GigaDevice Semiconductor Inc.

GD32VF103V-EVAL
User Guide
V1.1

Table of Contents

Table of Contents 1

List of Figures 4

List of Tables 5

1. Summary 6

2. Function Pin Assign 6

3. Getting started 7

4. Hardware layout overview 8

4.1. Power supply 8

4.2. Boot option 8

4.3. LED 9

4.4. KEY 9

4.5. USART 10

4.6. ADC 10

4.7. DAC 10

4.8. I2S 11

4.9. I2C 11

4.10. SPI 11

4.11. CAN 12

4.12. LCD 12

4.13. USBFS 13

4.14. GD-Link 14

4.15. Extension 14

4.16. MCU 15

5. Routine use guide 15

5.1. GPIO_Running_Led 15

5.1.1. DEMO purpose 15

5.1.2. DEMO running result 16

5.2. GPIO_Key_Polling_mode 16

5.2.1. DEMO purpose 16

5.2.2. DEMO running result 16

5.3. EXTI_Key_Interrupt_mode 16
5.3.1. DEMO purpose 16
5.3.2. DEMO running result 17
5.4. USART_Printf 17
5.4.1. DEMO purpose 17
5.4.2. DEMO running result 17
5.5. USART_Echo_Interrupt_mode 17
5.5.1. DEMO purpose 17
5.5.2. DEMO running result 17
5.6. USART_DMA 18
5.6.1. DEMO purpose 18
5.6.2. DEMO running result 18
5.7. ADC_conversion_triggered_by_timer 19
5.7.1. DEMO purpose 19
5.7.2. DEMO running result 19
5.8. ADC0_ADC1_Follow_up_mode 19
5.8.1. DEMO purpose 19
5.8.2. DEMO running result 19
5.9. ADC0_ADC1_Regular_Parallel_mode 20
5.9.1. DEMO purpose 20
5.9.2. DEMO running result 20
5.10. DAC_Output_Voltage_Value 21
5.10.1. DEMO purpose 21
5.10.2. DEMO running result 22
5.11. I2C_EEPROM 22
5.11.1. DEMO purpose 22
5.11.2. DEMO running result 22
5.12. SPI_SPI_Flash 22
5.12.1. DEMO purpose 22
5.12.2. DEMO running result 23
5.13. I2S_Audio_Player 24
5.13.1. DEMO purpose 24
5.13.2. DEMO running result 24
5.14. EXMC_TouchScreen 24
5.14.1. DEMO purpose 24
5.14.2. DEMO running result 24
5.15. CAN_Network 25
5.15.1. DEMO purpose 25
5.15.2. DEMO running result 25

5.16. RCU_Clock_Out 26
5.16.1. DEMO purpose 26
5.16.2. DEMO running result 26
5.17. PMU_sleep_wakeup 26
5.17.1. DEMO purpose 26
5.17.2. DEMO running result 27
5.18. RTC_Calendar 27
5.18.1. DEMO purpose 27
5.18.2. DEMO running result 27
5.19. TIMER_Breath_LED 28
5.19.1. DEMO purpose 28
5.19.2. DEMO running result 28
5.20. USBFS_Device 29
5.20.1. CDC_ACM 29
5.20.2. MSC 29
5.21. USBFS_Host 30
5.21.1. HID 30
5.21.2. MSC 31
6. Revision history 32

List of Figures

Figure 4-1 Schematic diagram of power supply 8
Figure 4-2 Schematic diagram of boot option 8
Figure 4-3 Schematic diagram of LED function 9
Figure 4-4 Schematic diagram of Key function 9
Figure 4-5 Schematic diagram of USART function 10
Figure 4-6 Schematic diagram of ADC function 10
Figure 4-7 Schematic diagram of DAC function 10
Figure 4-8 Schematic diagram of I2S function 11
Figure 4-9 Schematic diagram of I2C function 11
Figure 4-10 Schematic diagram of SPI function 11
Figure 4-11 Schematic diagram of CAN function 12
Figure 4-12 Schematic diagram of EXMC_LCD function 12
Figure 4-13 Schematic diagram of USBFS function 13
Figure 4-14 Schematic diagram of GD-Link function 14
Figure 4-15 Schematic diagram of Extension Pin 14
Figure 4-16 Schematic diagram of MCU Pin 15

List of Tables

Table 2-1 Pin assignment 6
Table 4-1 Boot configuration 8
Table 6-1 Revision history 32

1. Summary

GD32VF103V-EVAL evaluation board uses GD32VF103VBT6 as the main controller. As a complete development platform of GD32VF103 powered by RISC-V core, the board supports full range of peripherals. It uses mini-USB interface or AC/DC adapter to supply 5V power. SWD, Reset, Boot, User button key, LED, CAN, I2C, I2S, USART, RTC, SPI, ADC, DAC, EXMC, USBFS and Extension Pins are also included.

2. Function Pin Assign

Table 2-1 Pin assignment

Function	Pin	Description
LED	PC0	LED1
	PC2	LED2
	PE0	LED3
	PE1	LED4
RESET		K1-Reset
KEY	PA0	KEY_A
	PC13	KEY_B
	PB14	KEY_C
	PC5	KEY_D
	PC4	KEY_Cet
USB	PA11	USB_DM
	PA12	USB_DP
	PA9	USB_VBUS
I2C	PB6	I2C0_SCL
	PB7	I2C0_SDA
I2S	PB12	I2S0_WS
	PB13	I2S0_CK
	PB15	I2S0_SD
	PC6	I2S0_MCK
USART0	PA9	USART0_TX
	PA10	USART0_RX
USART1	PA2	USART1_TX
	PA3	USART2_RX
SPI	PA5	SPI0_SCK
	PA6	SPI0_MISO
	PA7	SPI0_MOSI
	PE3	SPI0_CS
CAN0	PD0	CAN0_RX
	PD1	CAN0_TX
CAN1	PB5	CAN1_RX

Function	Pin	Description
	PB6	CAN1_TX
LCD	PD14	EXMC_D0
	PD15	EXMC_D1
	PD0	EXMC_D2
	PD1	EXMC_D3
	PE7	EXMC_D4
	PE8	EXMC_D5
	PE9	EXMC_D6
	PE10	EXMC_D7
	PE11	EXMC_D8
	PE12	EXMC_D9
	PE13	EXMC_D10
	PE14	EXMC_D11
	PE15	EXMC_D12
	PD8	EXMC_D13
	PD9	EXMC_D14
	PD10	EXMC_D15
	PE2	EXMC_A23
	PD4	EXMC_NOE
	PD5	EXMC_NWE
	PD7	EXMC_NE0
ADC	PC3	ADC01_IN13
DAC	PA4	DAC_OUT0
	PA5	DAC_OUT1

