GigaDevice Semiconductor Inc.

GD32207I-EVAL Evaluation Board

User Manual

Table of Contents

List of Figures

List of Tables

1. Introduction

GD32207I-EVAL evaluation board uses GD32F207IKT6 as the main controller. As a complete development platform of GD32F207xx connectivity line powered by ARM® Cortex™-M3 core, the board supports full range of peripherals. It uses Mini USB interface or AC/DC adapter as 5V power supply. JTAG, Reset, Boot, User button key, LED, CAN, I2C, I2S,DCI,SDRAM,USART, RTC, EXMC, SPI, USB_OTG, ADC, DAC, GD-Link、TLDI RGB-LCD、SDIO, Ethernet and Extension Pin are also included. This document details its hardware schematic and the relevant applications.

2. Function pin assignment

Table 2-1 Pin assignment

Function	Pin	Description
LED	PF6	LED1
	PF7	LED2
	PF8	LED3
	PF9	LED4
RESET		K1-Reset
KEY	PA0	KEY1
	PC13	KEY2
	PB14	KEY3
USB_OTG	PA9	USB_VBUS
	PA11	USB_DM
	PA12	USB_DP
	PD13	VBUS control pin
CAN	PB8	CAN1_RX
	PB9	CAN1_TX
	PB5	CAN2_RX
	PB6	CAN2_TX
I2C	PB6	I2C1_SCL
	PB7	I2C1_SDA
I2S	PB12	I2S_WS
	PB15	I2S_DIN
	PB13	I2S_CK
	PC6	I2S_MCK
	PA4	MSEL
	PA5	MCLK
	PA7	MDIN

Function	Pin	Description
	RESET	NRST
USART1	PA9	USART1_TX
	PA10	USART1_RX
EXMC	PD14	EXMC_D0
	PD15	EXMC_D1
	PD0	EXMC_D2
	PD1	EXMC_D3
	PE7	EXMC_D4
	PE8	EXMC_D5
	PE9	EXMC_D6
	PE10	EXMC_D7
	PD11	EXMC_A16
	PD12	EXMC_A17
	PD4	EXMC_NOE
	PD5	EXMC_NWE
	PD6	EXMC_NWAIT
	PD7	EXMC_NE1
SPI	PA2	SPI1_WP_IO2
	PA3	SPI1_$\overline{\text{HOLD}}$_IO3
	PA5	SPI1_SCK
	PA6	SPI1_MISO_IO1
	PA7	SPI1_MOSI_IO0
	PB1	SPIFlash_CS
ADC	PC3	ADC123_IN13
DAC	PA4	DAC_OUT1
	PA5	DAC_OUT2
SDIO	PC12	SDIO_CLK
	PD2	SDIO_CMD
	PC8	SDIO_DAT0
	PC9	SDIO_DAT1
	PC10	SDIO_DAT2
	PC11	SDIO_DAT3
DCI	PC6	DCI_D0
	PC7	DCI_D1
	PC8	DCI_D2
	PC9	DCI_D3
	PC11	DCI_D4
	PD3	DCI_D5
	PB8	DCI_D6
	PB9	DCI_D7
	PA6	DCI_PIXCLK

Function	Pin	Description
	PA8	DCI_XCLK
	PG9	DCI_VSYNC
	PH8	DCI_HSYNC
	PB10	DCI_I2C2_SCL
	PB11	DCI_I2C2_SDA
RGB_TFTLCD	PH4	LCD_R0
	Pi3	LCD_R1
	PC10	LCD_R2
	PH9	LCD_R3
	PH10	LCD_R4
	PH11	LCD_R5
	PH12	LCD_R6
	PG6	LCD_R7
	PE5	LCD_G0
	PE6	LCD_G1
	PH13	LCD_G2
	PH14	LCD_G3
	PH15	LCD_G4
	Pi0	LCD_G5
	Pi1	LCD_G6
	Pi2	LCD_G7
	PE4	LCD_B0
	PG12	LCD_B1
	PG10	LCD_B2
	PG11	LCD_B3
	Pi4	LCD_B4
	Pi5	LCD_B5
	Pi6	LCD_B6
	Pi7	LCD_B7
	PG7	LCD_CLK
	Pi10	LCD_HSYNC
	PF10	LCD_DE
	Pi9	LCD_VSYNC
	PH7	LCD_Touch_Busy
	PH6	LCD_PWM_BackLight
	PA5	LCD_SPI1_SCK
	PA7	LCD_SPI1_MOSI
	PH5	LCD_Touch_PENIRQ
	PG3	LCD_SPI1_NSS
	PA6	LCD_SPI1_MISO
SDARM	PD14	EXMC_D0

Function	Pin	Description
	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
	PF0	EXMC_A0
	PF1	EXMC_A1
	PF2	EXMC_A2
	PF3	EXMC_A3
	PF4	EXMC_A4
	PF5	EXMC_A5
	PF12	EXMC_A6
	PF13	EXMC_A7
	PF14	EXMC_A8
	PF15	EXMC_A9
	PG0	EXMC_A10
	PG1	EXMC_A11
	PG2	EXMC_A12
	PE0	EXMC_NBL0
	PE1	EXMC_NBL1
	PH2	EXMC_SDCKE0
	PG4	EXMC_BA0
	PG5	EXMC_BA1
	PG8	EXMC_SDCLK
	PG15	EXMC_SDNCAS
	PF11	EXMC_SDNRAS
	PH3	EXMC_SDNE0
	PC0	EXMC_SDNWE
Ethernet	PB11	RMII_TX_EN
	PB12	RMII_TXD0
	PB13	RMII_TXD1

Function	Pin	Description
	PC4	RMII_RXD0
	PC5	RMII_RXD1
	PA7	RMII_CRS_DV
	PC1	RMII_MDC
	PA2	RMII_MDIO
	PB0	RMII_INT
	PA1	RMII_REF_CLK

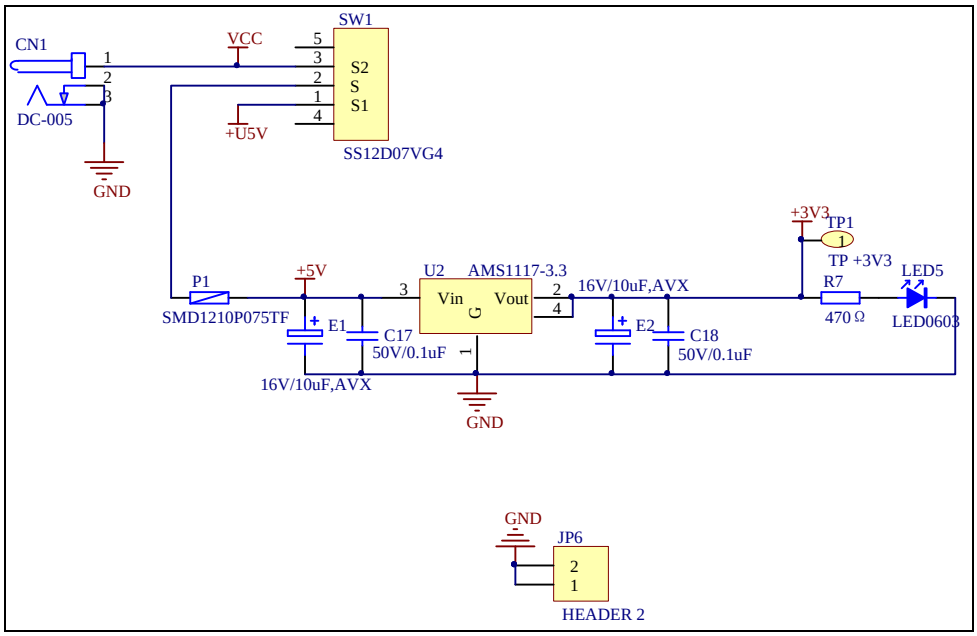
3. Getting started

The EVAL Board uses Mini USB connecter or adapter to get power, the hardware system power is +3.3V. A Mini USB cable and a J-Link tool are necessary to down programs. Select the correct boot mode and then power on, the LED5 will turn on, which indicates the power supply is ready.

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

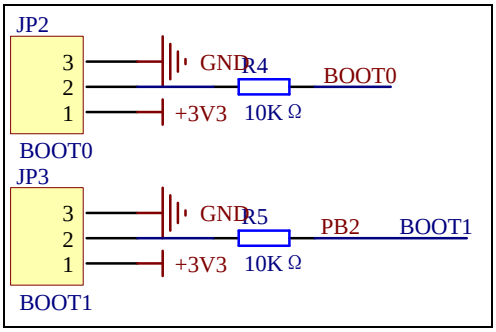
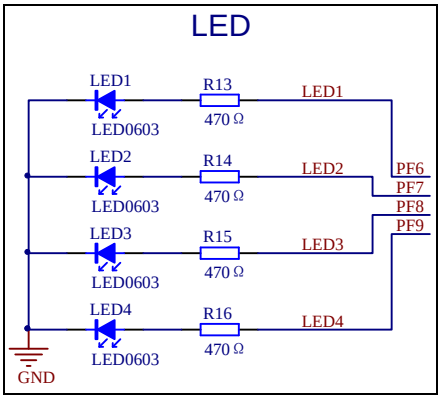


Table 4-1 Boot configuration

BOOT1	BOOT0	Boot Mode
Any	2-3	User memory
2-3	1-2	System memory
1-2	1-2	SRAM memory

