

GigaDevice Semiconductor Inc.

GD32M531R-EVAL

Arm[®] Cortex[®]-M33 32-bit MCU

User Guide

Revision 1.0

(Feb. 2026)

Tables of Contents

TABLES OF CONTENTS	1
LIST OF FIGURES	3
LIST OF TABLES	4
1. SUMMARY	5
2. FUNCTION PIN ASSIGN	5
3. GETTING STARTED	6
4. HARDWARE LAYOUT OVERVIEW	6
4.1. Power supply	6
4.2. Boot option	7
4.3. LED	7
4.4. KEY	7
4.5. UART	8
4.6. ADC	8
4.7. CMP	8
4.8. DAC	9
4.9. I2C.....	9
4.10. SPI.....	9
4.11. CAN	10
4.12. Extension.....	10
4.13. GD-Link.....	11
4.14. MCU.....	12
5. ROUTINE USE GUIDE	13
5.1. GPIO_Running_LED	13
5.1.1. DEMO purpose	13
5.1.2. DEMO running result	13
5.2. GPIO_Key_Polling_mode.....	13
5.2.1. DEMO purpose	13
5.2.2. DEMO running result	13
5.3. EXTI_Key_Interrupt_mode.....	14
5.3.1. DEMO purpose	14
5.3.2. DEMO running result	14
5.4. UART_Printf	14
5.4.1. DEMO purpose	14
5.4.2. DEMO running result	14
5.5. UART_HyperTerminal_Interrupt	15
5.5.1. DEMO purpose	15
5.5.2. DEMO running result	15

5.6. UART_DMA	15
5.6.1. DEMO purpose	15
5.6.2. DEMO running result	15
5.7. ADC_Temperature_Vrefint	16
5.7.1. DEMO purpose	16
5.7.2. DEMO running result	16
5.8. ADC_Conversion_Triggered_By_Timer	17
5.8.1. DEMO purpose	17
5.8.2. DEMO running result	17
5.9. DAC_Output_Voltage_Value	17
5.9.1. DEMO purpose	17
5.9.1. DEMO running result	17
5.10. Comparator_Obtain_Brightness	18
5.10.1. DEMO purpose	18
5.10.2. DEMO running result	18
5.11. I2C_EEPROM	18
5.11.1. DEMO purpose	18
5.11.2. DEMO running result	18
5.12. QSPI_FLASH	19
5.12.1. DEMO purpose	19
5.12.2. DEMO running result	20
5.13. SPI_TFT_LCD_Driver	21
5.13.1. DEMO purpose	21
5.13.2. DEMO running result	21
5.14. RCU_Clock_Out	21
5.14.1. DEMO purpose	21
5.14.2. DEMO running result	22
5.15. PMU_Sleep_Wakeup	22
5.15.1. DEMO purpose	22
5.15.2. DEMO running result	22
5.16. TIMER_Breath_LED	22
5.16.1. DEMO purpose	22
5.16.2. DEMO running result	23
5.17. CAN_Network	23
5.17.1. DEMO purpose	23
5.17.2. DEMO running result	23
6. REVISION HISTORY	24

List of Figures

Figure 4-1. Schematic diagram of power supply.....	6
Figure 4-2. Schematic diagram of boot option	7
Figure 4-3. Schematic diagram of LED function	7
Figure 4-4. Schematic diagram of Key function	7
Figure 4-5. Schematic diagram of UART	8
Figure 4-6. Schematic diagram of ADC	8
Figure 4-7. Schematic diagram of CMP	8
Figure 4-8. Schematic diagram of DAC	9
Figure 4-9. Schematic diagram of I2C	9
Figure 4-10. Schematic diagram of SPI	9
Figure 4-11. Schematic diagram of CAN	10
Figure 4-12. Schematic diagram of Extension.....	10
Figure 4-13. Schematic diagram of GD-Link.....	11
Figure 4-14. Schematic diagram of MCU.....	12

List of Tables

Table 2-1. Function pin assignment.....	5
Table 6-1. Revision history	24

1. Summary

GD32M531R-EVAL uses GD32M531RCT7 as the main controller. It uses GD-Link Type-C interface to supply 5V power. Reset, Boot, Wakeup KEY, USER KEY, LED, ADC, CAN, CMP, DAC, I2C_EEPROM, SPI_LCD, QSPI_FLASH, UART to USB interface are also included. For more details please refer to GD32M531R-EVAL-V1.2 schematic.

2. Function Pin Assign

Table 2-1. Function pin assignment

Function	Pin	Description
LED	PD9	LED1
	PD10	LED2
	PD11	LED3
	PD12	LED4
RESET	PN5	K1-Reset
KEY	PA0	K2-Wakeup
	PB0	K3-User
ADC	PC0	ADC0_IN0
CMP	PC1	CMP_IP
	PC11	CMP_IM
	PC12	
	PD4	
	PD5	
DAC	PD4	DAC0_OUT
I2C	PF9	I2C_SCL
	PF10	I2C_SDA
SPI Flash	PD8	SPIFlash_CS
	PF11	SPI_SCK
	PF8	SPI_MOSI
	PF12	SPI_MISO
	PF9	SPI_IO2
	PF10	SPI_IO3
SPI_LCD	PC7	SPILCD_D/C
	PC6	SPILCD_RESET
	PG15	SPILCD_CS
	PF11	SPI_SCK
	PF8	SPI_MOSI
	PF12	SPI_MISO
UART	UART3_TX	PF13
	UART3_RX	PF14

Function	Pin	Description
CAN	CAN_TX	PF13
	CAN_RX	PF14

3. Getting started

The EVAL board uses GD-Link Type-C interface to get power DC +5V, which is the hardware system normal work voltage. A J-Link tool or GD-Link tool 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 OK.