3. Getting started

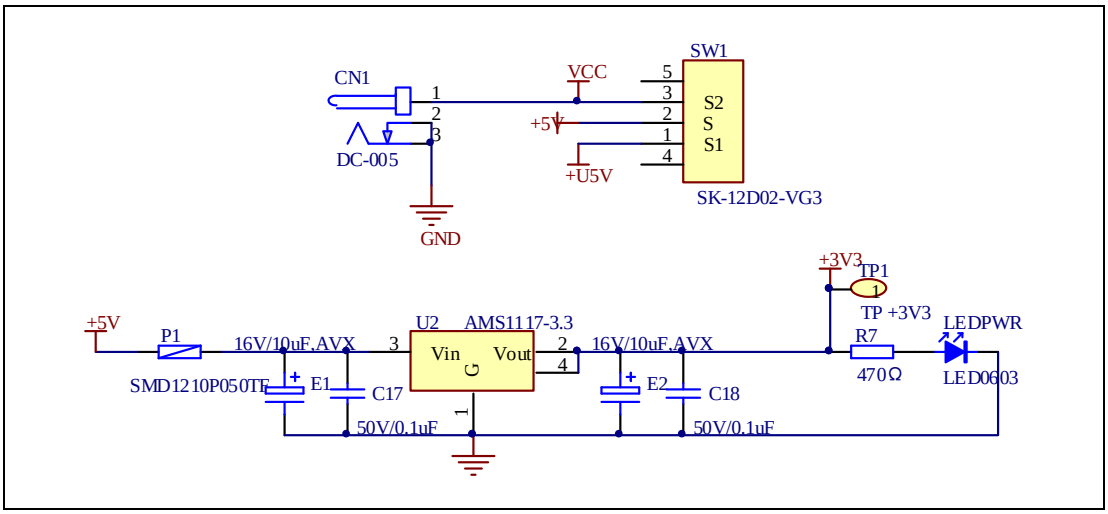
The EVAL board uses Mini USB connecter or DC-005 connector to get power DC +5V, which is the hardware system normal work voltage. A J-Link tool or GD-Link on board is necessary in order to download and debug programs. Select the correct boot mode and then power on, the LEDPWR will turn on, which indicates the power supply is ready.

The projects are created based on eclipse 4.7.2. Note that to configure the “Debug Configurations” before debug and download.

4. Hardware layout overview

4.1. Power supply

Figure 4-1 Schematic diagram of power supply



4.2. Boot option

Figure 4-2 Schematic diagram of boot option

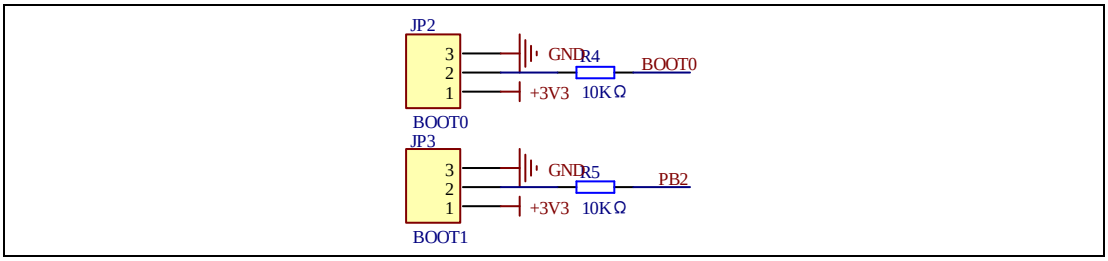
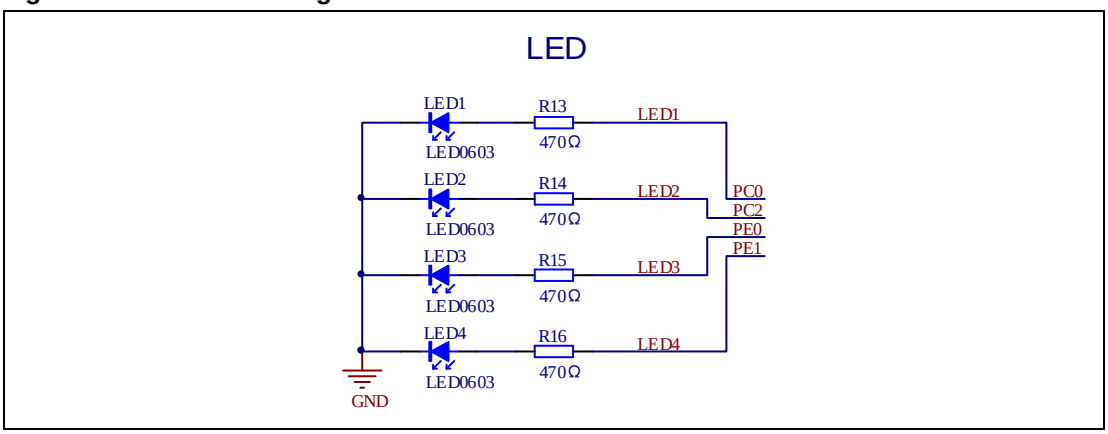


Table 4-1 Boot configuration

BOOT1	BOOT0	Boot Mode
Default	2-3	User memory
	1-2	System memory
Changed by ISP	1-2	SRAM memory