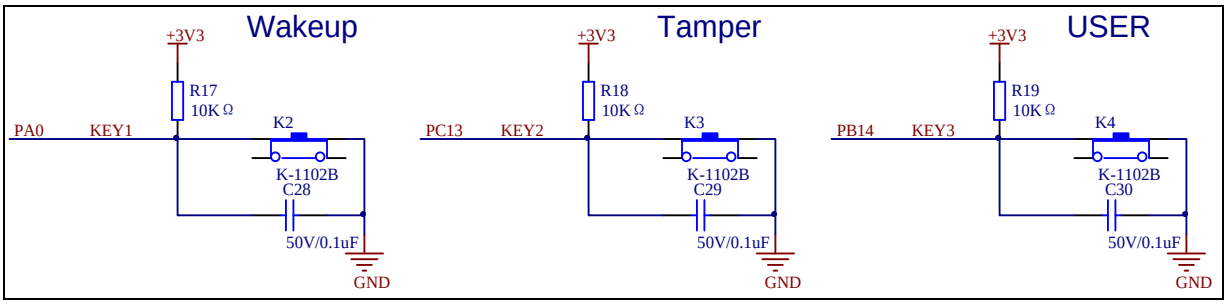
4.3. LED

Figure 4-3 Schematic diagram of LED function



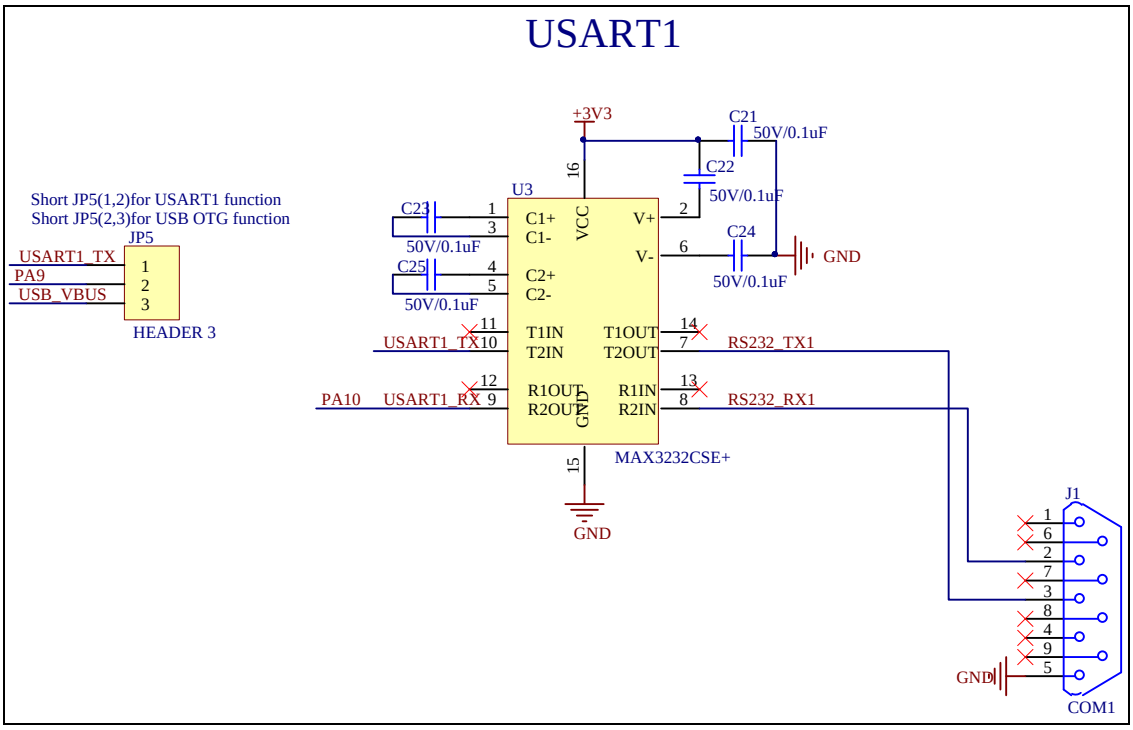
4.4. Key

Figure 4-4 Schematic diagram of Key function



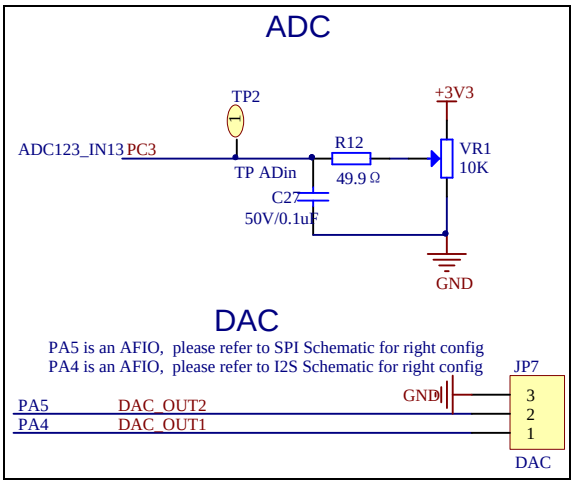
4.5. USART1

Figure 4-5 Schematic diagram of USART1 function



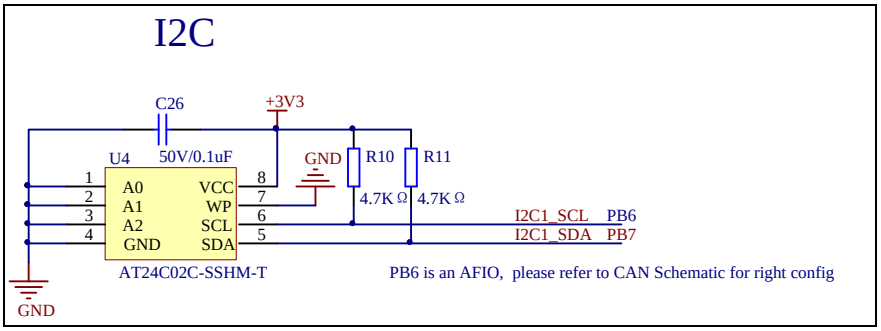
4.6. ADC/DAC

Figure 4-6 Schematic diagram of ADC/DAC function



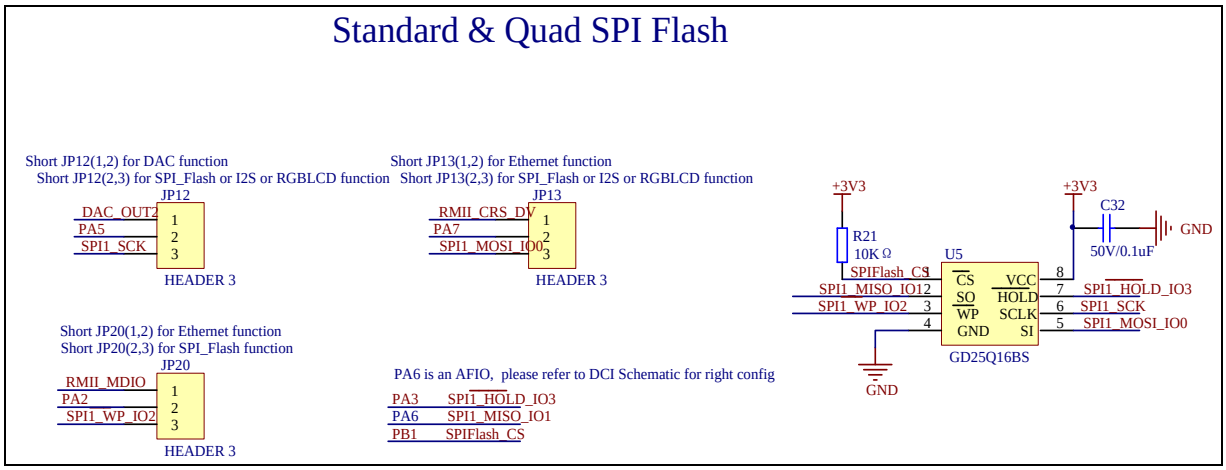
4.7. I2C

Figure 4-7 Schematic diagram of I2C function



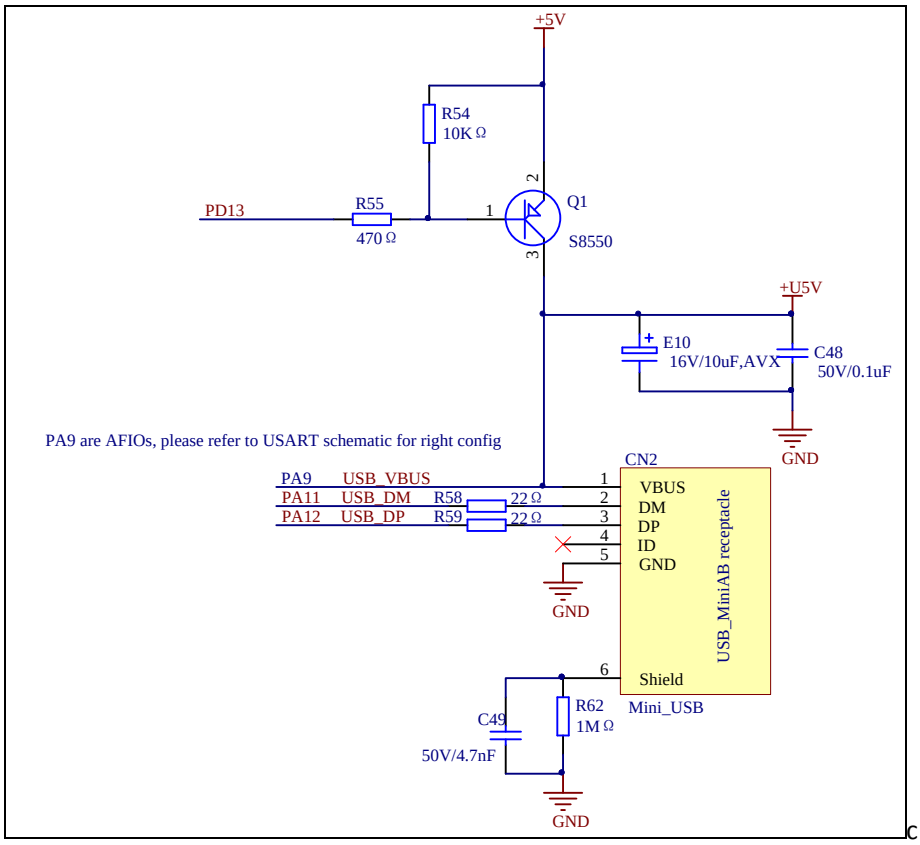
4.8. SPI-Serial Flash

Figure 4-8 Schematic diagram of SPI-Serial Flash function



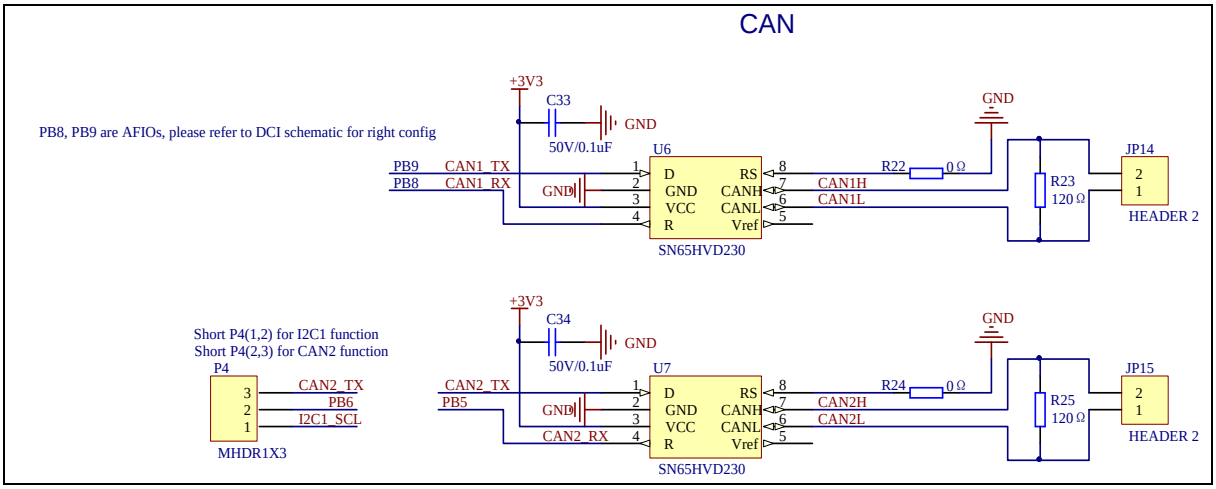
4.9. USB

Figure 4-9 Schematic diagram of USB function



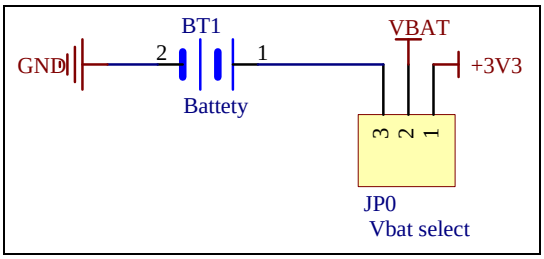
4.10. CAN

Figure 4-10 Schematic diagram of CAN function



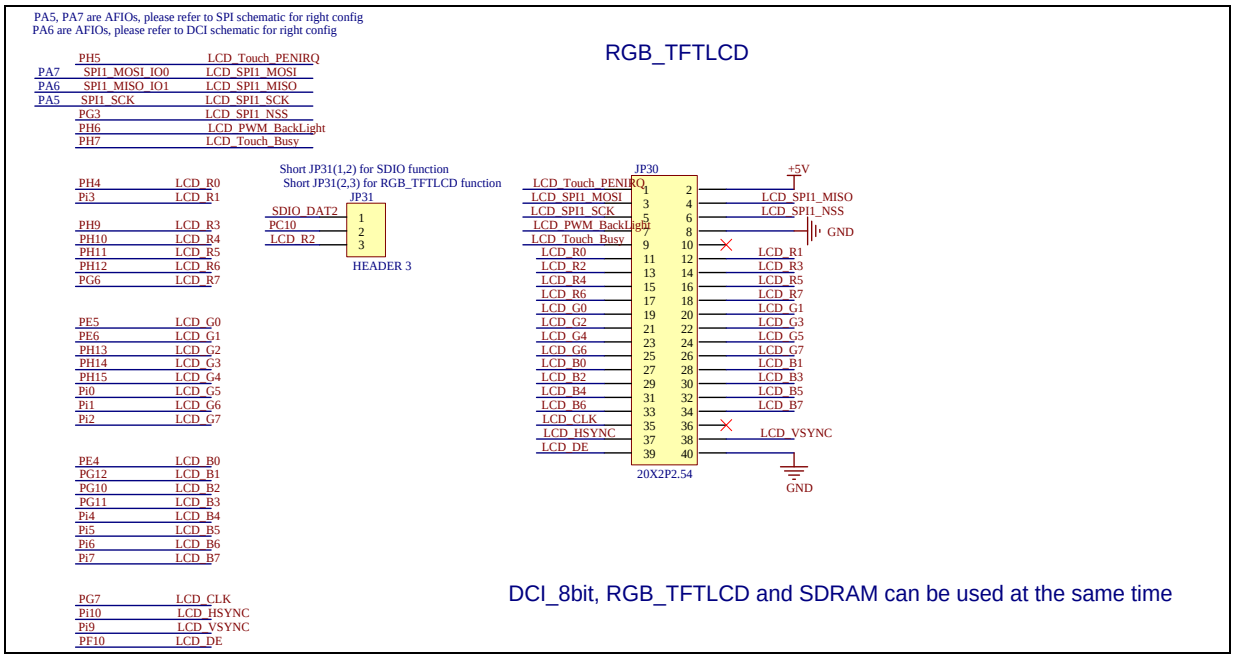
4.11. RTC

Figure 4-11 Schematic diagram of RTC function



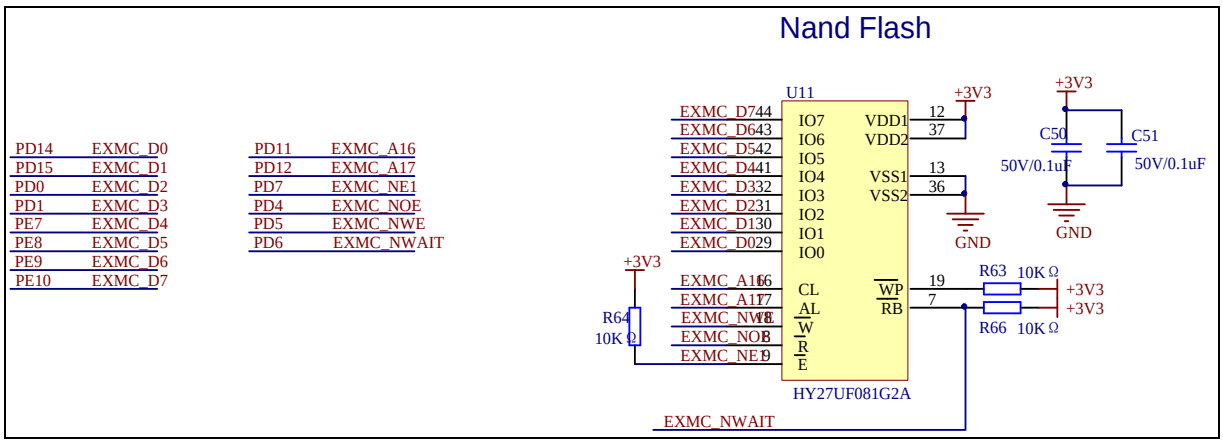
4.12. TLDI RGB-LCD

Figure 4-12 Schematic diagram of TLDI RGB-LCD function



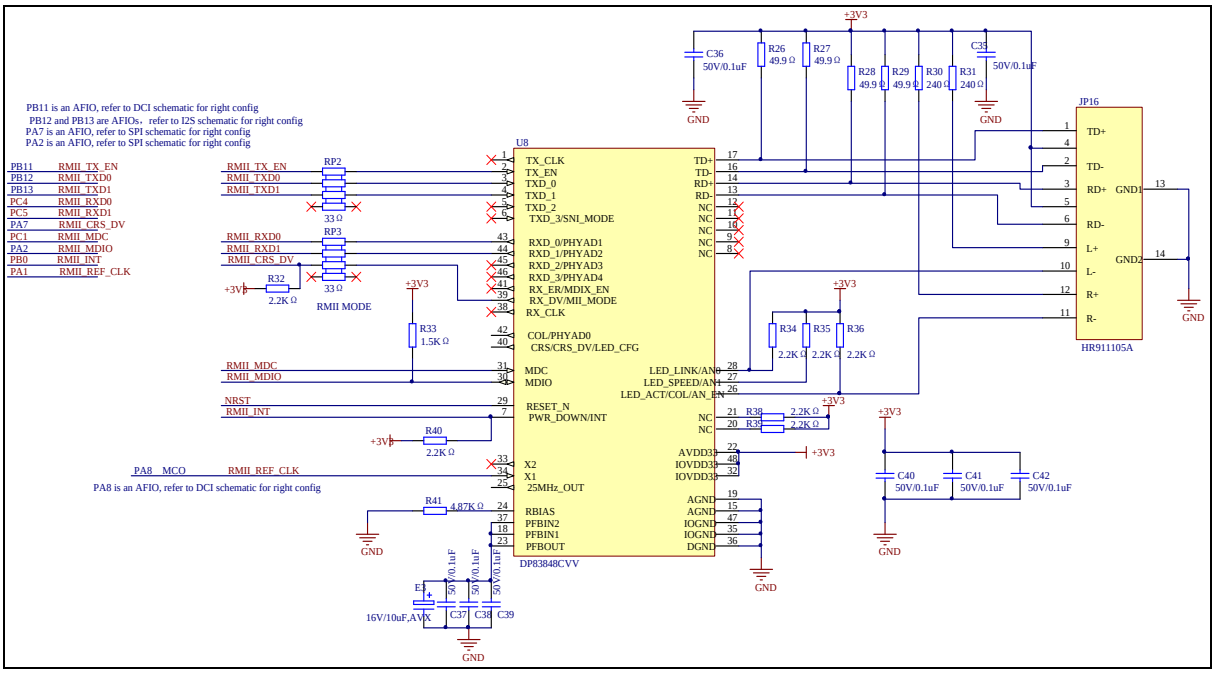
4.13. EXMC-NAND Flash

Figure 4-13 Schematic diagram of EXMC-NAND Flash function