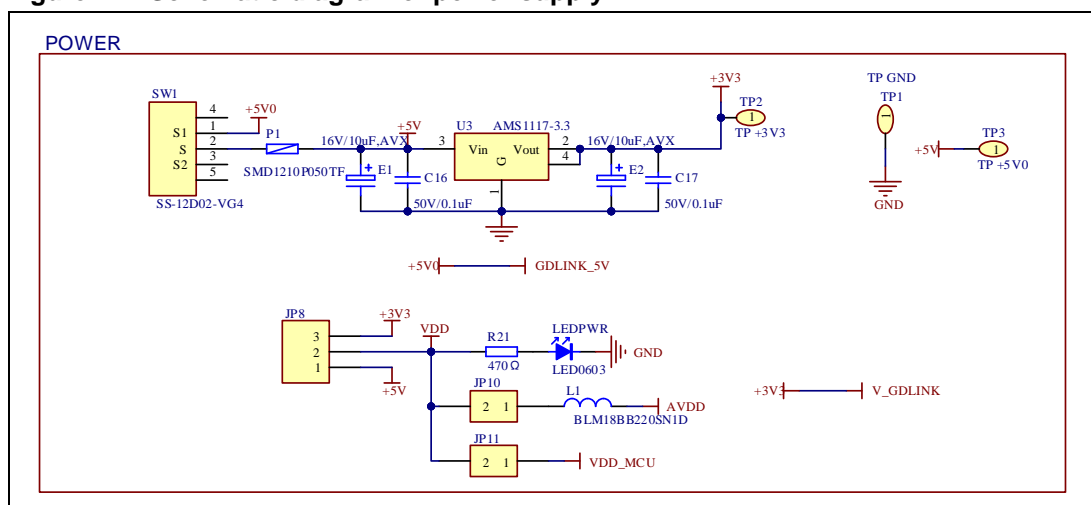
There are Keil version, IAR version and GD32EBuilder version of all projects. Keil version of the projects are created based on Keil MDK-ARM 5.29 uVision5. IAR version of the projects are created based on IAR Embedded Workbench for ARM 8.32.1 and GD32EBuilder version of the projects are created based on GD32EmbeddedBuilder_v1.5.5_Rel. During use, the following points should be noted:

1. If you use Keil uVision5 to open the project. In order to solve the "Device Missing (s)" problem, the latest version of GigaDevice.GD32M53x_DFP (URL: <https://www.gd32mcu.com>) should be installed to load related files.
2. If you use IAR to open the project, the latest version of IAR_GD32M53x_ADDON(URL: <https://www.gd32mcu.com>) should be installed to load related files.

4. Hardware layout overview

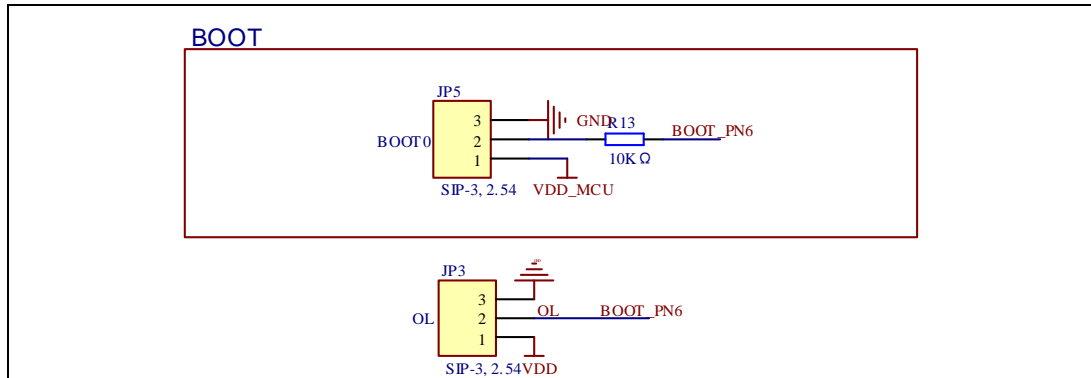
4.1. Power supply

Figure 4-1. Schematic diagram of power supply



4.2. Boot option

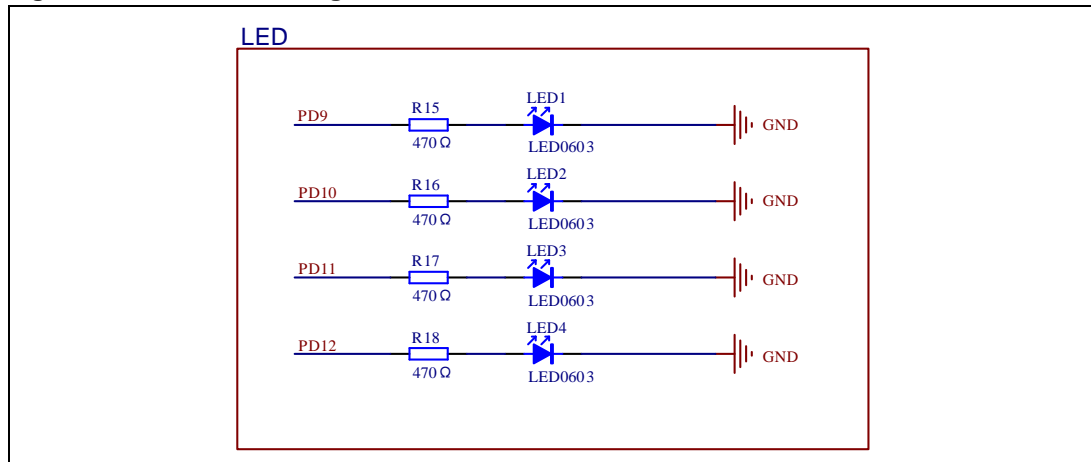
Figure 4-2. Schematic diagram of boot option



Note: Since OL pin of onboard debugger is internal pull-up by default, if booting from system memory, JP3 needs to be configured to GND; If booting from the main Flash, configure JP5 to VDD_MCU and not JP3 to GND.