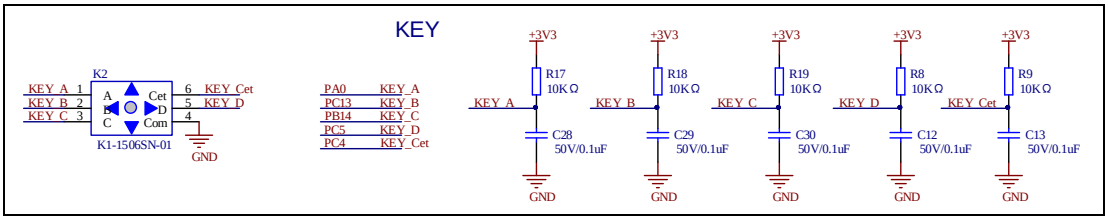
4.3. LED

Figure 4-3 Schematic diagram of LED function



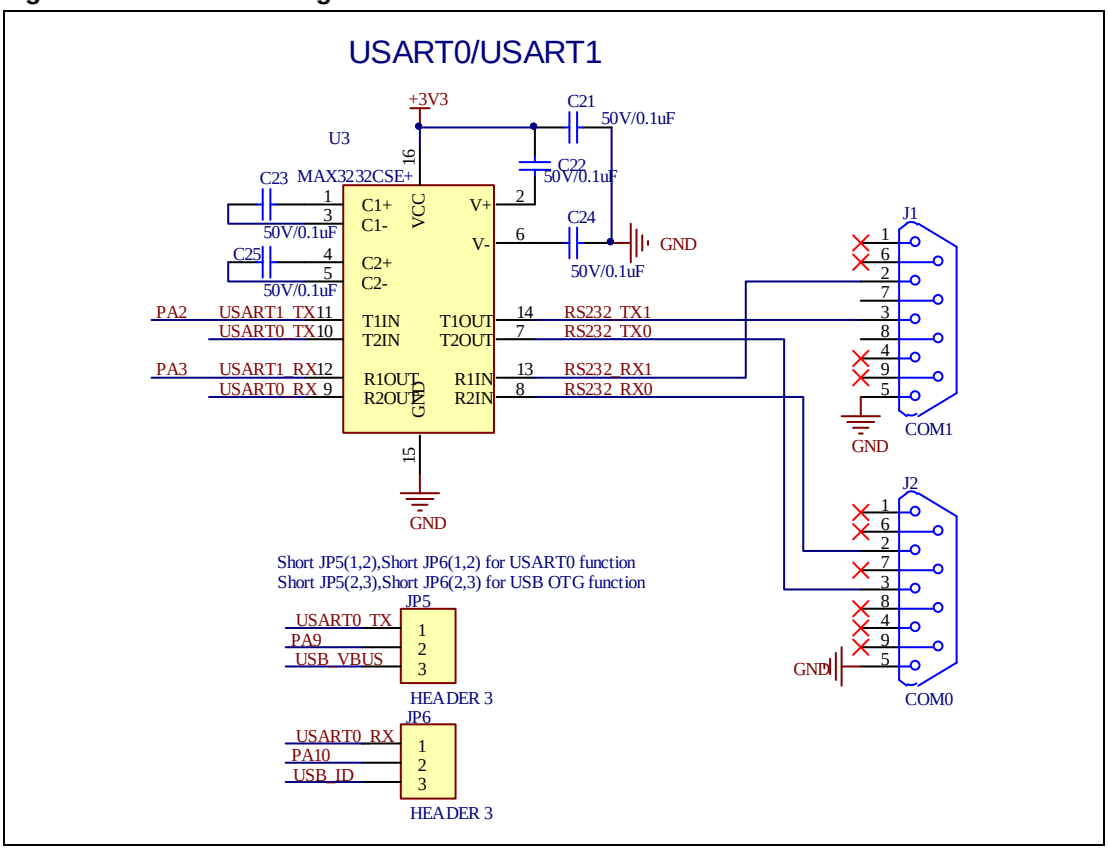
4.4. KEY

Figure 4-4 Schematic diagram of Key function



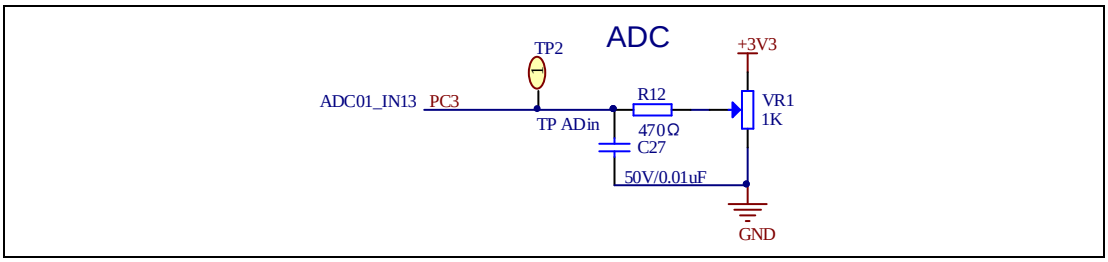
4.5. USART

Figure 4-5 Schematic diagram of USART function



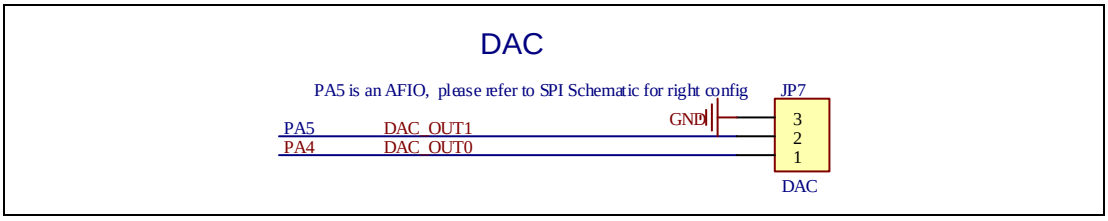
4.6. ADC

Figure 4-6 Schematic diagram of ADC function



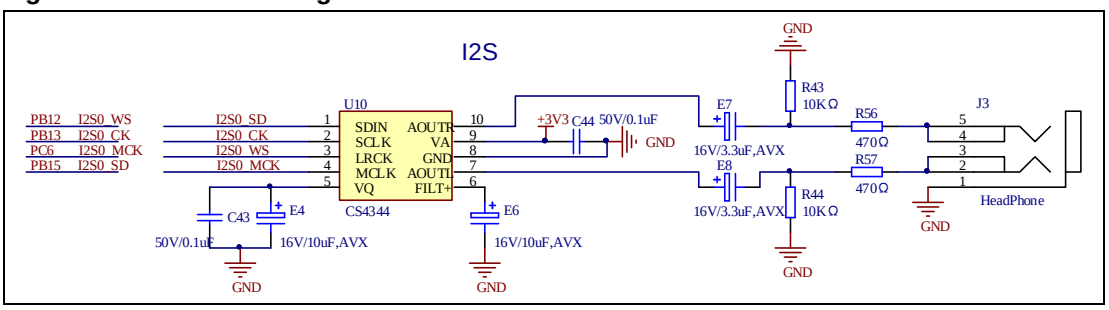
4.7. DAC

Figure 4-7 Schematic diagram of DAC function



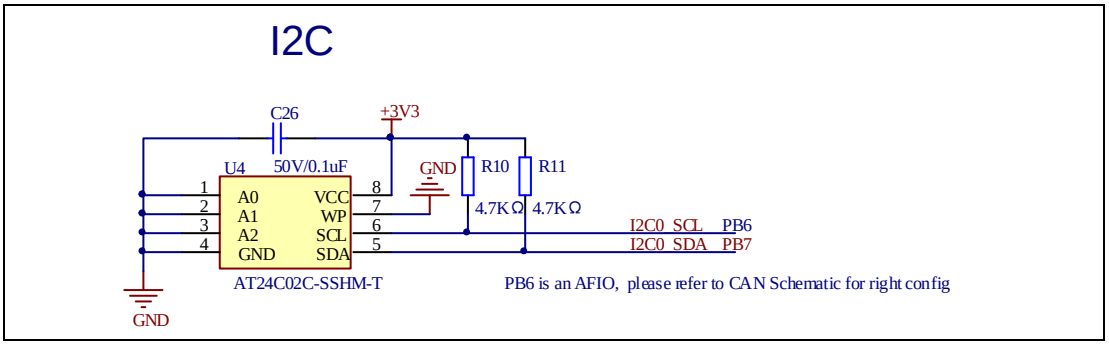
4.8. I2S

Figure 4-8 Schematic diagram of I2S function



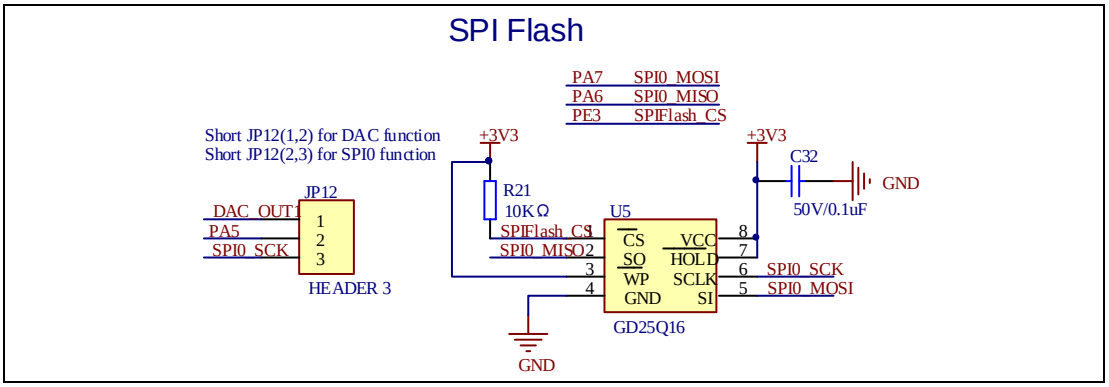
4.9. I2C

Figure 4-9 Schematic diagram of I2C function



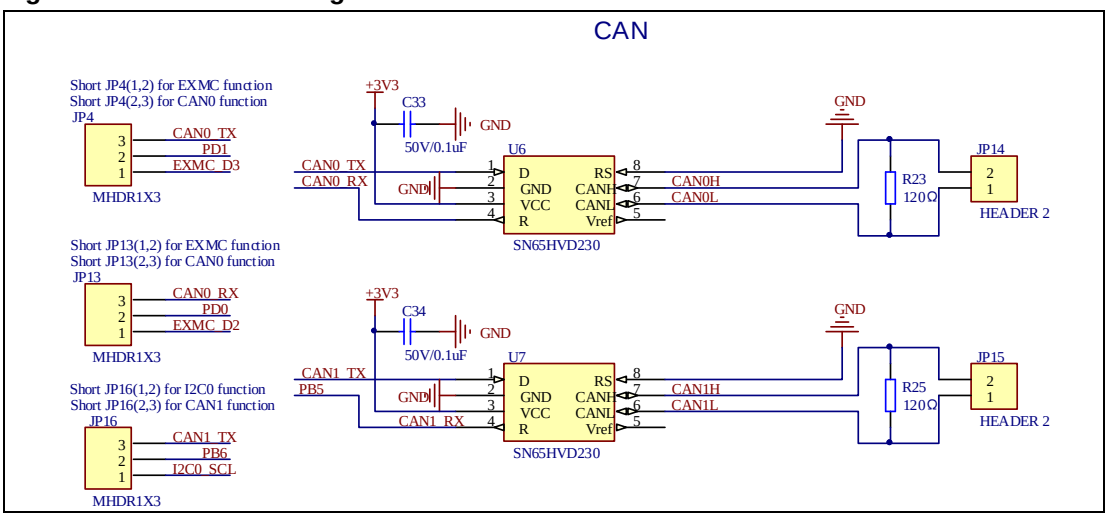
4.10. SPI

Figure 4-10 Schematic diagram of SPI function



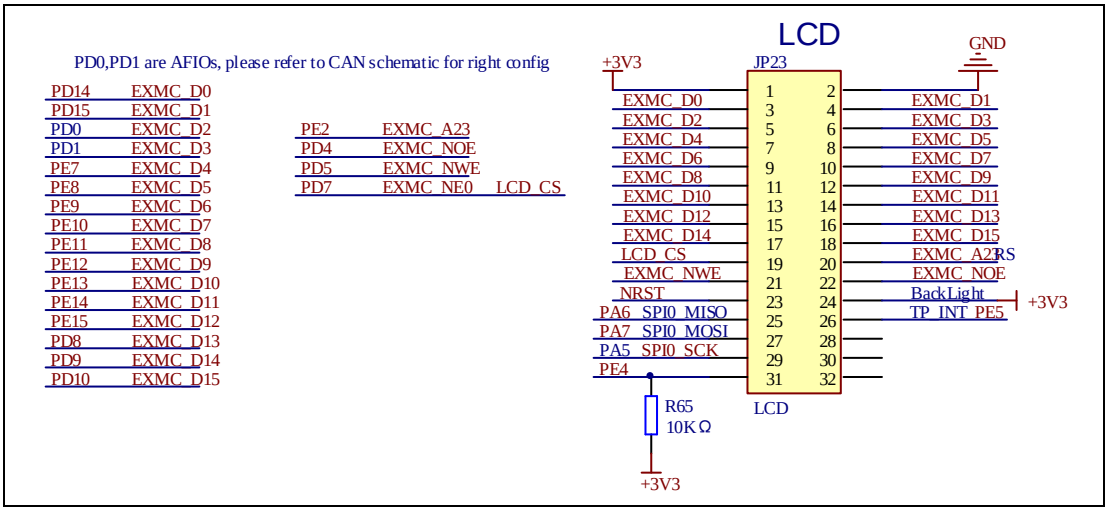
4.11. CAN

Figure 4-11 Schematic diagram of CAN function



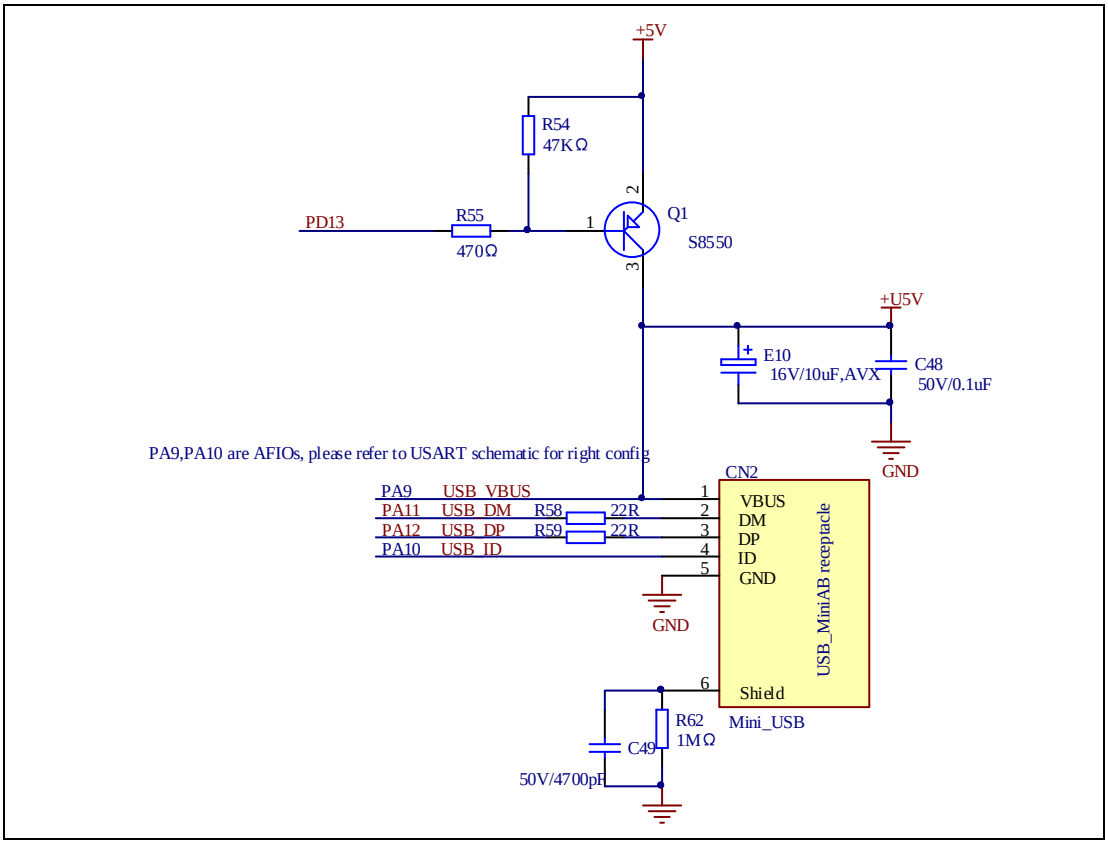
4.12. LCD

Figure 4-12 Schematic diagram of EXMC_LCD function



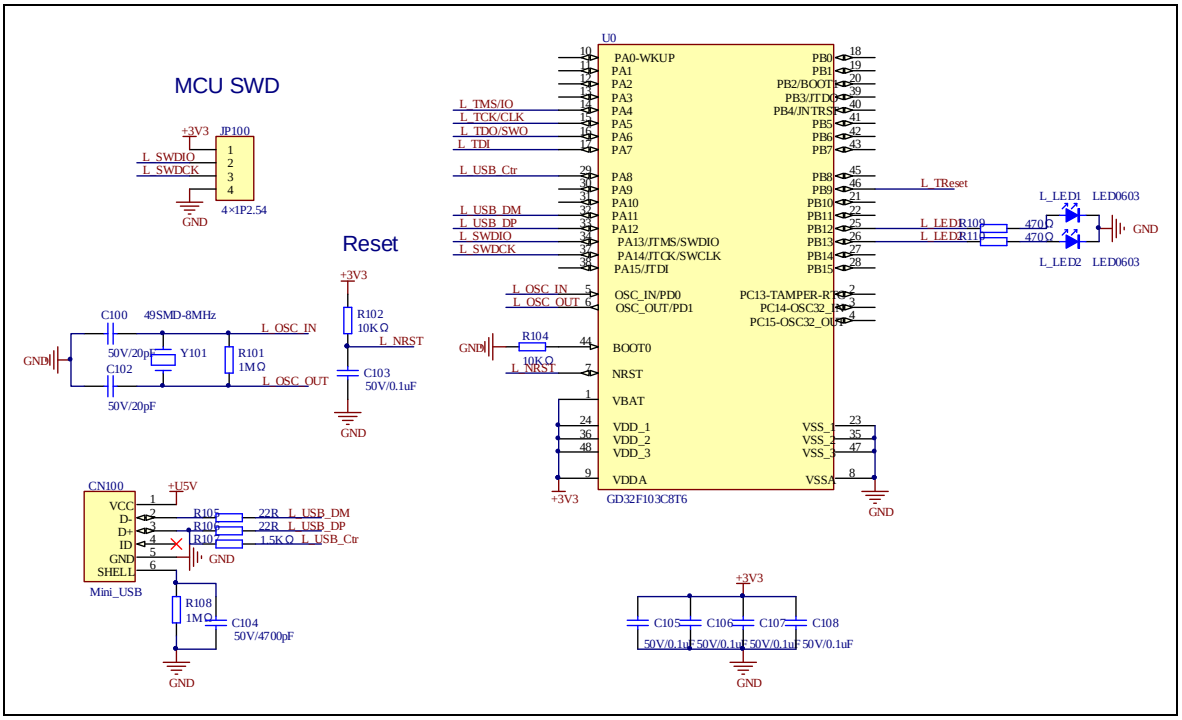
4.13. USBFS