4.14. Ethernet

Figure 4-14 Schematic diagram of Ethernet



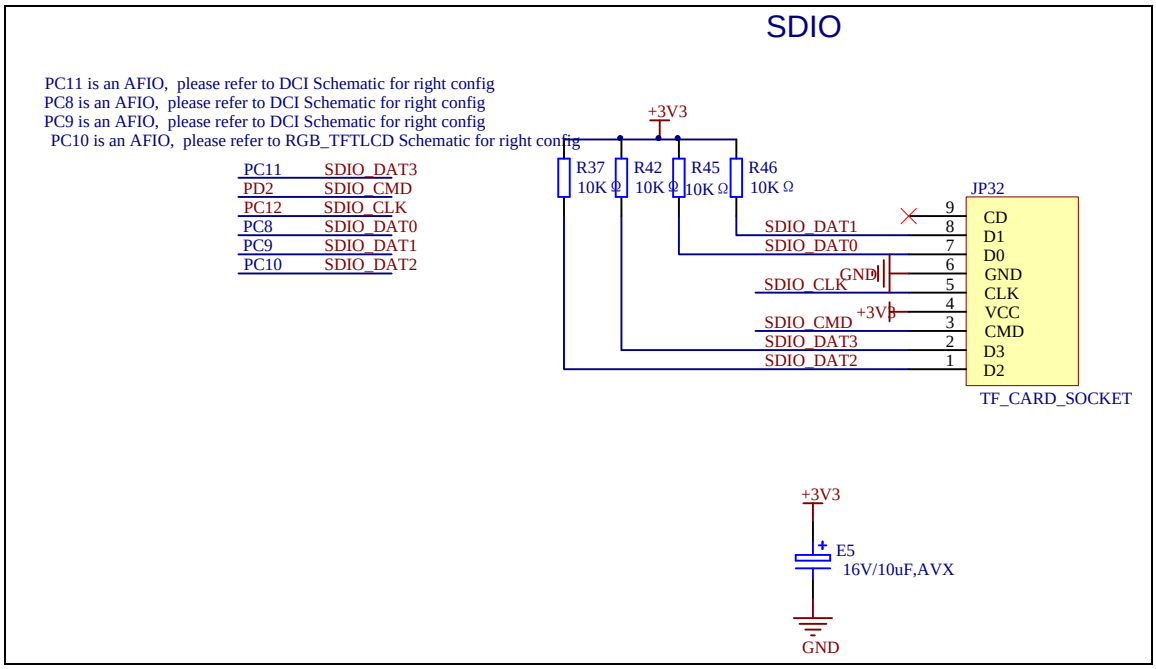
4.15. GD-Link

Figure 4-15 Schematic diagram of GD-Link

L_TDI	PA15
L_TMS/IO	PA13
L_TCK/CLK	PA14
L_TDO/SWO	PB3
L_TReset	NRST

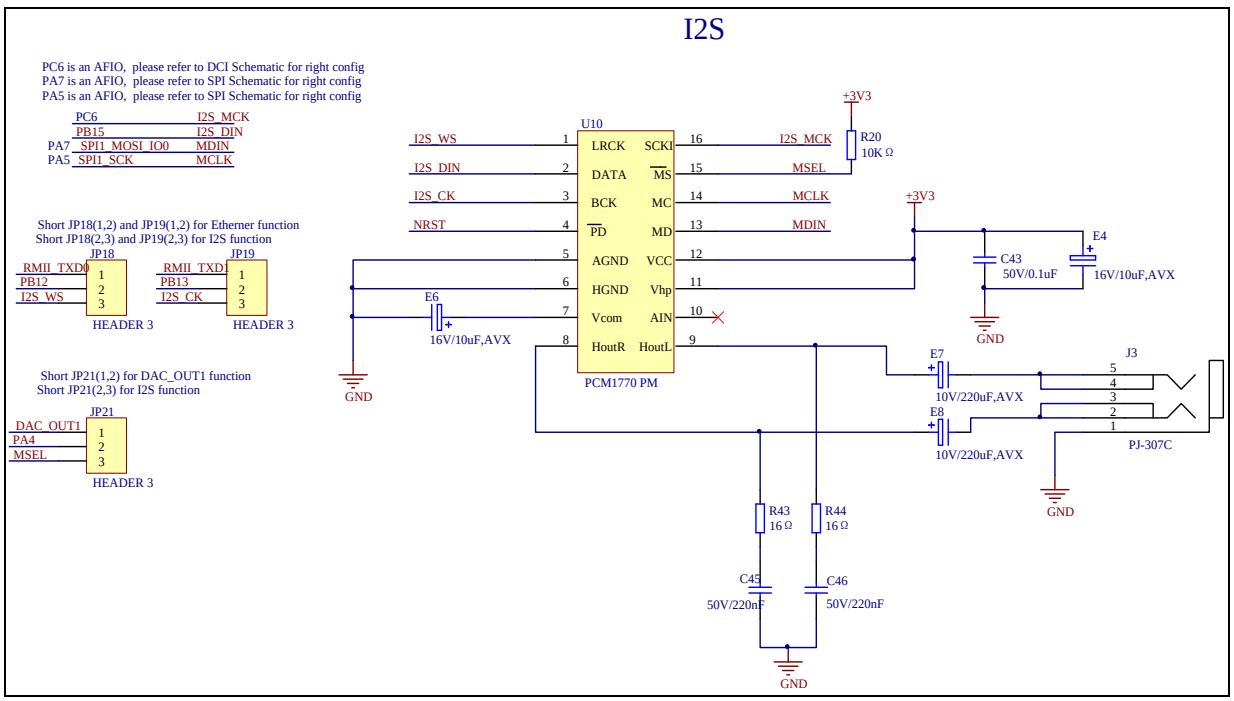
4.16. SDIO

Figure 4-16 Schematic diagram of SDIO



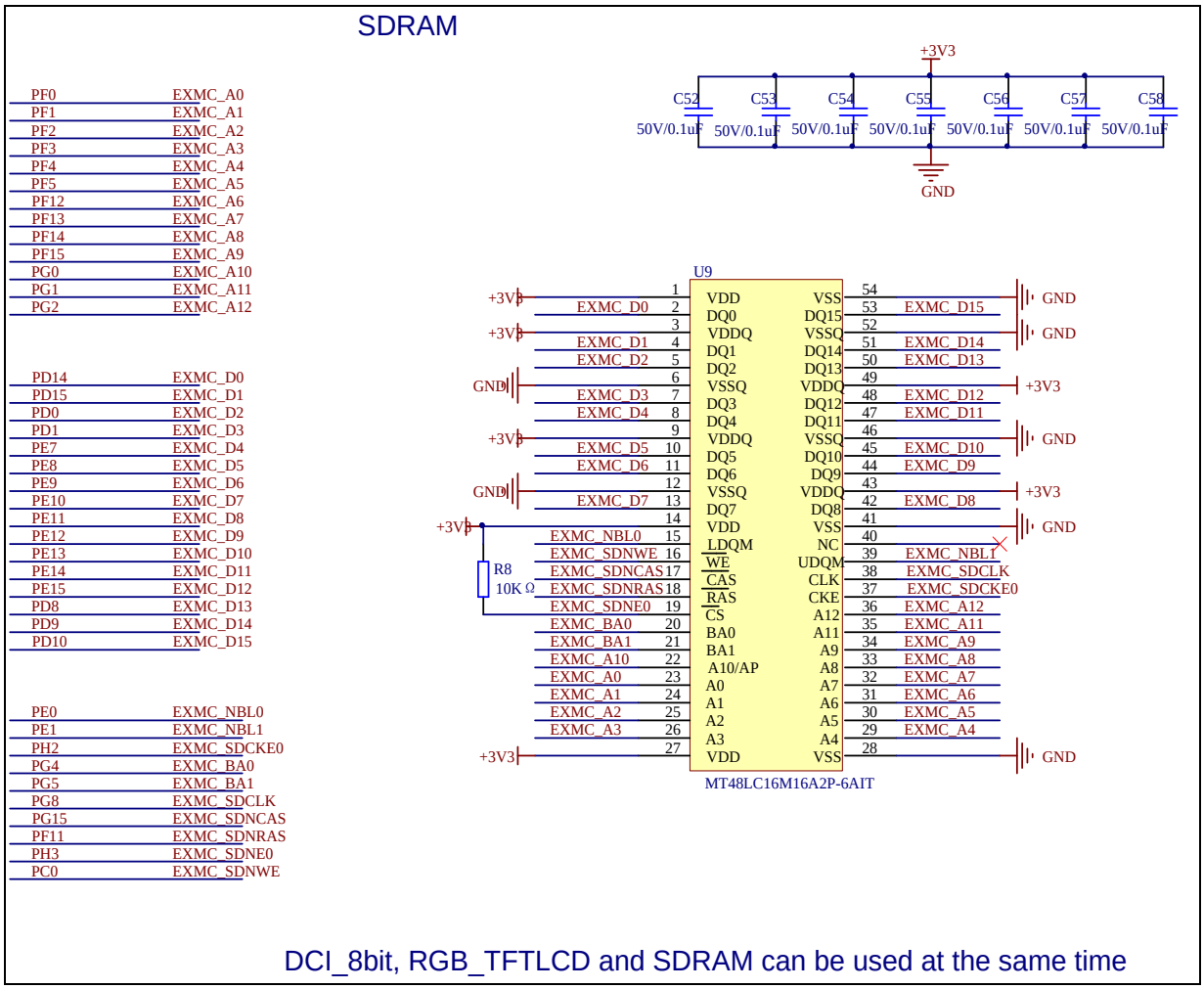
4.17. I2S

Figure 4-17 Schematic diagram of I2S



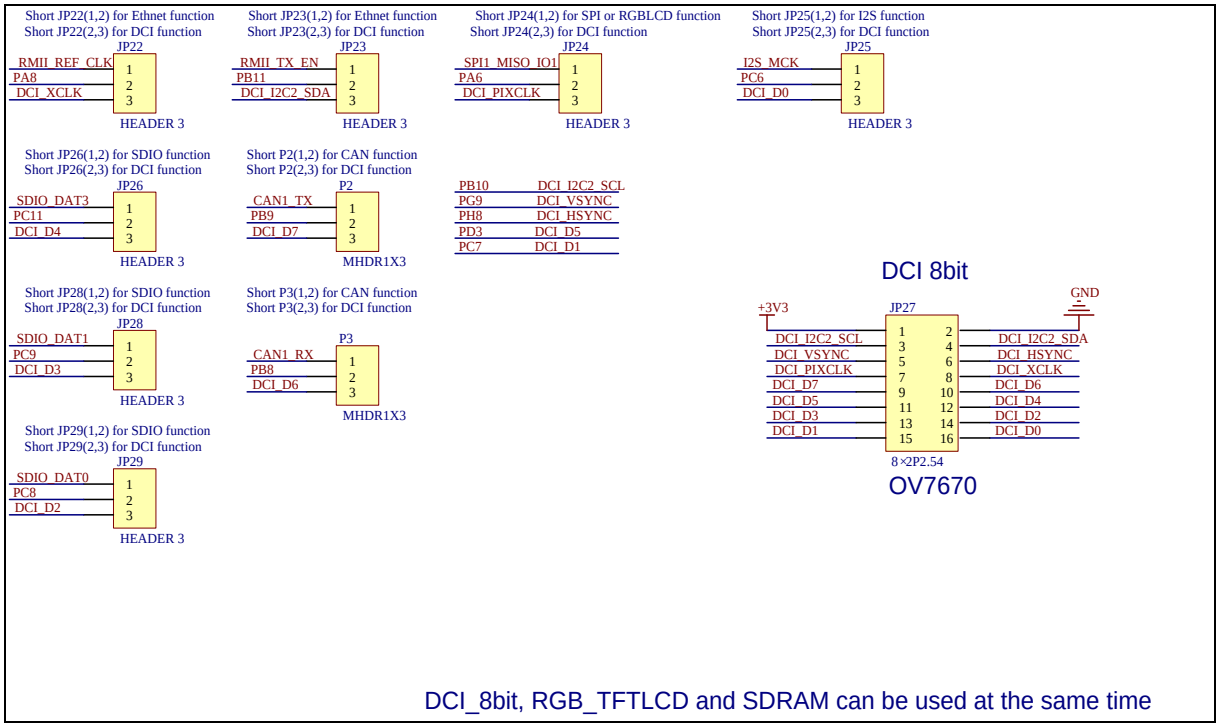
4.18. SDRAM

Figure 4-18 Schematic diagram of SDRAM



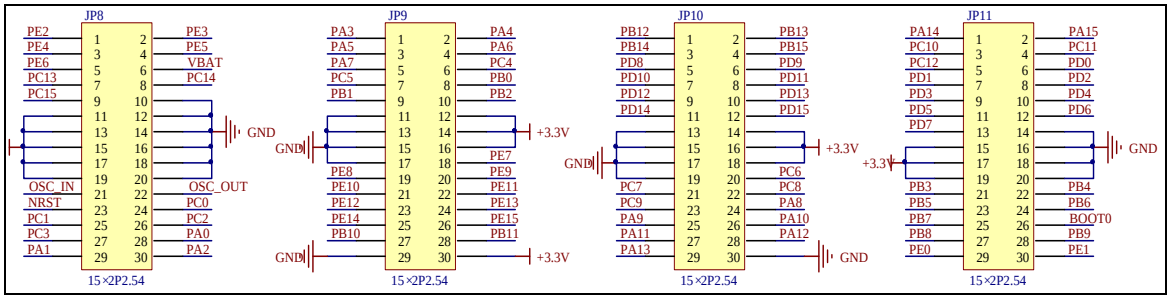
4.19. DCI

Figure 4-19 Schematic diagram of DCI



4.20. Extension

Figure 4-20 Schematic diagram of Extension Pin



5. Routine use guide

5.1. GPIO running light

5.1.1. DEMO Purpose

GD32207I-EVAL-V1.0 development board has four LED lights:LED1，LED2，LED3 and LED4.This DEMO will show you how to control the effect of four LED lights flashing in implementation of running water. Also the DEMO will tell you the demonstration of GD32F20X internal GPIO configuration method, and shows you GPIO output characteristic. The key is how to control the IO output different levels of this DEMO, and then the LED flashing, in this DEMO, you will master the basic use of GD32F20X GPIO.