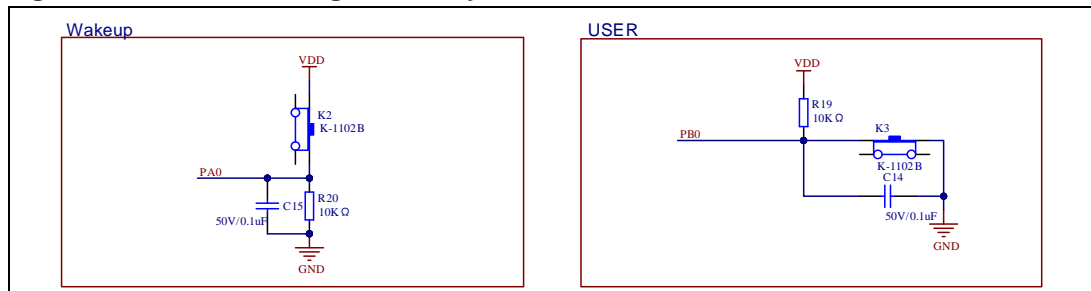
4.3. LED

Figure 4-3. Schematic diagram of LED function



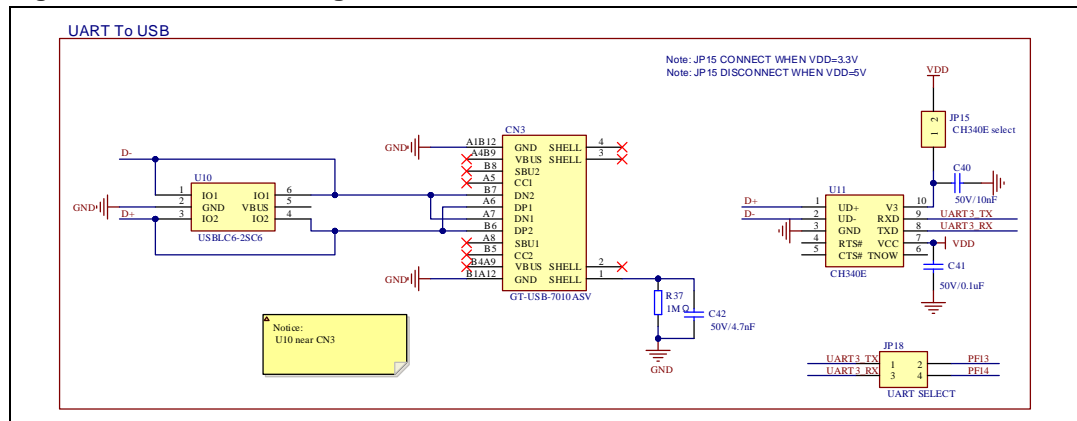
4.4. KEY

Figure 4-4. Schematic diagram of Key function



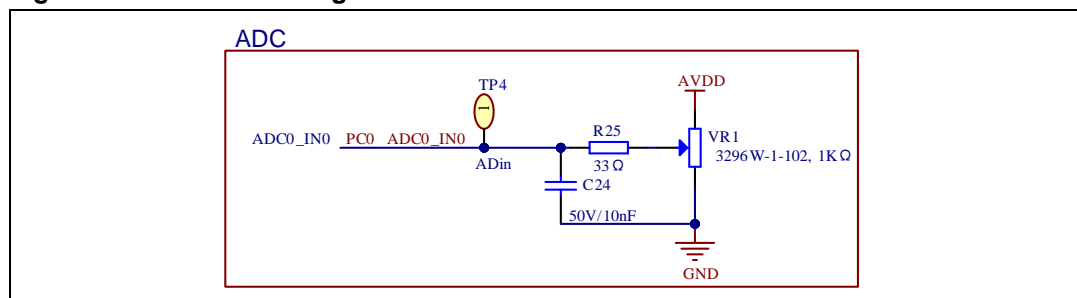
4.5. UART

Figure 4-5. Schematic diagram of UART



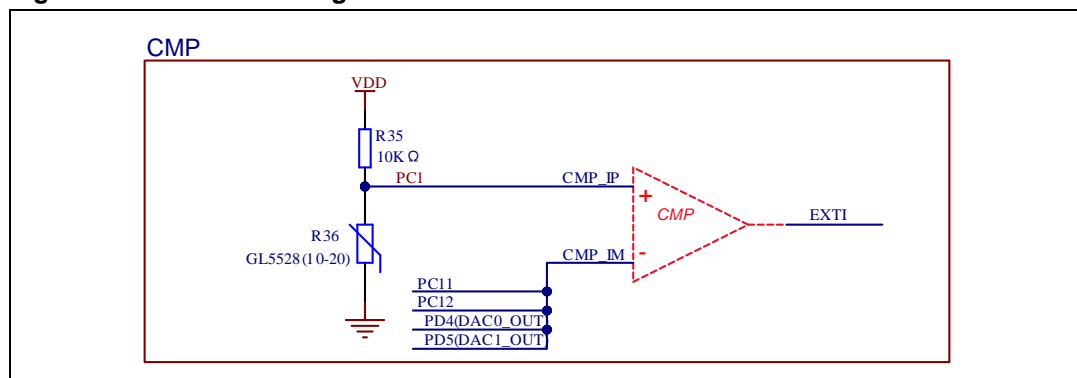
4.6. ADC

Figure 4-6. Schematic diagram of ADC



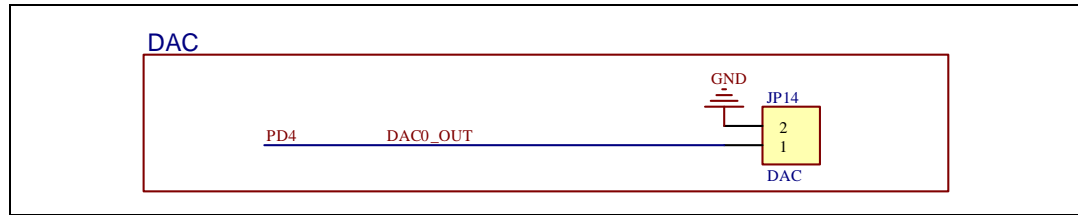
4.7. CMP

Figure 4-7. Schematic diagram of CMP



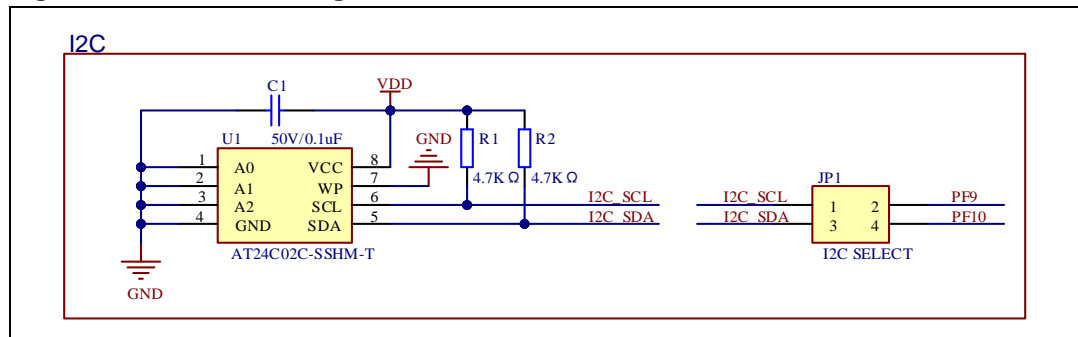
4.8. DAC

Figure 4-8. Schematic diagram of DAC



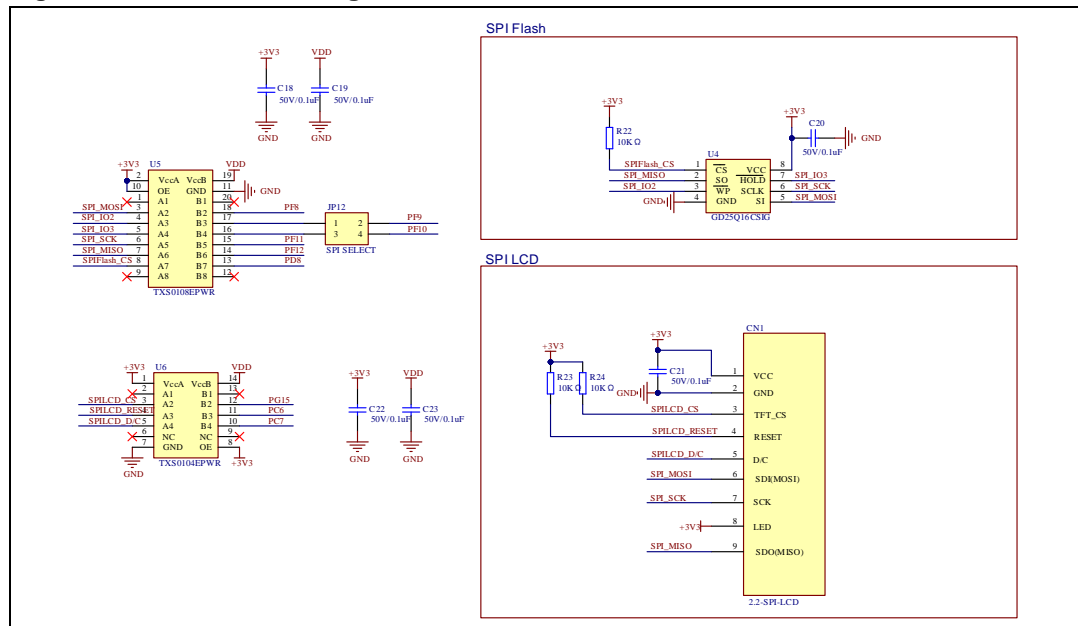
4.9. I2C

Figure 4-9. Schematic diagram of I2C



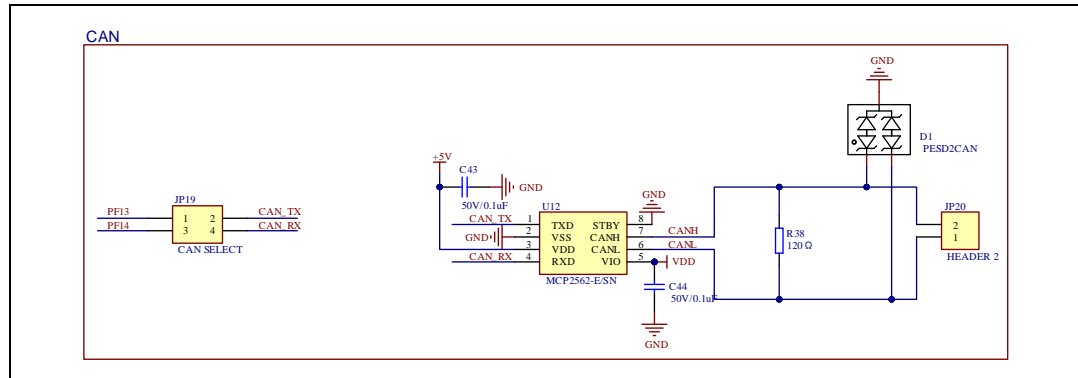
4.10. SPI

Figure 4-10. Schematic diagram of SPI



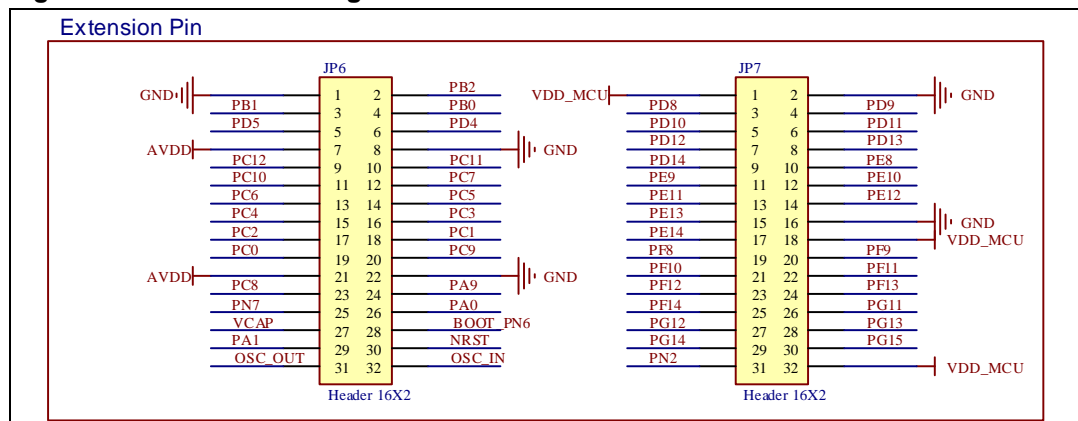
4.11. CAN

Figure 4-11. Schematic diagram of CAN



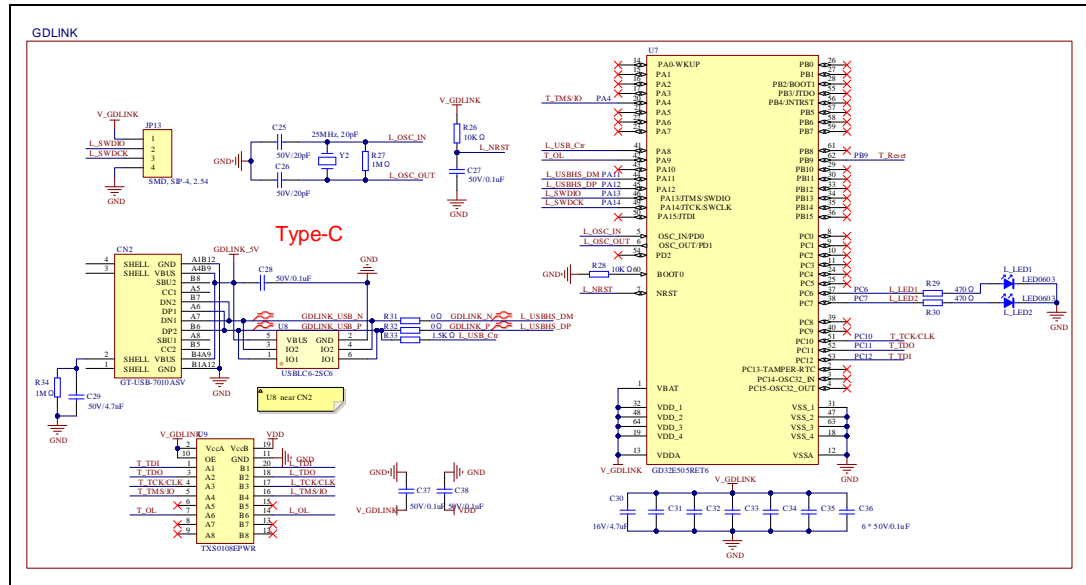
4.12. Extension

Figure 4-12. Schematic diagram of Extension



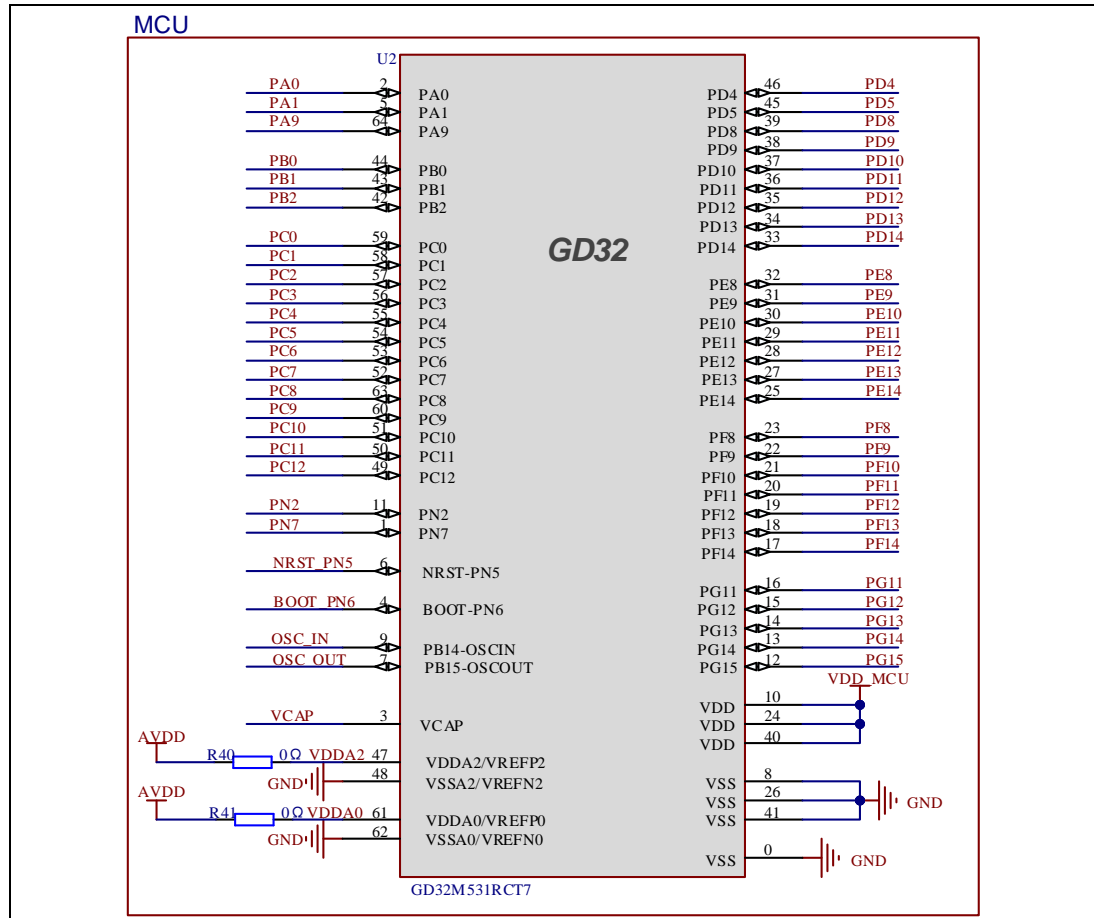
4.13. GD-Link

Figure 4-13. Schematic diagram of GD-Link



4.14. MCU

Figure 4-14. 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.

GD32M531R-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, LED1, LED2, LED3 and LED4 will change the state like running water and then repeat the whole process over and over 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.