Figure 4-13 Schematic diagram of USBFS function



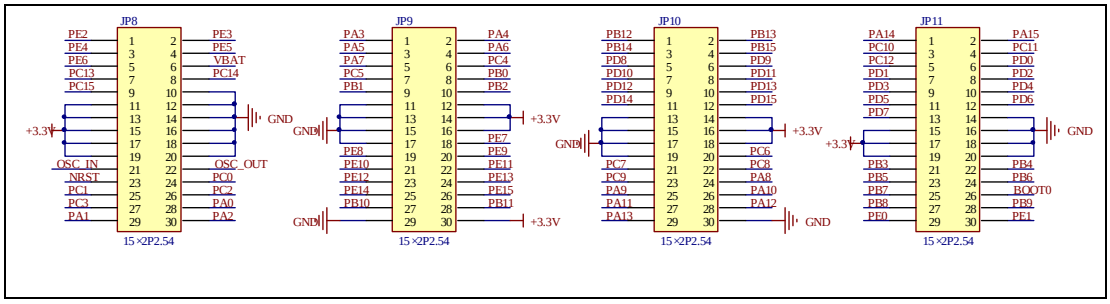
4.14. GD-Link

Figure 4-14 Schematic diagram of GD-Link function



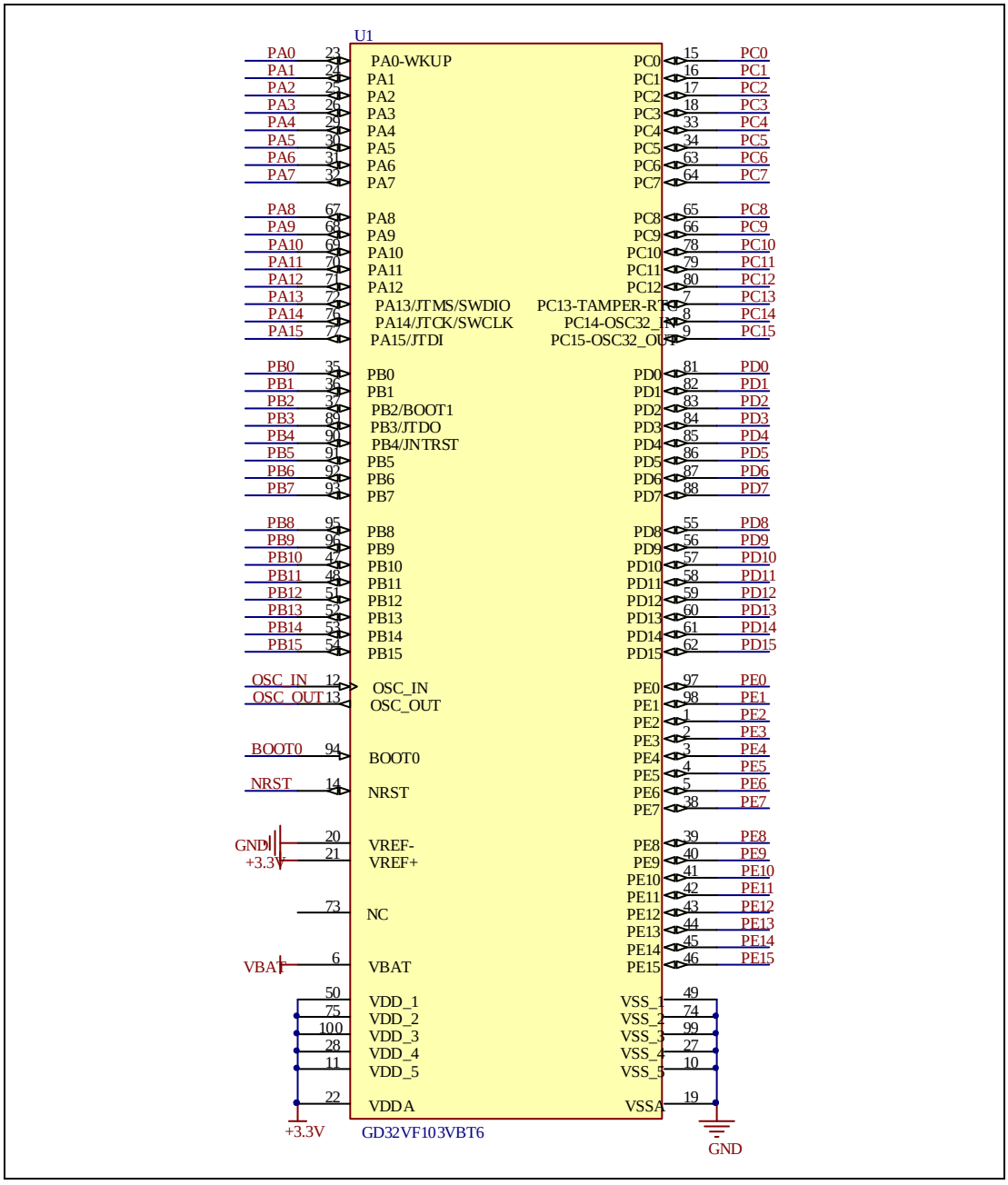
4.15. Extension

Figure 4-15 Schematic diagram of Extension Pin



4.16. MCU

Figure 4-16 Schematic diagram of MCU Pin



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 for controlling the LED
- Learn to use SysTick to generate 1ms delay

GD32VF103V-EVAL board has four LEDs. The LED1, LED2, LED3 and LED4 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, four LEDs will turn on one by one from LED1 to LED4 every 200ms, and then turn off together. 200ms later, the four LEDs work like previous again.

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

This demo will show how to use the KEY_B to control the LED2. When press down the KEY_B, it will check the input value of the IO port. If the value is 0, wait for 50ms. Then check the input value of the IO port again. If the value is still 0, indicates that the button is pressed down successfully, and light the LED2.

5.2.2. DEMO running result

Download the program <02_GPIO_Key_Polling_mode> to the EVAL board, first of all, all the LEDs will be flashed once for test. Then press down the KEY_B, LED2 will be turned on. Press down the KEY_B 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 to control the LED and the KEY
- Learn to use EXTI to generate external interrupt

This demo will show how to use EXTI interrupt line to control the LED2. When press down the KEY_B, 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, first of all, all the LEDs will be flashed once for test. Then press down the KEY_B, LED2 will be turned on. Press down the KEY_B 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 retarget the C library printf function to the USART

5.4.2. DEMO running result

Download the program < 04_USART_Printf > to the EVAL board, connect serial cable to EVAL_COM0 and jump JP5 and JP6 to USART0. This implementation outputs “USART printf example: please press the KEY_B” on the HyperTerminal using EVAL_COM0. Press the KEY_B, serial port will output “USART printf example”.