GD32F20X GPIO main features:

- Input/output direction control
- Each pin weak pull-up/pull-down function
- Output push-pull/open drain enable control
- Output set/reset control
- External interrupt with programmable trigger edge – using EXTI configuration registers
- Analog input/output configurations
- Alternate function input/output configurations
- Port configuration lock

5.1.2. DEMO Principle

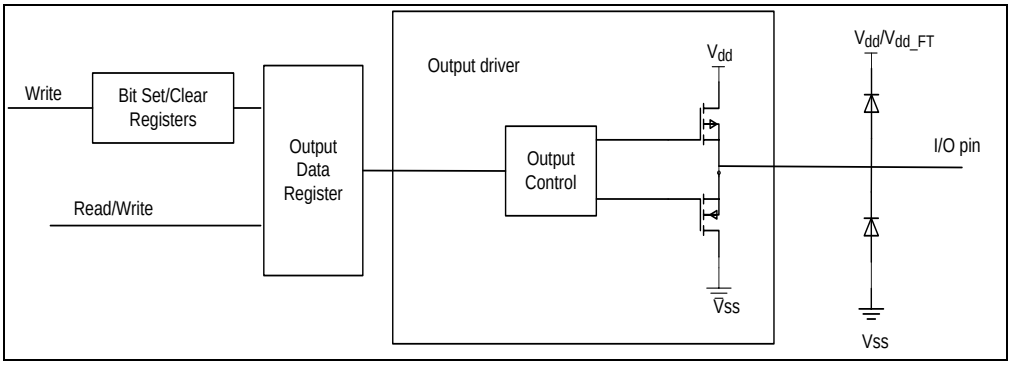
GD32F20X GPIO port can be configured into 8 modes by software:

- Analog input
- Floating input
- Pull-up input
- Pull-down input
- Open-drain output
- Push-pull output
- Alternate Open-drain output
- Alternate Push-pull output

Corresponding to the GD32 library file is defined as follows：

```
typedef enum
{
    GPIO_MODE_AIN = 0x0,
    GPIO_MODE_IN_FLOATING = 0x04,
    GPIO_MODE_IPD = 0x28,
    GPIO_MODE_IPU = 0x48,
    GPIO_MODE_OUT_OD = 0x14,
    GPIO_MODE_OUT_PP = 0X10,
    GPIO_MODE_AF_OD = 0X1C,
    GPIO_MODE_AF_PP = 0X18
}GPIO_ModePara;
```

Figure 5-1 GPIO output driver block diagram



Note: V_{dd_FT} is dedicated for five-volt tolerant I/Os and is different from V_{dd}

