 0xDA 0xDB 0xDC 0xDD 0xDE 0xDF
0xE0 0xE1 0xE2 0xE3 0xE4 0xE5 0xE6 0xE7 0xE8 0xE9 0xEA 0xEB 0xEC 0xED 0xEE 0xEF
0xF0 0xF1 0xF2 0xF3 0xF4 0xF5 0xF6 0xF7 0xF8 0xF9 0xFA 0xFB 0xFC 0xFD 0xFE 0xFF
AT24C02 reading...
0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x0A 0x0B 0x0C 0x0D 0x0E 0x0F
0x10 0x11 0x12 0x13 0x14 0x15 0x16 0x17 0x18 0x19 0x1A 0x1B 0x1C 0x1D 0x1E 0x1F
0x20 0x21 0x22 0x23 0x24 0x25 0x26 0x27 0x28 0x29 0x2A 0x2B 0x2C 0x2D 0x2E 0x2F
0x30 0x31 0x32 0x33 0x34 0x35 0x36 0x37 0x38 0x39 0x3A 0x3B 0x3C 0x3D 0x3E 0x3F
0x40 0x41 0x42 0x43 0x44 0x45 0x46 0x47 0x48 0x49 0x4A 0x4B 0x4C 0x4D 0x4E 0x4F
0x50 0x51 0x52 0x53 0x54 0x55 0x56 0x57 0x58 0x59 0x5A 0x5B 0x5C 0x5D 0x5E 0x5F
0x60 0x61 0x62 0x63 0x64 0x65 0x66 0x67 0x68 0x69 0x6A 0x6B 0x6C 0x6D 0x6E 0x6F
0x70 0x71 0x72 0x73 0x74 0x75 0x76 0x77 0x78 0x79 0x7A 0x7B 0x7C 0x7D 0x7E 0x7F
0x80 0x81 0x82 0x83 0x84 0x85 0x86 0x87 0x88 0x89 0x8A 0x8B 0x8C 0x8D 0x8E 0x8F
0x90 0x91 0x92 0x93 0x94 0x95 0x96 0x97 0x98 0x99 0x9A 0x9B 0x9C 0x9D 0x9E 0x9F
0xA0 0xA1 0xA2 0xA3 0xA4 0xA5 0xA6 0xA7 0xA8 0xA9 0xAA 0xAB 0xAC 0xAD 0xAE 0xAF
0xB0 0xB1 0xB2 0xB3 0xB4 0xB5 0xB6 0xB7 0xB8 0xB9 0xBA 0xBB 0xBC 0xBD 0xBE 0xBF
0xC0 0xC1 0xC2 0xC3 0xC4 0xC5 0xC6 0xC7 0xC8 0xC9 0xCA 0xCB 0xCC 0xCD 0xCE 0xCF
0xD0 0xD1 0xD2 0xD3 0xD4 0xD5 0xD6 0xD7 0xD8 0xD9 0xDA 0xDB 0xDC 0xDD 0xDE 0xDF
0xE0 0xE1 0xE2 0xE3 0xE4 0xE5 0xE6 0xE7 0xE8 0xE9 0xEA 0xEB 0xEC 0xED 0xEE 0xEF
0xF0 0xF1 0xF2 0xF3 0xF4 0xF5 0xF6 0xF7 0xF8 0xF9 0xFA 0xFB 0xFC 0xFD 0xFE 0xFF
I2C-AT24C02 test passed!
```

5.9. SPI_LCD

5.9.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

■ Learn how to use SPI to drive TFT LCD screen and display.

GD32VW553H-EVAL board has a TFT LCD screen which supports SPI interface. In this demo, tests of font, number, draw and color are displayed on the LCD screen respectively.

5.9.2. DEMO running result

LCD is controlled by SPI module. Download the program <09_SPI_LCD > to the EVAL board. All the LEDs are turned on and then turned off for test. After that, the LCD screen on the board will display the GUI tests in infinite loop.

5.10. TRNG_Get_Random

5.10.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

■ Learn to use TRNG generate the random number
■ Learn to communicate with PC by USART

5.10.2. DEMO running result

Download the program <10_TRNG_Get_Random> to the EVAL board and run. Connect serial cable to USART, open the serial terminal tool supporting hex format communication. When the program is running, the serial terminal tool will display the initial information. User can use the serial terminal tool to input the minimum and maximum values (for example, the minimum value is 0x011, the maximum value is 0x33), then application will generate random number in the input range and display it by the serial terminal tool.

Information via a serial port output as following:

```
/=============Gigadevice TRNG test=============/
TRNG init ok
Please input min num (hex format, the range is 0~0xFF):
The input min num is 0x11
Please input max num hex format, the range is 0~0xFF):
The input max num is 0x33
Generate random num1 is 0x26
Generate random num2 is 0x33
Please input min num (hex format, the range is 0~0xFF):
```

5.11. CAU

5.11.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn DES, Triple-DES and AES algorithm;
- Learn Electronic codebook (ECB) mode, Cipher block chaining (CBC) mode, Counter (CTR) mode, Galois / counter (GCM) mode, combined cipher machine (CCM) mode, Cipher Feedback (CFB) mode, and Output Feedback (OFB) mode;
- Learn to use CAU to encrypt and decrypt;
- Learn to communicate with PC by USART.

5.11.2. DEMO running result

Download the program <11_CAU> to the EVAL board and run. When the program is running, the serial terminal tool will display the information, as shown in the following figure. Plaintext data value, the encryption algorithm, and the mode can be selected are shown. After the user setting the algorithm and mode according to the serial output information indicating, serial port will print out selected algorithm and mode, as shown below.

```
Plain data :
0x6B 0xC1 0xBE 0xE2 0x2E 0x40 0x9F 0x96 0xE9 0x3D 0x7E 0x11 0x73 0x93 0x17 0x2A    [Block 0]
0xAE 0x2D 0x8A 0x57 0x1E 0x03 0xAC 0x9C 0x9E 0xB7 0x6F 0xAC 0x45 0xAF 0x8E 0x51    [Block 1]
0x30 0xC8 0x1C 0x46 0xA3 0x5C 0xE4 0x11 0xE5 0xFB 0xC1 0x19 0x1A 0x0A 0x52 0xEF    [Block 2]
0xF6 0x9F 0x24 0x45 0xDF 0x4F 0x9B 0x17 0xAD 0x2B 0x41 0x7B 0xE6 0x6C 0x37 0x10    [Block 3]
========Choose CAU algorithm======
1: DES algorithm
2: TDES algorithm
3: AES algorithm


You choose to use DES algorithm
========Choose CAU mode======
1: ECB mode
2: CBC mode
3: CTR mode only when choose AES algorithm
4: GCM mode only when choose AES algorithm
5: CCM mode only when choose AES algorithm
6: CFB mode only when choose AES algorithm
7: OFB mode only when choose AES algorithm


You choose to use ECB mode
```

After selection, the program starts encryption and decryption operations, the results are printed through the serial port.

```
 Encrypted data with DES  Mode  ECB :

0x6E 0xDF 0xD1 0xB7 0xA0 0x01 0xCD 0x17 0xCD 0xC5 0x7F 0xF7 0x9C 0xF8 0x72 0xD0    [Block 0]
0x11 0x97 0xA6 0xD2 0x13 0x59 0x4F 0x7A 0x3D 0x7C 0x7C 0xEC 0xBC 0xDD 0xD2 0x20    [Block 1]
0x3A 0x75 0x8B 0x06 0x75 0x2E 0x18 0x0D 0x55 0x0F 0xDD 0x57 0x5A 0xF1 0x3B 0x94    [Block 2]
0x18 0x3D 0x4D 0xA1 0x1E 0x14 0x75 0x6B 0x0F 0xD9 0xD9 0x64 0x16 0xA0 0x60 0x14    [Block 3]

 Decrypted data with DES  Mode  ECB :