The output information via the serial port is as following.

```
USART printf example: please press the KEY_B
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 serial terminal tool

5.5.2. DEMO running result

Download the program < 05_USART_Echo_Interrupt_mode > to the EVAL board, connect serial cable to EVAL_COM0 and jump JP5 and JP6 to USART0. Firstly, all the LEDs are turned on and off for test. Then, the EVAL_COM0 sends the tx_buffer array (from 0x00 to 0xFF) to the serial terminal tool supporting hex format communication and waits for receiving data of BUFFER_SIZE bytes from the serial terminal. The data MCU has received is stored in the rx_buffer array. After that, compare tx_buffer with rx_buffer. If tx_buffer is same with rx_buffer, LED1, LED2, LED3, LED4 flash by turns. Otherwise, LED1, LED2,

LED3, LED4 toggle together.

The output information via the serial port 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
```

5.6. 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 EVAL_COM0 and jump JP5 and JP6 to USART0. Firstly, all the LEDs are turned on and off for test. Then, the EVAL_COM0 sends the tx_buffer array (from 0x00 to 0xFF) to the serial terminal tool supporting hex format communication and waits for receiving data of same bytes as tx_buffer from the serial terminal. The data MCU have received is stored in the rx_buffer array. After that, compare tx_buffer with rx_buffer. If tx_buffer is same with rx_buffer, LED1, LED2, LED3, LED4 flash by turns. Otherwise, LED1, LED2, LED3, LED4 toggle together.

The output information via the serial port 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 ADC to convert analog to digital
- Learn to use TIMER to generate a CC event
- Learn to use LCD to show the ADC converted result

TIMER1 CH1 event triggers ADC conversion, the value displayed on the LCD corresponds to the ADC analog input, and changes with it. The converted data are moved to SRAM through DMA continuously.

5.7.2. DEMO running result

Download the program <07_ADC_conversion_triggered_by_timer> to the GD32VF103V-EVAL board, adjust the adjustable potentiometer knob to change the analog input. The ADC, which is triggered by TIMER1 CH1 event, will convert the analog input, and you will see the result, a voltage curve, on the LCD. The curve adjusts with the analog input.

5.8. ADC0_ADC1_Follow_up_mode

5.8.1. 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 ADC0 and ADC1 follow-up mode

5.8.2. DEMO running result

Download the program <08_ADC0_ADC1_Follow_up_mode> to the GD32VF103V-EVAL board. Connect serial cable to EVAL_COM0, open the HyperTerminal. PA3 and PA0 pin voltage access by external voltage.

TIMER0_CH0 is the trigger source of ADC0 and ADC1. When the rising edge of TIMER0_CH0 coming, ADC0 starts immediately and ADC1 starts after a delay of several ADC clock cycles. The values of ADC0 and ADC1 are transmitted to array adc_value[0] and adc_value[1] by DMA.

When the first rising edge of TIMER0_CH0 coming, the value of the ADC0 conversion of PA0 pin is stored into the low half word of adc_value[0], and after a delay of several ADC clock cycles the value of the ADC1 conversion of PA3 pin is stored into the high half word of adc_value[0]. When the second rising edge of TIMER0_CH0 coming, the value of the ADC0

conversion of PA3 pin is stored into the low half word of adc_value[1], and after a delay of several ADC clock cycles the value of the ADC1 conversion of PA0 pin is stored into the high half word of adc_value[1].

When the program is running, HyperTerminal display the regular value of ADC0 and ADC1 by adc_value[0] and adc_value[1].

```
ADC0 regular channel 1 data = 0FFF0FEF

ADC0 regular channel 0 data = 0FEF0FFF

ADC0 regular channel 1 data = 0FFB0FFF

ADC0 regular channel 0 data = 0FEF0FFF

ADC0 regular channel 1 data = 0FFF0FFF

ADC0 regular channel 0 data = 0FD90FFF

ADC0 regular channel 1 data = 0FFF0FEF
```

5.9. ADC0_ADC1_Regular_Parallel_mode

5.9.1. 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 ADC0 and ADC1 regular parallel mode

5.9.2. DEMO running result

Download the program <09_ADC0_ADC1_Regular_Parallel_mode> to the GD32VF103V-EVAL board. Connect serial cable to EVAL_COM0, open the HyperTerminal. PA0 and PA3 pin connect to external voltage input.

TIMER0_CH0 is the trigger source of ADC0 and ADC1. When the rising edge of

TIMER0_CH0 coming, ADC0 and ADC1 convert the regular channel group parallelly. The values of ADC0 and ADC1 are transmitted to array adc_value[0] and adc_value[1] by DMA.

When the first rising edge of TIMER0_CH0 coming, the value of the ADC0 conversion of PA0 pin is stored into the low half word of adc_value[0], the value of the ADC1 conversion of PA3 pin is stored into the high half word of adc_value[0]. When the second rising edge of TIMER0_CH0 coming, the value of the ADC0 conversion of PA3 pin is stored into the low half word of adc_value[1], the value of the ADC1 conversion of PA0 pin is stored into the high half word of adc_value[1].

When the program is running, HyperTerminal displays the regular value of ADC0 and ADC1 stored in adc_value[0] and adc_value[1].

```
ADC0 regular channel 1 data = 0FFF0FFF


ADC0 regular channel 0 data = 0FE00FFF

ADC0 regular channel 1 data = 0FFF0FE9


ADC0 regular channel 0 data = 0FE00FFF

ADC0 regular channel 1 data = 0FFF0FE1


ADC0 regular channel 0 data = 0FEF0FFF

ADC0 regular channel 1 data = 0FFF0FFF
```

5.10. DAC_Output_Voltage_Value

5.10.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use DAC channel to generate different voltages on DAC output

5.10.2. DEMO running result

Download the program <10_DAC_Output_Voltage_Value> to the EVAL board, the digital value is 0x7ff0, its converted analog voltage should be VREF/2, using the voltmeter to measure PA4, its value is 1.648V.

5.11. I2C_EEPROM

5.11.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

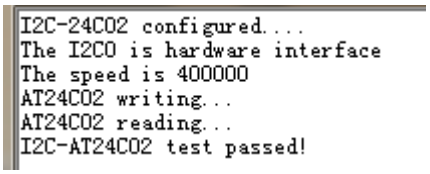
- Learn how to use the master transmitting mode of the I2C module
- Learn how to use the master receiving mode of the I2C module
- Learn to read and write the EEPROM with the I2C interface

5.11.2. DEMO running result

Jump JP5 and JP6 to USART, jump JP16 to I2C, then download the program <11_I2C_EEPROM> to the EVAL board and run. Connect serial cable to COM, and open the HyperTerminal to show the print message.

Firstly, the data of 256 bytes will be written to the EEPROM from the address 0x00. Then, reading the EEPROM from address 0x00 for 256 bytes. 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 four LEDs lights flashing, otherwise the serial port will output "Err: data read and write aren't matching." and all the four LEDs light.

The output information via the serial port is as following.