When GPIO pin is configured as output:

- The Schmitt Trigger Input is activated.
- The weak pull-up and pull-down resistors are disabled.
- The Output Buffer is enabled:

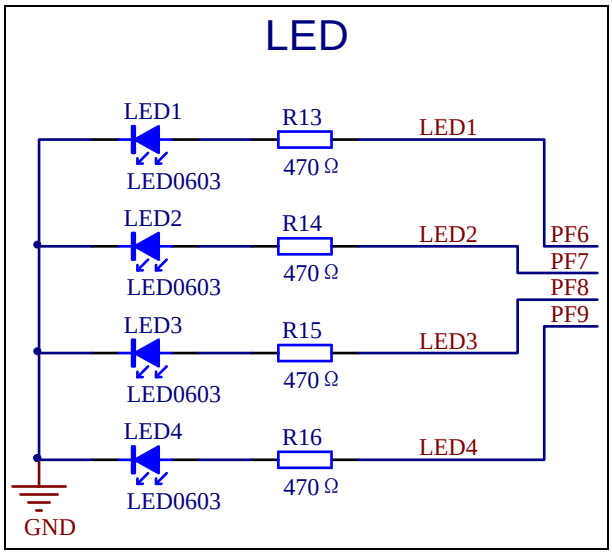
  Open Drain Mode: a “0” in the output register activates the N-MOS while a “1” in the Output register leaves the port in Hi-Z.

  Push-Pull Mode: a “0” in the output register activates the N-MOS while a “1” in the Output register activates the P-MOS.

- Read the Data Output Register gets the last written value in Push-Pull mode
- Read the Data Input Register gets the I/O state in open drain mode

The GPIO is configured as a push-pull output, and in order to realize the effect of light water, the IO output is controlled by the GPIO setting or resetting function. When the port is set , the GPIO output will be high, and the LED will light，When the port is reset, the GPIO output will be low, and the LED will be off.

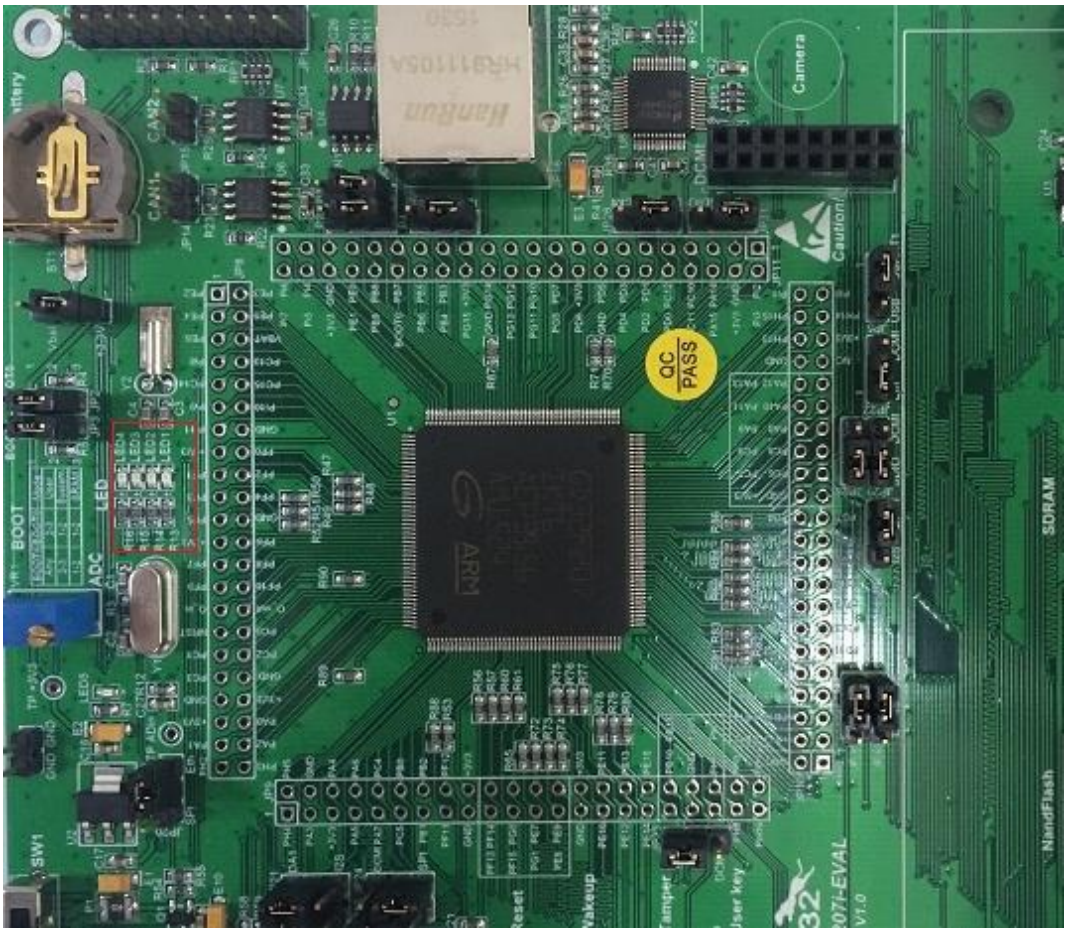
Figure 5-2 Schematic diagram of LED



The above shows the LED circuit diagram, where the resistance of the LED series, in series with the LED is mainly used to limit the current to avoid damaging the LED and the GPIO port.