0x6B 0xC1 0xBE 0xE2 0x2E 0x40 0x9F 0x96 0xE9 0x3D 0x7E 0x11 0x73 0x93 0x17 0x2A    [Block 0]
0xAE 0x2D 0x8A 0x57 0x1E 0x03 0xAC 0x9C 0x9E 0xB7 0x6F 0xAC 0x45 0xAF 0x8E 0x51    [Block 1]
0x30 0xC8 0x1C 0x46 0xA3 0x5C 0xE4 0x11 0xE5 0xFB 0xC1 0x19 0x1A 0x0A 0x52 0xEF    [Block 2]
0xF6 0x9F 0x24 0x45 0xDF 0x4F 0x9B 0x17 0xAD 0x2B 0x41 0x7B 0xE6 0x6C 0x37 0x10    [Block 3]

 Example restarted...
```

And then restart for users to select a different algorithm and mode to repeat demo, as shown below.

```
Plain data :
0x6B 0xC1 0xBE 0xE2 0x2E 0x40 0x9F 0x96 0xE9 0x3D 0x7E 0x11 0x73 0x93 0x17 0x2A    [Block 0]
0xAE 0x2D 0x8A 0x57 0x1E 0x03 0xAC 0x9C 0x9E 0xB7 0x6F 0xAC 0x45 0xAF 0x8E 0x51    [Block 1]
0x30 0xC8 0x1C 0x46 0xA3 0x5C 0xE4 0x11 0xE5 0xFB 0xC1 0x19 0x1A 0x0A 0x52 0xEF    [Block 2]
0xF6 0x9F 0x24 0x45 0xDF 0x4F 0x9B 0x17 0xAD 0x2B 0x41 0x7B 0xE6 0x6C 0x37 0x10    [Block 3]
========Choose CAU algorithm======
1: DES algorithm
2: TDES algorithm
3: AES algorithm
```

5.12. HAU

5.12.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn SHA-1, SHA-224, SHA-256 and MD5 algorithm;
- Learn HASH mode and HMAC (keyed-hash message authentication code) mode;
- Learn to use HAU to calculate digest for the input message;
- Learn to communicate with PC by USART.

5.12.2. DEMO running result

Download the program <12_HAU> to the EVAL board and run. When the program is running, the serial terminal tool will display the information, as shown in the following figure. After the user setting the algorithm and mode according to the serial output information indicating, serial port will print out selected algorithm and mode, as shown below.

```
message to be hashed:

The hash processor is a fully compliant implementation of the secure
hash algorithm (SHA-1), the MD5 (message-digest algorithm 5) hash
algorithm and the HMAC (keyed-hash message authentication code)
algorithm suitable for a variety of applications.========Choose HAU
algorithm=======
1: SHA1 algorithm
2: SHA224 algorithm
3: SHA256 algorithm
4: MD5 algorithm


You choose to use SHA1 algorithm
========Choose HAU mode=======
1: HASH mode
2: HMAC mode


You choose to use HASH mode
```

After selection, the program starts digest calculation, the results are printed through the serial port. And then restart for users to select a different algorithm and mode to repeat demo, as shown below.

```
message digest with SHA-1  Mode  HASH (160 bits):

0x08 0x54 0x77 0x5E
0xC2 0xA1 0x0C 0x7F
0x4B 0x80 0x37 0xD9
0xE7 0x7C 0xA7 0x30
0xF0 0x5D 0xFA 0x2E

 Example restarted...

message to be hashed:

The hash processor is a fully compliant implementation of the secure
hash algorithm (SHA-1), the MD5 (message-digest algorithm 5) hash
algorithm and the HMAC (keyed-hash message authentication code)
algorithm suitable for a variety of applications.========Choose HAU
algorithm=======
1: SHA1 algorithm
2: SHA224 algorithm
3: SHA256 algorithm
4: MD5 algorithm
```

5.13. PKCAU_Modular_Addition_Interrupt

5.13.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

■ Learn modular addition algorithm by interrupt

5.13.2. DEMO running result

Download the program < 13_PKCAU_Modular_Addition_Interrupt > to the EVAL board. After system start-up, it is initialized first, then, perform modular addition computation, when the computation is completed, an interrupt will be generated, finally, read results from specific PKCAU RAM address and compare with the expected result, if the result is the same with the expected one, LED1 will on, or else, LED2 is on.

5.14. RCU_Clock_Out

5.14.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

■ Learn to use GPIO control the LED

■ Learn to use the clock output function of RCU

■ Learn to communicate with PC by USART

5.14.2. DEMO running result

Download the program <14_RCU_Clock_Out> to the EVAL board and run. Connect serial cable to USART, open the HyperTerminal. When the program is running, HyperTerminal will display the initial information. Then user can choose the type of the output clock by pressing the TAMPER key. After pressing, the corresponding LED will be turned on and HyperTerminal will display which mode be selected. The frequency of the output clock can be observed through the oscilloscope by PA8 and PB11 pin.

Information via a serial port output as following:

```
/========= Gigadevice Clock Output Demo =========/
press tamper/wakeup key to select clock output source
CK_OUT0: system clock clock, DIV:5
CK_OUT0: IRC16M, DIV:1
CK_OUT0: HXTAL, DIV:2
CK_OUT0: LXTAL
CK_OUT1: system clock, DIV:5
CK_OUT1: PLLDIG, DIV:4
```

5.15. PMU_Sleep_Wakeup

5.15.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

■ Learn to use the USART receive interrupt to wake up the PMU from sleep mode

5.15.2. DEMO running result

Download the program <15_PMU_Sleep_Wakeup> to the EVAL board, connect serial cable to USART. After power-on, all the LEDs are off. The mcu will enter sleep mode and the software stop running. When the USART0 receives a byte of data from the HyperTerminal, the mcu will wake up from a receive interrupt. And all the LEDs will flash together.

5.16. RTC_Calendar

5.16.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

■ Learn to use RTC module to implement calendar function

■ Learn to use USART module to implement time display

5.16.2. DEMO running result

Download the program <16_RTC_Calendar> to the EVAL board and run. Connect serial cable to USART, open the HyperTerminal. After start-up, the program will ask to set the time on the HyperTerminal. The calendar will be displayed on the HyperTerminal.

```
****************** RTC calendar demo ******************

======Configure RTC Time======

please input hour:

10

please input minute:


12

please input second:


14


** RTC time configuration success! **

Current time: 10:12:14
```

5.17. IFRP

5.17.1. DEMO purpose

This Demo includes the following functions of GD32 MCU:

- Learn to use general timer output PWM wave
- Learn to use general timer generated update interrupt
- Learn to use general timer capture interrupt
- Learn to use general timer TIMER15 and TIMER16 implement Infrared function

5.17.2. DEMO running result

Download the program <17_IFRP > to the EVAL board and run. When the program is running, if the infrared receiver received data is correct, LED1, LED2, LED3 light in turn, otherwise LED1, LED2, LED3 toggle together.

5.18. TIMER_Breath_LED

5.18.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use TIMER output PWM wave
- Learn to update channel value

5.18.2. DEMO running result

Use the DuPont line to connect the TIMER0_CH0 (PA8) and LED1 (PA4), and then download the program <18_TIMER_Breath_LED> to the board and run. PA8 should not be reused by other peripherals.

When the program is running, you can see LED1 lighting from dark to bright gradually and then gradually darken, ad infinitum, just like breathing as rhythm.

5.19. QSPI_Flash

5.19.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use QSPI to write and read spi flash data

5.19.2. DEMO running result

Download the program <19_QSPI_Flash> to the EVAL board and run. Connect serial cable to USART, open the HyperTerminal to show the print message. If the data read from the flash is the same with the data write to the flash, the USART will print "SPI FLASH WRITE AND READ TEST SUCCESS!". Otherwise, the USART will print "SPI FLASH WRITE AND READ TEST ERROR!".

The output information via the serial port is as following.

6. Wi-Fi & BLE

For Wi-Fi & BLE signaling and non-signaling RF tests, we usually recommend to use another GD development board called GD32VW553K-START.(Complete START board resource with test firmware and test tools can be downloaded from GD MCU website.) For the convenience, EVAL board also support the above functions.

The MD1 module on the EVAL board supports PCB antenna or an external antenna via a test connector. The antenna is switched by soldering to change the direction of a zero-ohm resistor of 0201 package next to the test connector. The default direction of this resistor is connected to the PCB antenna, if you need to connect an external antenna or need to perform RF conductive test, you need to switch the direction of this resistor.

Test firmwares can be fetched via GD32VW55x_Demo_Suites\ GD32W553H_EVAL_Demo_Suites \ Docs\ WirelessBin, and contain rf test firmware and signaling firmware, which can be burned to the main chip via GD-Link on the EVAL board.

Test tools are same as START board. Test method can refer to the official website of the following document, please refer to the documentations on the official website for test methods.

《 AN149 Test Guidelines for RF Performance and Transceiver Power Consumption ofGD32VW553》

This test manual is mainly used to give instructions to clients to test various RF parameters and corresponding power consumption of transmitting and receiving of the WiFi and BLE development board corresponding to the chips of GD32VW553 series in non-signaling mode.

《 AN150 Test Guidelines for Throughput and Scenario-based Power Consumption of GD32VW553》

This test manual is mainly used to give instructions to clients to test transmitting and receiving throughput performance and various scenario-based power consumption performance of the Wi-Fi development board corresponding to the chips of GD32VW553 series in signaling mode.

《AN146 GD32VW553 Certification Test Guidelines》

This test guideline is mainly used to give guidance to clients to test RF certification and regulatory performance of chips of GD32VW553 series. The certification regulations mainly refer to FCC, CE, and SRRC.

7. Revision history

Table 7-1. Revision history

Revision No.	Description	Date
1.0	Initial Release	Aug.10, 2023
1.1	Add introduction of Wi-Fi and BLE content	Jan. 17, 2024
1.2	Modify the schematic diagram, upgrade the onboard GD-Link chip to GD32E505RET6	Mar. 11, 2024

Important Notice

This document is the property of GigaDevice Semiconductor Inc. and its subsidiaries (the "Company"). This document, including any product of the Company described in this document (the "Product"), is owned by the Company under the intellectual property laws and treaties of the People's Republic of China and other jurisdictions worldwide. The Company reserves all rights under such laws and treaties and does not grant any license under its patents, copyrights, trademarks, or other intellectual property rights. The names and brands of third party referred thereto (if any) are the property of their respective owner and referred to for identification purposes only.

The Company makes no warranty of any kind, express or implied, with regard to this document or any Product, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. The Company does not assume any liability arising out of the application or use of any Product described in this document. Any information provided in this document is provided only for reference purposes. It is the responsibility of the user of this document to properly design, program, and test the functionality and safety of any application made of this information and any resulting product. Except for customized products which has been expressly identified in the applicable agreement, the Products are designed, developed, and/or manufactured for ordinary business, industrial, personal, and/or household applications only. The Products are not designed, intended, or authorized for use as components in systems designed or intended for the operation of weapons, weapons systems, nuclear installations, atomic energy control instruments, combustion control instruments, airplane or spaceship instruments, transportation instruments, traffic signal instruments, life-support devices or systems, other medical devices or systems (including resuscitation equipment and surgical implants), pollution control or hazardous substances management, or other uses where the failure of the device or Product could cause personal injury, death, property or environmental damage ("Unintended Uses"). Customers shall take any and all actions to ensure using and selling the Products in accordance with the applicable laws and regulations. The Company is not liable, in whole or in part, and customers shall and hereby do release the Company as well as it's suppliers and/or distributors from any claim, damage, or other liability arising from or related to all Unintended Uses of the Products. Customers shall indemnify and hold the Company as well as it's suppliers and/or distributors harmless from and against all claims, costs, damages, and other liabilities, including claims for personal injury or death, arising from or related to any Unintended Uses of the Products.

Information in this document is provided solely in connection with the Products. The Company reserves the right to make changes, corrections, modifications or improvements to this document and Products and services described herein at any time, without notice.