GD32M531R-EVAL board has three keys and four LEDs. The three keys are Reset key, Wakeup key and User key. The LED1, LED2, LED3 and LED4 are controlled by GPIO.

This demo will show how to use the User key to control the LED1. When press down the User Key, 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 LED1.

5.2.2. DEMO running result

Download the program <02_GPIO_Key_Polling_mode> to the EVAL board. Press down the User Key, LED1 will be turned on. Press down the User Key again, LED1 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

GD32M531R-EVAL board has three keys and four LEDs. The three keys are Reset key, Wakeup key and User key. The LED1, LED2, LED3 and LED4 are controlled by GPIO.

This demo will show how to use the EXTI interrupt line to control the LED1. When press down the User key, it will produce an interrupt. In the interrupt service function, the demo will toggle LED1.

5.3.2. DEMO running result

Download the program <03_EXTI_Key_Interrupt_mode> to the EVAL board. After startup, the LED1 flash once, press down the User key, LED1 will be turned on, press down the User key again, LED1 will be turned off.

5.4. UART_Printf

5.4.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use GPIO control the LED
- Learn to retarget the C library printf function to the UART

5.4.2. DEMO running result

Download the program < 04_UART_Printf > to the EVAL board, ensure JP18 is connected, and then connect serial cable to UART. Firstly, all the LEDs flash 2 times for test. Then, this implementation outputs "UART printf example: please press the User key" on the HyperTerminal using UART. Press the User key, the serial port will output "UART printf example".

The output information via the HyperTerminal is as following:

```
UART printf example: please press the User key

UART printf example
```

5.5. UART_HyperTerminal_Interrupt

5.5.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use the UART transmit and receive interrupts to communicate with the HyperTerminal.

5.5.2. DEMO running result

Download the program <05_UART_HyperTerminal_Interrupt> to the EVAL board, ensure JP18 is connected, and then connect serial cable to UART. Firstly, all the LEDs are turned on and off for test. Then, the UART 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, LED3 and LED4, flash by turns. Otherwise, LED1, LED2, LED3 and LED4 toggle together.

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
```

5.6. UART_DMA

5.6.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use the UART transmit and receive data using DMA.

5.6.2. DEMO running result

Download the program <06_UART_DMA> to the EVAL board, ensure JP18 is connected, and then connect serial cable to UART. Firstly, all the LEDs are turned on and off for test. Then, the UART 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, LED3 and LED4, flash by turns. Otherwise, LED1, LED2, LED3 and LED4 toggle together.

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_Temperature_Vrefint

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 convert the value of temperature sensor channel and V_{REFINT} channel

5.7.2. DEMO running result

Download the program <07_ADC_Temperature_Vrefint> to the EVAL board, and ensure JP18 is connected.

Connect serial cable to EVAL_COM, open the HyperTerminal.

When the program is running, HyperTerminal display the value of temperature and internal voltage reference (V_{REFINT}).

When the program runs, the LED will keep blinking.

Notice: because there is an offset, when inner temperature sensor is used to detect accurate temperature, an external temperature sensor part should be used to calibrate the offset error.