5.1.3. DEMO Implementation Result

Download the program to the development board, the implementation of the results is the development board LED1 will be the light first, and then off, and LED2 be light So LED1, LED2, LED3, LED4 are successfully be light in a flow. One of the LED switch to another every 500ms to achieve the function of running light water. The position of the four LEDs are shown in the red region of the graph.



5.2. GPIO key polling mode

5.2.1. DEMO Purpose

GD32207I-EVAL-V1.0 development board has four keys: K1 (Reset), K2 (Wakeup), K3 (Tamper), K4 (User Key).In this Demo, only K3 (Temper) is used. This Demo will introduce the GD32F20X GPIO Input function, and demonstrate the GD32F20X internal GPIO configuration method, show the GPIO input characteristics.

GD32F20X GPIO main features:

- Input/output direction control
- Each pin weak pull-up/pull-down function
- Output push-pull/open drain enable control
- Output set/reset control
- External interrupt with programmable trigger edge – using EXTI configuration registers

- Analog input/output configurations
- Alternate function input/output configurations
- Port configuration lock

5.2.2. DEMO Principle

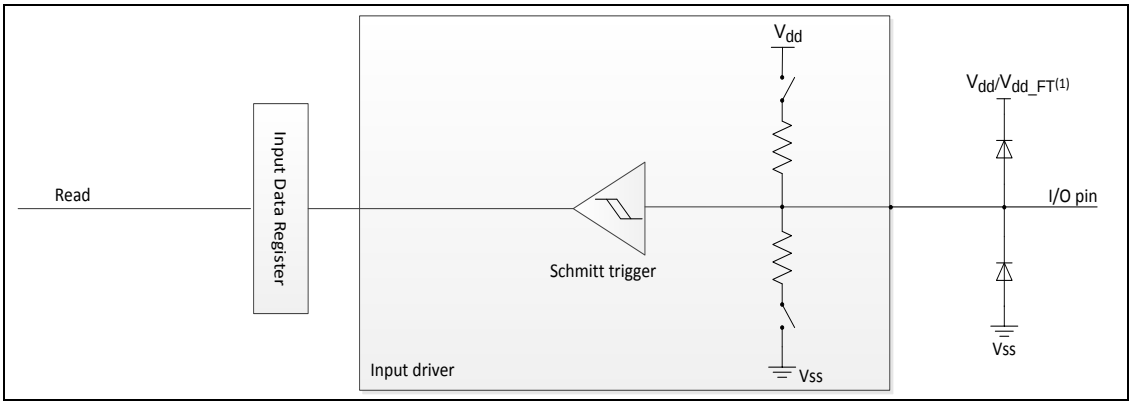
GD32F20X GPIO port can be configured into 8 modes of software:

- Analog input
- Floating input
- Pull-up input
- Pull-down input
- Open-drain output
- Push-pull output
- Alternate Open-drain output
- Alternate Push-pull output

Corresponding to the GD32 library file is defined as follows：

```
typedef enum
{
GPIO_MODE_AIN = 0x0,
GPIO_MODE_IN_FLOATING = 0x04,
GPIO_MODE_IPD = 0x28,
GPIO_MODE_IPU = 0x48,
GPIO_MODE_OUT_OD = 0x14,
GPIO_MODE_OUT_PP = 0X10,
GPIO_MODE_AF_OD = 0X1C,
GPIO_MODE_AF_PP = 0X18
}GPIO_ModePara;
```

Figure 5-3 GPIO input drive block diagram

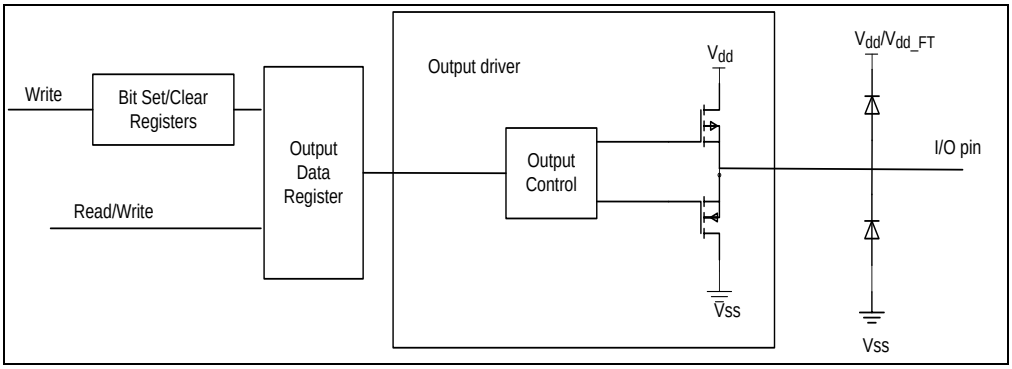


Note: V_{dd_FT} is dedicated for five-volt tolerant I/Os and is different from V_{dd}

When GPIO pin is configured as Input:

- The Schmitt Trigger Input is activated
- The weak pull-up and pull-down resistors could be chosen
- The data present on the I/O pad is sampled into the data input register every APB2 clock cycle
- The Output Buffer is disabled

Figure 5-4 GPIO output driver block diagram



Note: V_{dd_FT} is dedicated for five-volt tolerant I/Os and is different from V_{dd}

When GPIO pin is configured as output:

- The Schmitt Trigger Input is activated.
- The weak pull-up and pull-down resistors are disabled.
- The Output Buffer is enabled:

  Open Drain Mode: a “0” in the output register activates the N-MOS while a “1” in the Output register leaves the port in Hi-Z.

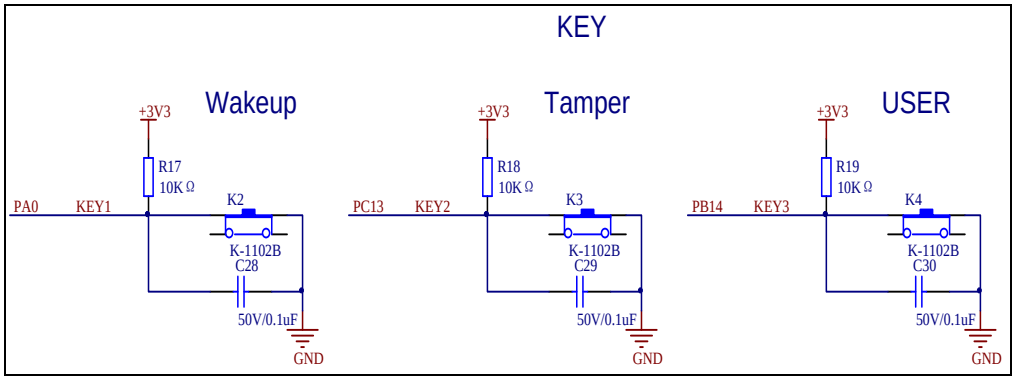
  Push-Pull Mode: a “0” in the output register activates the N-MOS while a “1” in the

Output register activates the P-MOS.

- Read the Data Output Register gets the last written value in Push-Pull mode
- Read the Data Input Register gets the I/O state in open drain mode

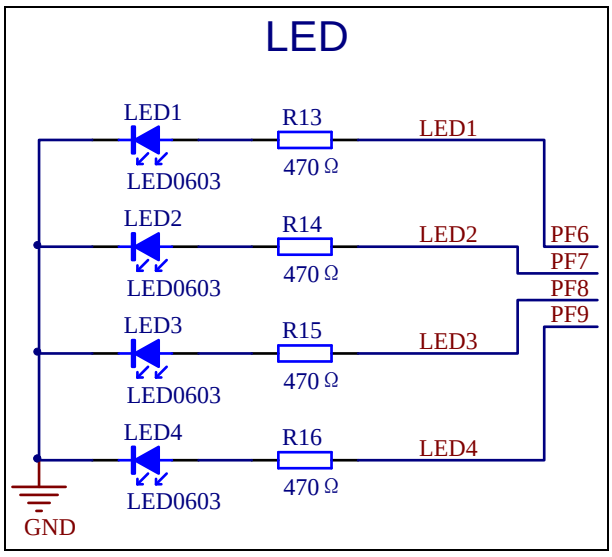
In this DEMO floating input mode is adopted, and external pull up resistance is used, Refer to the output driver block diagram, the GPIO is configured as a floating input. When the key is pressed down, read the data input register. if the data is zero, then delay some time and read pin data input register again，if the data is one，It means that the key pressing is not successful, it is still be zero, then the key pressing is successful, and the LED2 toggles.

Figure 5-5 Schematic diagram of KEY



The GPIO is configured as a push-pull output, and in order to realize the effect of light water, the IO output is controlled by the GPIO setting or resetting function. When the port is set , the GPIO output will be high, and the LED will light，When the port is reset, the GPIO output will be low, and the LED will be off.