5.12. SPI_SPI_Flash

5.12.1. DEMO purpose

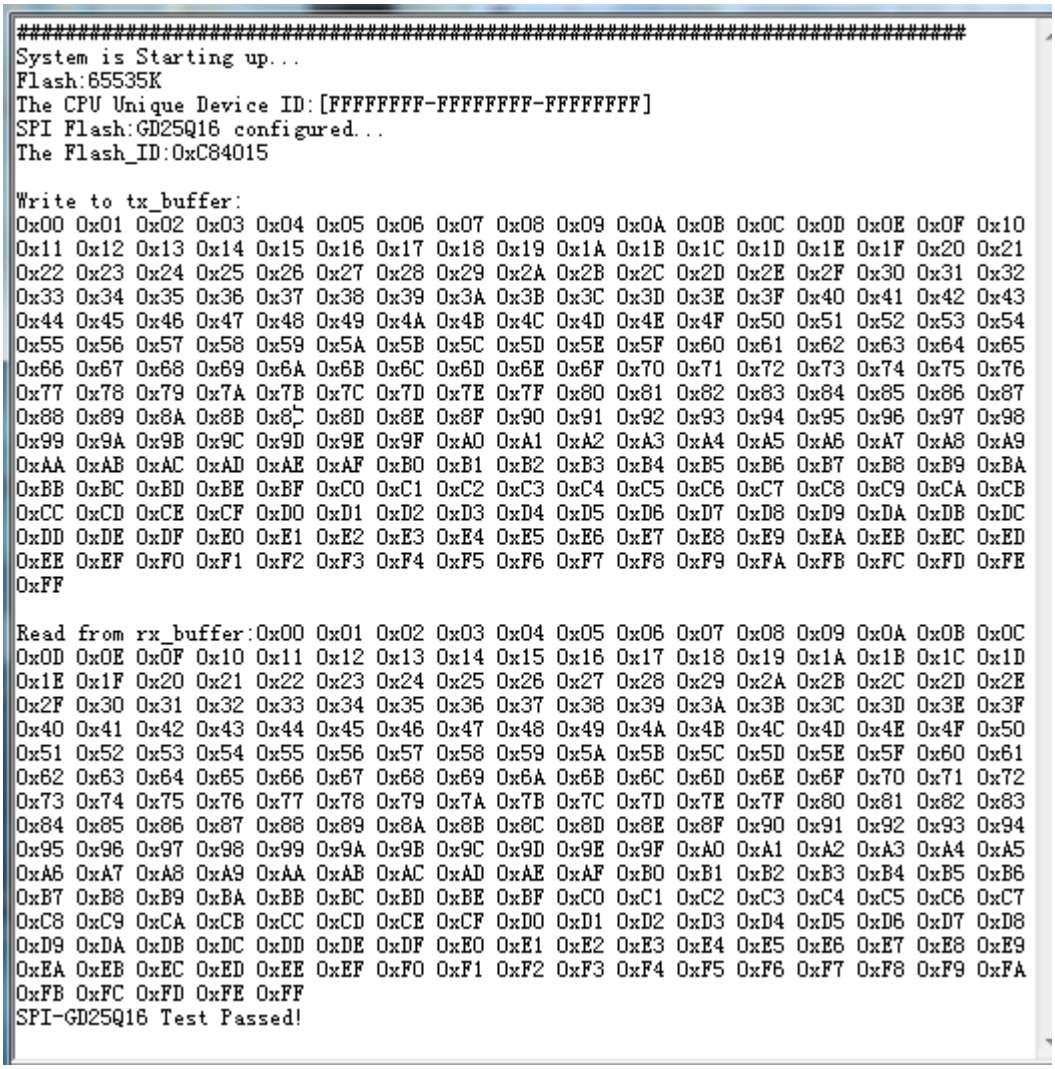
This demo includes the following functions of GD32 MCU:

- Learn to use the master mode of SPI unit to read and write NOR Flash with the SPI interface

5.12.2. DEMO running result

The computer serial port line connected to the COM0 port of development board, set the baud rate of HyperTerminal software to 115200, 8 bits data bit, 1 bit stop bit. At the same time, you should jump the JP12 to SPI, jump JP5 and JP6 to USART.

Download the program <12_SPI_SPI_Flash> to the EVAL board, the HyperTerminal software can observe the operation condition and will display the ID of the flash, 256 bytes data which are written to and read from flash. Compare the data that were written to the flash and the data that were read from the flash. If they are the same, the serial port will output “SPI-GD25Q16 Test Passed!”, otherwise, the serial port will output “Err: Data Read and Write aren't Matching.”. At last, turn on and off the LEDs one by one. The following is the experimental results.



5.13. I2S_Audio_Player

5.13.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use I2S module to output audio file

GD32VF103V-EVAL board integrates the I2S (Inter-IC Sound) module, and the module can communicate with external devices using the I2S audio protocol. This demo mainly shows how to use the I2S interface of the board for audio output.

5.13.2. DEMO running result

Download the program <13_I2S_Audio_Player> to the EVAL board. After downloading the program, insert the earphone into the audio port J3, then listen to the audio file.

5.14. EXMC_TouchScreen

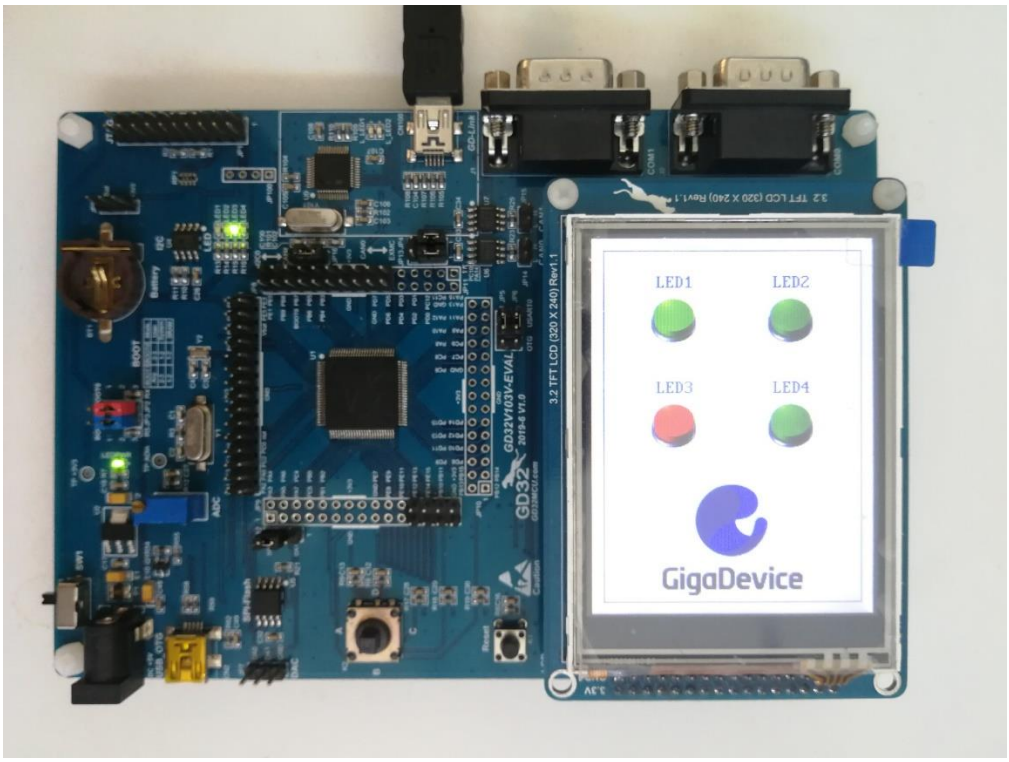
5.14.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use EXMC control LCD
- Learn to use SPI control touch panel

5.14.2. DEMO running result

GD32VF103V-EVAL board has EXMC module to control LCD. Before running the demo, JP12 must be fitted to the SPI port, JP13 and JP14 must be fitted to the EXMC port. Download the program <14_EXMC_TouchScreen> to the EVAL board. This demo displays GigaDevice logo and four green buttons on the LCD screen by EXMC module. Users can touch the green button to turn on the corresponding LED on board, and then the color of button you had touched will change to red.



5.15. CAN_Network

5.15.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use the CAN0 communication between two boards

GD32VF103V-EVAL development board integrates the CAN (Controller Area Network) bus controller, which is a common industrial control bus. CAN bus controller follows the CAN bus protocol of 2.0 A and 2.0 B. This demo mainly shows how to communicate two EVAL boards through CAN0.