```
the temperature data is 27 degrees Celsius
the reference voltage data is 1.188V

the temperature data is 29 degrees Celsius
the reference voltage data is 1.193V
```

5.8. ADC_Conversion_Triggered_By_Timer

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 TIMER to generate an ADC trigger event

5.8.2. DEMO running result

Download the program < 08_ADC_Conversion_Triggered_By_Timer > to the EVAL board, and ensure JP18 is connected, adjust the adjustable potentiometer knob to change the analog input. The ADC, which is triggered by TIMER0_TRGOF event, will convert the analog input, and you will see the result by COM.

When the program runs, the LED will keep blinking.

```
the result of ADC0_IN0 is 1608
the result of ADC0_IN0 is 1611
the result of ADC0_IN0 is 1613
the result of ADC0_IN0 is 1613
the result of ADC0_IN0 is 1612
the result of ADC0_IN0 is 1610
the result of ADC0_IN0 is 1611
the result of ADC0_IN0 is 1602
the result of ADC0_IN0 is 1566
the result of ADC0_IN0 is 1544
the result of ADC0_IN0 is 1533
the result of ADC0_IN0 is 1533
the result of ADC0_IN0 is 1535
the result of ADC0_IN0 is 1531
the result of ADC0_IN0 is 1529
the result of ADC0_IN0 is 1534
the result of ADC0_IN0 is 1535
the result of ADC0_IN0 is 1534
the result of ADC0_IN0 is 1535
the result of ADC0_IN0 is 1530
the result of ADC0_IN0 is 1534
```

5.9. DAC_Output_Voltage_Value

5.9.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use DAC to output voltage on DAC0_OUT0 output

5.9.1. DEMO running result

Download the program <09_DAC_Output_Voltage_Value> to the EVAL board and run.

Firstly, all the LEDs will turn on and turn off for test. And then the digital value 0x7FF0, which should be 1.65V ($V_{REF}/2$), would be output on PD4.

The voltage on PD4 can be observed through the oscilloscope.

5.10. Comparator_Obtain_Brightness

5.10.1. DEMO purpose

This Demo includes the following functions of GD32 MCU:

- Learn to use comparator output compare result

The comparator has two inputs, in this demo, one input is PC1, and the other one is the DAC output voltage. Compare the two input voltages, the output is a high or low level, and the LED2 will performs the corresponding action.

5.10.2. DEMO running result

Download the program <10_Comparator_Obtain_Brightness> to the EVAL board, comparing two input voltage, if output level is high, LED2 is on, otherwise LED2 is off.

5.11. I2C_EEPROM

5.11.1. 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

5.11.2. DEMO running result

Download the program <11_I2C_EEPROM> to the EVAL board and run. Ensure that JP1 and JP19 are connected. Connect serial cable to UART, 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 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.

```
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!
```

5.12. QSPI_FLASH

5.12.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use the Quad-SPI 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 COM of development board, set the baud rate of HyperTerminal software to 115200, 8 bits' data bit, 1 bit stop bit.

Download the program <12_QSPI_Flash> to the EVAL board, ensure JP12 and JP18 are connected, 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.

```
#####
GD32M531R-EVAL System is Starting up...
GD32M531R-EVAL Flash:256K
GD32M531R-EVAL The CPU Unique Device ID:[FFFF0048-70FF17-32450CF0]
GD32M531R-EVAL SPI Flash:GD25Q16 configured...

The Flash_ID:0xC84015

Write to tx_buffer:

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

Read from rx_buffer:

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

SPI-GD25Q16 Test Passed!
```

5.13. SPI_TFT_LCD_Driver

5.13.1. DEMO purpose

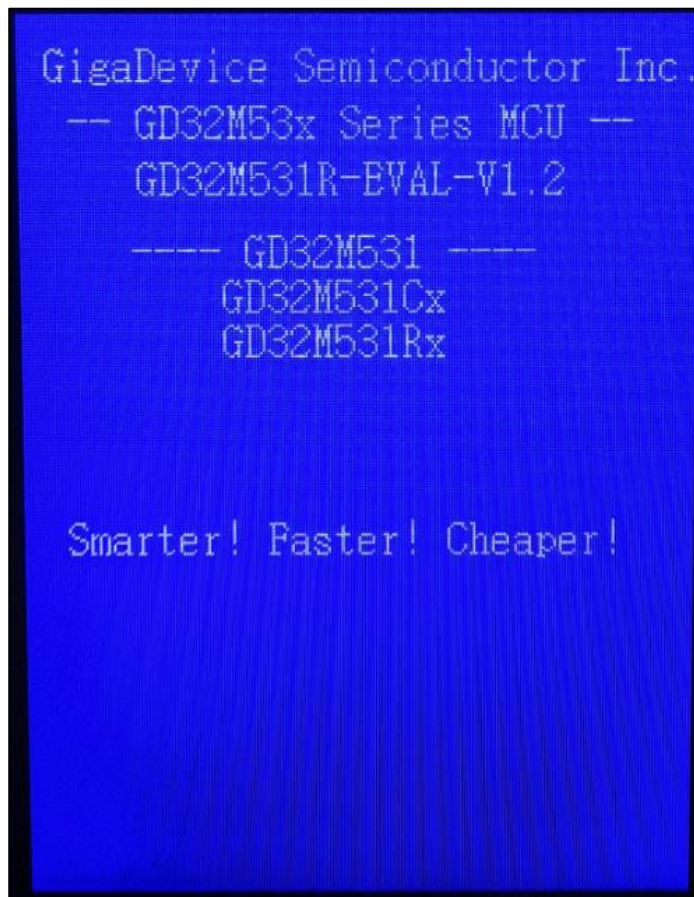
This demo includes the following functions of GD32 MCU:

- Learn how to use SPI to drive TFT LCD screen and display

GD32M531R-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.

5.13.2. DEMO running result

Download the program <13_SPI_TFT_LCD_Driver> 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.



5.14. 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 UART

5.14.2. DEMO running result

Download the program <14_RCU_Clock_Out> to the EVAL board and run, and ensure JP18 is connected. Connect serial cable to UART, 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 User 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 PF12 pin.

Information via a serial port output as following:

```
/===== Gigadevice Clock output Demo =====/
press USER key to select clock output source
CK_OUT0: system clock divided by 16
CK_OUT0: PLL clock / 8 divided by 4
CK_OUT0: HXTAL clock
CK_OUT0: IRC32M clock divided by 2
CK_OUT0: IRC32K clock
CK_OUT0: system clock divided by 16
```

5.15. PMU_Sleep_Wakeup

5.15.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use the UART 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, and ensure JP18 is connected, connect serial cable to UART. After power-on, all the LEDs are off. The MCU will enter sleep mode and the software stop running. When the UART 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. TIMER_Breath_LED

5.16.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use TIMER output PWM wave

- Learn to update channel value

5.16.2. DEMO running result

Download the program <16_TIMER_Breath_LED> to the board and run. 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.

5.17. CAN_Network

5.17.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use the CAN0 communication between two boards

GD32M531R-EVAL board integrates CAN (controller area network) bus controller. It is a common industrial control bus. The CAN bus controller follows the 2.0A and 2.0B bus protocols. This routine demonstrates the communication between two boards through CAN.

5.17.2. DEMO running result

This example is tested with two GDM531R-EVAL boards. JP19 must be connected. Connect L pin to L pin and H pin to H pin of JP20 on the boards for sending and receiving frames. Download the program <17_CAN_Network> to the two EVAL boards.

If the print function is needed, make sure that there are no jumper caps connected to JP1 and JP12. Then connect PF10 to pin 1 of JP18 and PF9 to pin 3 of JP18 using DuPont wire. Meanwhile, modify the definition of the EVAL_COM macro in the gd32m53x_eval.h file to EVAL_COM1. Connect CN3 to the PC.

Firstly, the COM sends “please press the User key to transmit data!” to the HyperTerminal. The frames are sent and the transmit data are printed by pressing User 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 User key to transmit data!  
  
can transmit data: a0 a1 a2 a3 a4 a5 a6 a7  
can receive data: a0 a1 a2 a3 a4 a5 a6 a7
```


6. Revision history

Table 6-1. Revision history

Revision No.	Description	Date
1.0	Initial Release	Feb.28, 2026

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 according to the laws of the People's Republic of China and other applicable laws. The Company reserves all rights under such laws and no Intellectual Property Rights are transferred (either wholly or partially) or licensed by the Company (either expressly or impliedly) herein. 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.

To the maximum extent permitted by applicable law, the Company makes no representations or warranties of any kind, express or implied, with regard to the merchantability and the fitness for a particular purpose of the Product, nor does the Company assume any liability arising out of the application or use of any Product. Any information provided in this document is provided only for reference purposes. It is the sole responsibility of the user of this document to determine whether the Product is suitable and fit for its applications and products planned, and properly design, program, and test the functionality and safety of its applications and products planned using the Product. The Product is designed, developed, and/or manufactured for ordinary business, industrial, personal, and/or household applications only, and the Product is not designed or intended for use in (i) safety critical applications such as weapons systems, nuclear facilities, atomic energy controller, combustion controller, aeronautic or aerospace applications, traffic signal instruments, pollution control or hazardous substance management; (ii) life-support systems, other medical equipment or systems (including life support equipment and surgical implants); (iii) automotive applications or environments, including but not limited to applications for active and passive safety of automobiles (regardless of front market or aftermarket), for example, EPS, braking, ADAS (camera/fusion), EMS, TCU, BMS, BSG, TPMS, Airbag, Suspension, DMS, ICMS, Domain, ESC, DCDC, e-clutch, advanced-lighting, etc.. Automobile herein means a vehicle propelled by a self-contained motor, engine or the like, such as, without limitation, cars, trucks, motorcycles, electric cars, and other transportation devices; and/or (iv) other uses where the failure of the device or the Product can reasonably be expected to result in personal injury, death, or severe property or environmental damage (collectively "Unintended Uses"). Customers shall take any and all actions to ensure the Product meets the applicable laws and regulations. The Company is not liable for, in whole or in part, and customers shall hereby release the Company as well as its suppliers and/or distributors from, any claim, damage, or other liability arising from or related to all Unintended Uses of the Product. Customers shall indemnify and hold the Company, and its officers, employees, subsidiaries, affiliates as well as its 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 Product.

Information in this document is provided solely in connection with the Product. The Company reserves the right to make changes, corrections, modifications or improvements to this document and the Product described herein at any time without notice. The Company shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. Information in this document supersedes and replaces information previously supplied in any prior versions of this document.