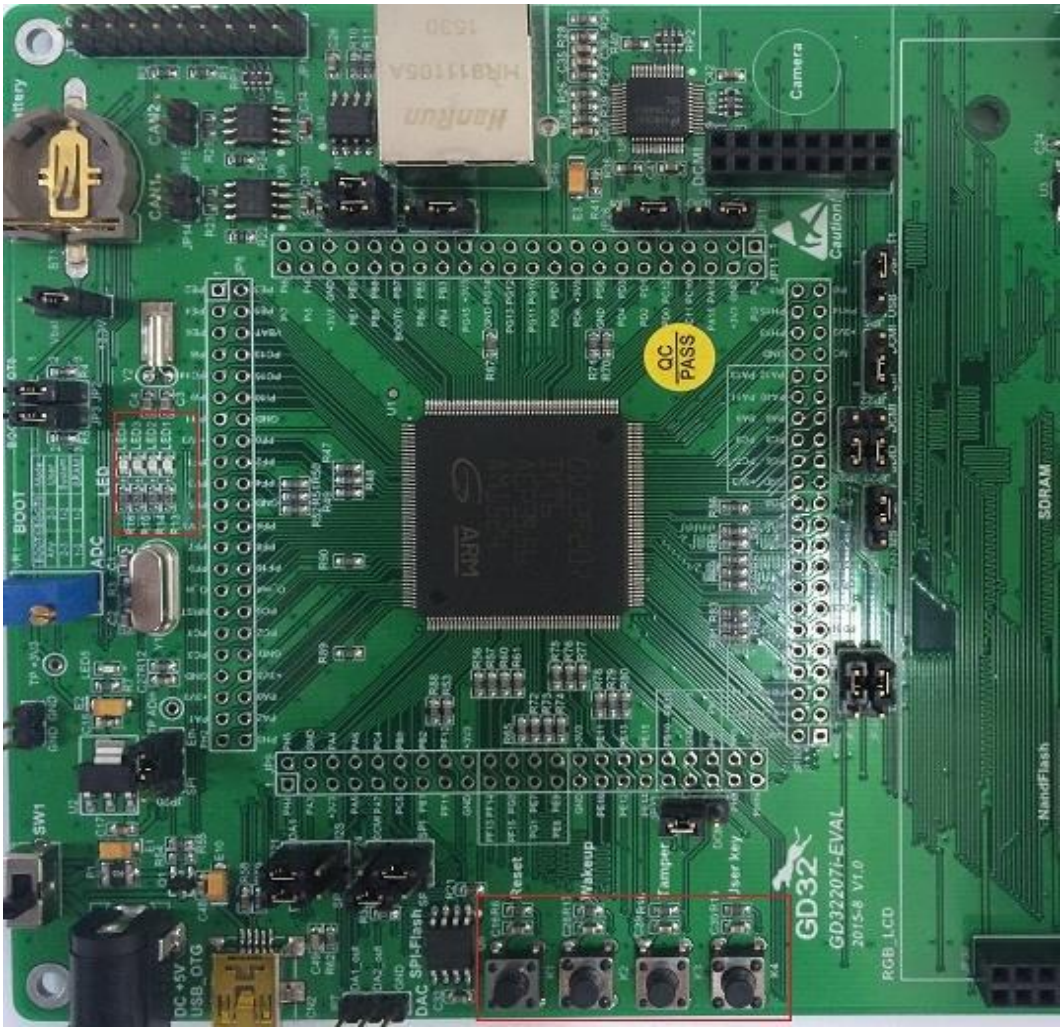
Figure 5-6 Schematic diagram of LED



The above shows the LED circuit diagram, where the resistance of the LED series, in series with the LED is mainly used to limit the current to avoid reducing LED use age.

5.2.3. DEMO Implementation Result

Download the program to the development board, the phenomenon is that the K3 be pressed down the first time in the development board, and LED1 turns on. and K3 be pressed down the second time, LED1 turns off. The position of the four LEDs and K3 is shown in the red region of the graph.



5.3. GPIO key interrupt mode

5.3.1. DEMO Purpose

GD32207I-EVAL-V1.0 development board has four keys: K1 (Reset), K2 (Wakeup), K3 (Tamper), K4 (User Key).In this DEMO, only K3 (Temper) is used. This DEMO will introduce the external interrupt characteristics of GD32F20X port, demonstrate the configuration method of GD32F20X GPIO external interrupt, and show the GPIO external interrupt characteristics. For EXTI principle learning.

GD32F20X GPIO main features:

- Input/output direction control
- Each pin weak pull-up/pull-down function
- Output push-pull/open drain enable control
- Output set/reset control
- External interrupt with programmable trigger edge – using EXTI configuration registers
- Analog input/output configurations
- Alternate function input/output configurations
- Port configuration lock

GD32F20X EXTI main features：

- Cortex-M3 system exception
- Up to 90maskable peripheral interrupts
- 4 bits interrupt priority configuration—16 priority levels
- Efficient interrupt processing
- Support exception pre-emption and tail chaining
- Wake up system from power saving mode
- 20 independent edge detectors in EXTI
- Three trigger types: rising, falling and both edges
- Software interrupt or event trigger
- Trigger sources configurable

5.3.2. DEMO Principle

GD32F20X GPIO port can be configured into 8 modes of software:

- Analog input
- Floating input
- Pull-up input
- Pull-down input
- Open-drain output
- Push-pull output
- Alternate Open-drain output

- Alternate Push-pull output

Corresponding to the GD32 library file is defined as follows：

typedef enum

{

GPIO_MODE_AIN = 0x0,

GPIO_MODE_IN_FLOATING = 0x04,

GPIO_MODE_IPD = 0x28,

GPIO_MODE_IPU = 0x48,

GPIO_MODE_OUT_OD = 0x14,

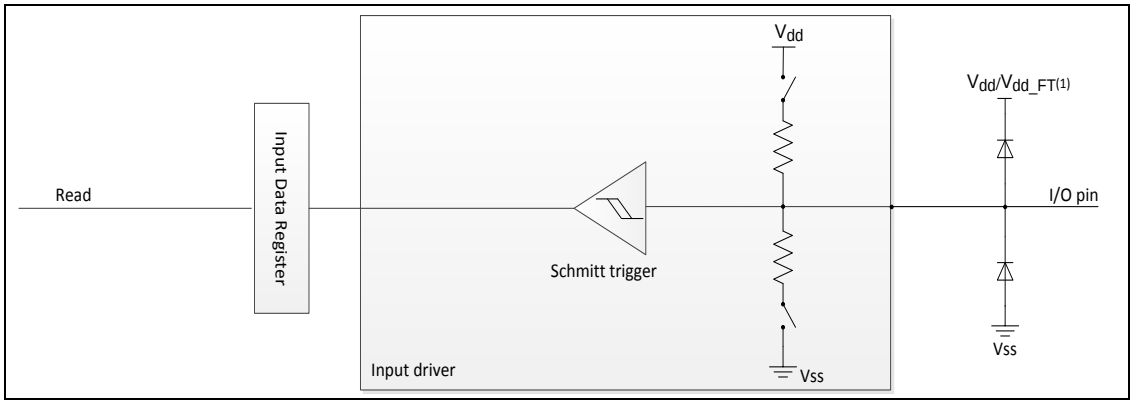
GPIO_MODE_OUT_PP = 0X10,

GPIO_MODE_AF_OD = 0X1C,

GPIO_MODE_AF_PP = 0X18

}GPIO_ModePara;

Figure 5-7 GPIO input drive block diagram

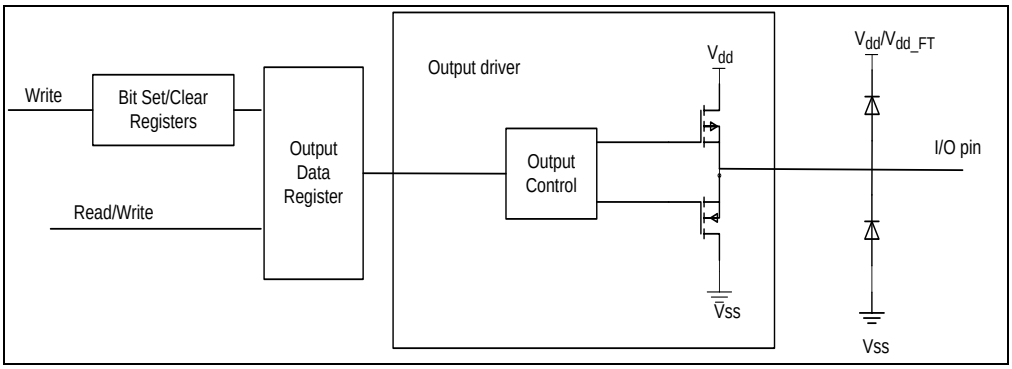


Note: V_{dd_FT} is dedicated for five-volt tolerant I/Os and is different from V_{dd}

When GPIO pin is configured as Input:

- The Schmitt Trigger Input is activated
- The weak pull-up and pull-down resistors could be chosen
- The data present on the I/O pad is sampled into the data input register every APB2 clock cycle
- The Output Buffer is disabled

Figure 5-8 GPIO output driver block diagram



Note: V_{dd_FT} is dedicated for five-volt tolerant I/Os and is different from V_{dd}

When GPIO pin is configured as output:

- The Schmitt Trigger Input is activated.
- The weak pull-up and pull-down resistors are disabled.
- The Output Buffer is enabled:

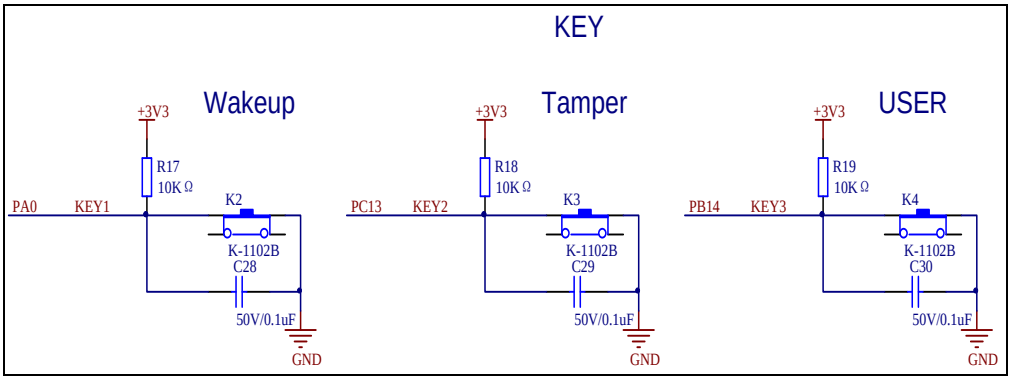
  Open Drain Mode: a “0” in the output register activates the N-MOS while a “1” in the Output register leaves the port in Hi-Z.

  Push-Pull Mode: a “0” in the output register activates the N-MOS while a “1” in the Output register activates the P-MOS.

- Read the Data Output Register gets the last written value in Push-Pull mode
- Read the Data Input Register gets the I/O state in open drain mode

In this DEMO floating input mode is adopted, and external pull up resistance is used, Refer to the output driver block diagram, The GPIO is configured to the falling edge trigged of an external interrupt. If the key (K3) is pressed down, an external interrupt will occur. In the external interrupt service function, the LED2 toggles.