5.15.2. DEMO running result

This example is tested with two GD32VF103V-EVAL boards. Jump the JP5, JP6 to USART and JP13, JP4 to CAN with the jumper cap. Connect L pin to L pin and H pin to H pin of JP14 on the boards for sending and receiving frames. Download the program <15_CAN_Network> to the two EVAL boards, and connect serial cable to EVAL_COM0. Firstly, the EVAL_COM0 sends “please press the CET key to transmit data!” to the HyperTerminal. The frames are sent and the transmit data are printed by pressing CET Key push button. When the frames are received, the receive data will be printed and the LED2 will toggle one time.

The output information via the serial port is as following.

```
please press the CET key to transmit data!
CAN0 transmit data: ab,cd

CAN0 receive data: ab,cd
```

5.16. RCU_Clock_Out

5.16.1. 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

5.16.2. DEMO running result

Download the program <16_RCU_Clock_Out> to the EVAL board and run. Connect serial cable to EVAL_COM, 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 key B. 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 pin.
Information via a serial port output as following:

```
/========== Gigadevice Clock output Demo ==========/
press key B to select clock output source
CK_OUT0: system clock
CK_OUT0: IRC8M
CK_OUT0: HXTAL
```

5.17. PMU_sleep_wakeup

5.17.1. 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

5.17.2. DEMO running result

Download the program < 17_PMU_sleep_wakeup > to the EVAL board, connect serial cable to EVAL_COM. 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.

5.18. RTC_Calendar

5.18.1. 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

5.18.2. DEMO running result

Jump the JP6 to USART0 with the jumper cap, Download the program <18_RTC_Calendar> to the EVAL board and Connect serial cable to COM. If the development board run the program for the first time, serial port output following information "RTC not yet configured…." It requires the user to set up hours、minutes and seconds.

According to the serial port output information prompt, setting time, as shown below, serial port output following information.

If the development board is not the first run of the program, time has been set up in the last run, after the system reset, as shown below, serial port output following information " No need to configured RTC....", serial port continue printing time information

5.19. TIMER_Breath_LED

5.19.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

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

5.19.2. DEMO running result

Use the DuPont line to connect the TIMER0_CH0 (PA8) and LED1 (PC0), and then download the program <19_TIMER_Breath_LED> to the GD32VF103V-EVAL 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.

5.20. USBFS_Device

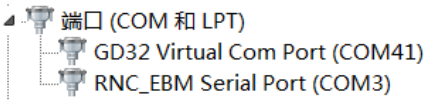
5.20.1. CDC_ACM

DEMO purpose

This demo includes the following functions of GD32 MCU:

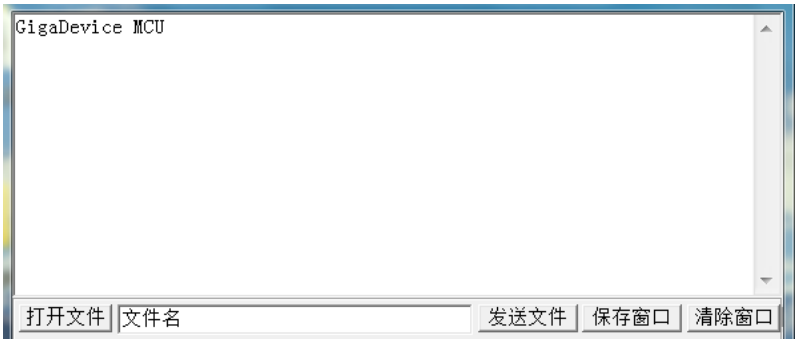
- Learn how to use the USBFS peripheral
- Learn how to implement USB CDC device

EVAL board has one USBFS interface. In this demo, the EVAL board is enumerated as an USB virtual COM port, which was shown in device manager of PC as below. This demo makes the USB device look like a serial port, and loops back the contents of a text file over USB port. To run the demo, input a message using the PC's keyboard. Any data that shows in HyperTerminal is received from the device.



DEMO running result

Download the program <20_USBFS\Device\CDC_ACM> to the EVAL board and run. When you input message through computer keyboard, the HyperTerminal will receive and shown the message. For example, when you input “GigaDevice MCU”, the HyperTerminal will get and show it as below.



5.20.2. MSC

DEMO Purpose

This demo includes the following functions of GD32 MCU:

- Learn how to use the USBFS

- Learn how to implement USB MSC(mass storage) device

This demo mainly implements a U disk. U disk is currently very widely used removable MSC devices. MSC, the Mass Storage device Class, is a transport protocol between a computer and mobile devices, which allow a universal serial bus (USB) equipment to access a host computing device, file transfer between them, mainly including mobile hard disk, mobile U disk drive, etc. The MSC device must have a storage medium, and this demo uses the MCU's internal SRAM as the storage medium. For more details of the MSC protocol please refer to the MSC protocol standard.

MSC device will use a variety of transport protocols and command formats for communication, so it need to choose the appropriate protocol and command format in the realization of the application. This demo selects the BOT (bulk only transport) protocol and the required SCSI (small computer interface) command, and is compatible with a wide variety of Window operating systems. Specific BOT protocol and SCSI command specification please refer to the standard of their agreement.

DEMO Running Result

Download the program <20_USBFS\Device\MSC > to the EVAL board and run. When the EV-board connect to the PC, you will find a USB large capacity storage device is in the universal serial bus controller, and there is 1 more disk drives in the equipment manager of PC.

Then, after opening the resource manager, you will see more of the 1 disk, as shown in the following diagram:

At this point, the write/read/formatting operation can be performed as the other mobile devices.

5.21. USBFS_Host

5.21.1. HID

DEMO Purpose

This demo includes the following functions of GD32 MCU:

- Learn to use the USBFS as a HID host
- Learn the operation between the HID host and the mouse device
- Learn the operation between the HID host and the keyboard device

GD32VF103V-EVAL board integrates the USBFS module, and the module can be used as a

USB device, a USB host or an OTG device. This demo mainly shows how to use the USBFS as a USB HID host to communicate with external USB HID device.

DEMO Running Result

Jump the JP5 to OTG. Then download the program <20_USBFS\Host\HID_Host> to the EVAL board and run.

If a mouse has been attached, the user will see the information of mouse enumeration. First pressing the CET key will see the inserted device is mouse, and then moving the mouse will show the position of mouse and the state of button in the screen.

If a keyboard has been attached, the user will see the information of keyboard enumeration. First pressing the CET key will see the inserted device is keyboard, and then pressing the keyboard will show the state of the button in the screen.

5.21.2. MSC

DEMO Purpose

This demo includes the following functions of GD32 MCU:

- Learn to use the USBFS as a MSC host
- Learn the operation between the MSC host and the Udisk

GD32VF103V-EVAL board integrates the USBFS module, and the module can be used as a USB device, a USB host or an OTG device. This demo mainly shows how to use the USBFS as a USB MSC host to communicate with external Udisk.

DEMO Running Result

Jump the JP5, JP6 to OTG. Then insert the OTG cable to the USB port, download the program <20_USBFS\Host\MSC_Host> to the EVAL board and run.

If an Udisk has been attached, the user will see the information of Udisk enumeration. First pressing the CET key will see the Udisk information, next pressing the CET key will see the root content of the Udisk, then press the C key will write file to the Udisk, finally the user will see information that the MSC host demo is end.

6. Revision history

Table 6-1 Revision history

Revision No.	Description	Date
1.0	Initial Release	Jun.05, 2019
1.1	Update the titles of chapter 5.20 and 5.21, update new logo in EXMC_TouchScreen	Sept.18, 2019

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.

© 2019 GigaDevice – All rights reserved