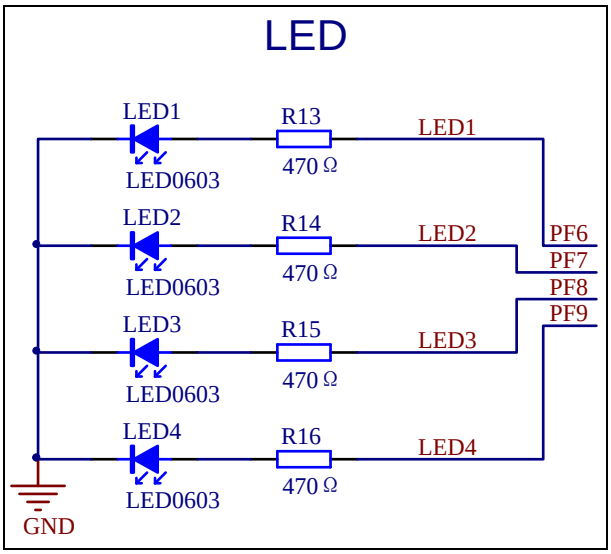
Figure 5-9 Schematic diagram of KEY



The GPIO is configured as a push-pull output, and in order to realize the effect of light water, the IO output is controlled by the GPIO setting or resetting function. When the port is set , the GPIO output will be high, and the LED will light， When the port is reset, the GPIO output will

be low, and the LED will be off.

Figure 5-10 Schematic diagram of LED

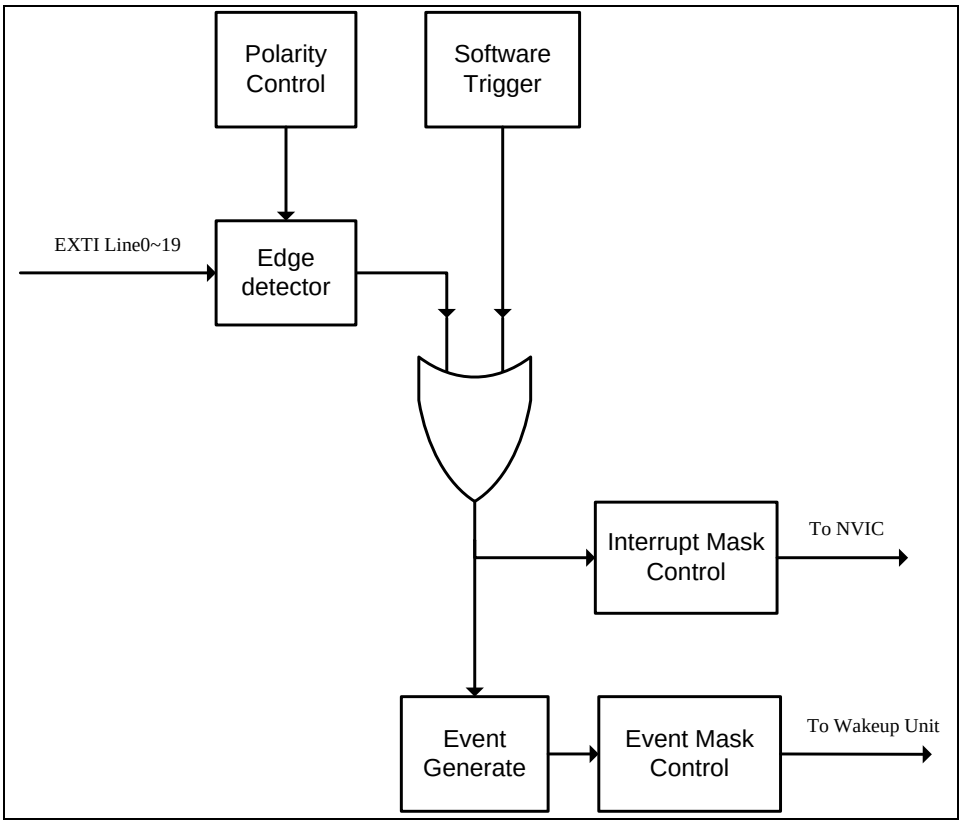


The above shows the LED circuit diagram, where the resistance of the LED

series, in series with the LED is mainly used to limit the current to avoid damaging the LED and the GPIO port.

External Interrupt and Event(EXTI)The EXTI contains 20 independent edge detectors and generates interrupts request or wake up event to the processer. The EXTI has three trigger types: rising edge, falling edge and both edges. Each edge detector in the EXTI can be configured and masked independently. Figure below is the block diagram of EXTI.

Figure 5-11 Block diagram of EXTI

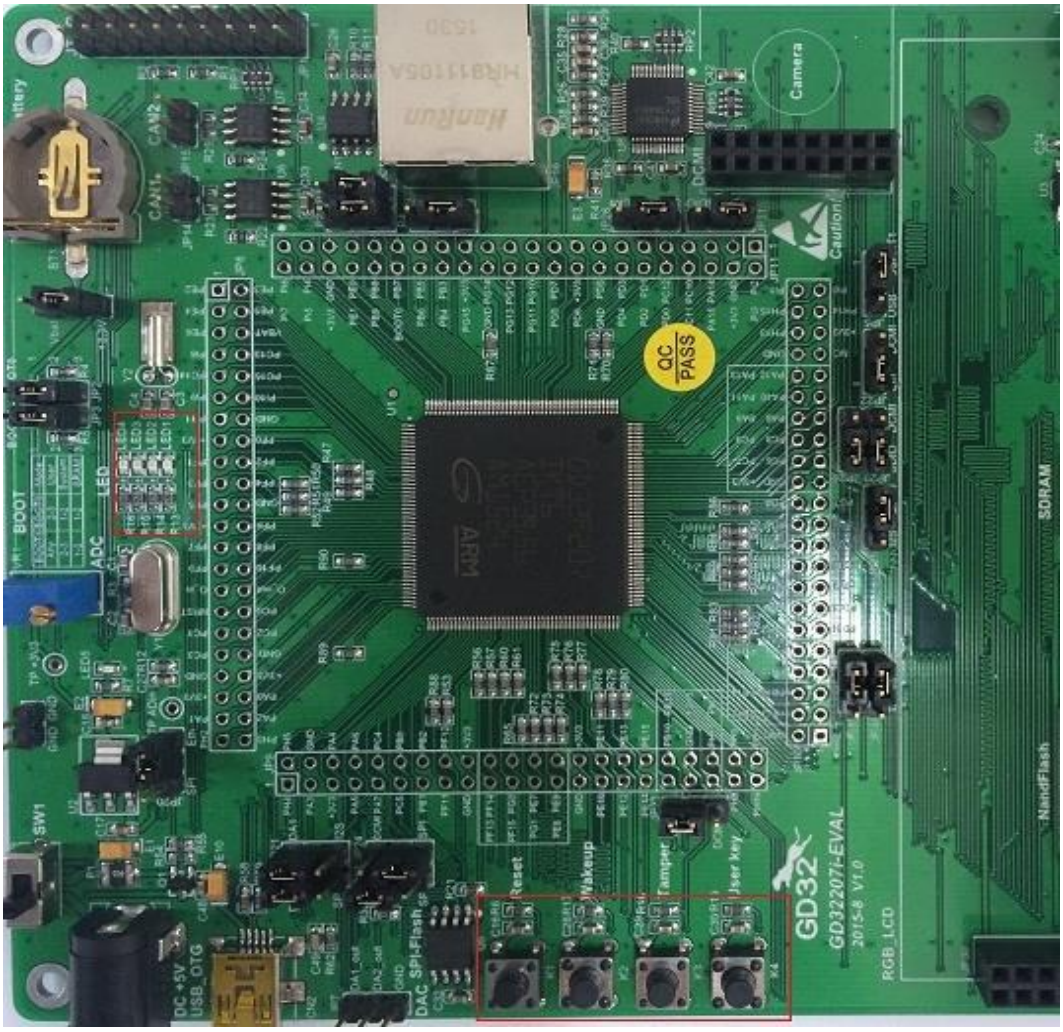


The EXTI trigger source includes 16 external lines from GPIO pins and 4 lines from internal modules (including LVD, RTC Alarm, USB Wake-up and Ethernet Wake-up, please), but this four EXTI lines are connected to the external trigger. All GPIO pins can be selected as an EXTI trigger source by configuring AFIO_ESSRx registers in GPAFIO module (please refer to GPIOs and AFIOs section for detail).

EXTI can provide not only interrupts but also event signals to the process. The Cortex-M3 processor fully implements the Wait For Interrupt (WFI), Wait For Event (WFE) and the Send Event (SEV) instructions. The GD32F20x include a Wake-up Interrupt Controller (WIC). This enables the processor and NVIC to be put into a very low-power sleep mode leaving the WIC to identify and prioritize interrupts and event. EXTI can be used to wake up processor and the whole system when some expected event occurs, such as a special GPIO pin toggling or RTC alarm.

5.3.3. DEMO Implementation Result

Download the program to the development board, the phenomenon is that the K3 be pressed down the first time in the development board, and LED1 turns on. And K3 be pressed down the second time, LED1 turns off. The position of the four LEDs and K3 is shown in the red region of the graph.



5.4. USART1_Printf

5.4.1. DEMO Purpose

GD32207I-EVAL-V1.0 board extract USART1 (Universal synchronous asynchronous receiver transmitter). GD32F207IKT6 hold 8 serial port, this Demo set USART1 for example, to show GD32F20X series USART print features and using method.

GD32F20X USART main features:

- Full duplex, asynchronous communications
- Half duplex single wire communications
- NRZ standard format (Mark/Space)
- Programmable baud-rate generator allowing speeds up to 7.5 MBits/s when the clock frequency is120 MHz and oversampling is by 16.
- Fully programmable serial interface characteristics:
  - Even, odd or no-parity bit generation/detection

  - A data word length can be 8 or 9 bits
  - 1, 1.5 or 2 stop bit generation
- Configurable data polarity
- Hardware Modem operations (CTS/RTS)
- Configurable multi-buffer communication using centralized DMA
- Separate enable bits for Transmitter and Receiver
- Transfer detection flags:
  - Receive buffer full
  - Transmit buffer empty
  - End of Transmission flags
- Parity control:
  - Transmits parity bit
  - Checks parity of received data byte
- Error detection: Overrun, Noise, Frame and Parity error
- LIN Break generation and detection
- IrDA Support
- Synchronous mode and transmitter clock output for synchronous transmission
- ISO 7816-3 (T=0 and T=1) compliant smart card interface
- Multiprocessor communication
  - Enter into mute mode if address match does not occur
  - Wake up from mute mode by idle line or address mark detection
- Support for ModBus communication
  - Timeout feature
  - CR/LF character recognition
- 12 interrupt sources with flags:
  - CTS changes
  - LIN break detection
  - Transmit data register empty
  - Transmission complete
  - Receive data register full
  - Idle line detected
  - Overrun error
  - Framing error
  - Noise error
  - Parity error
  - Receiver timeout interrupt
  - End of block interrupt

While USART1/2/3/6 is fully implemented, UART4/5/7/8 is only partially implemented with the following features not supported.

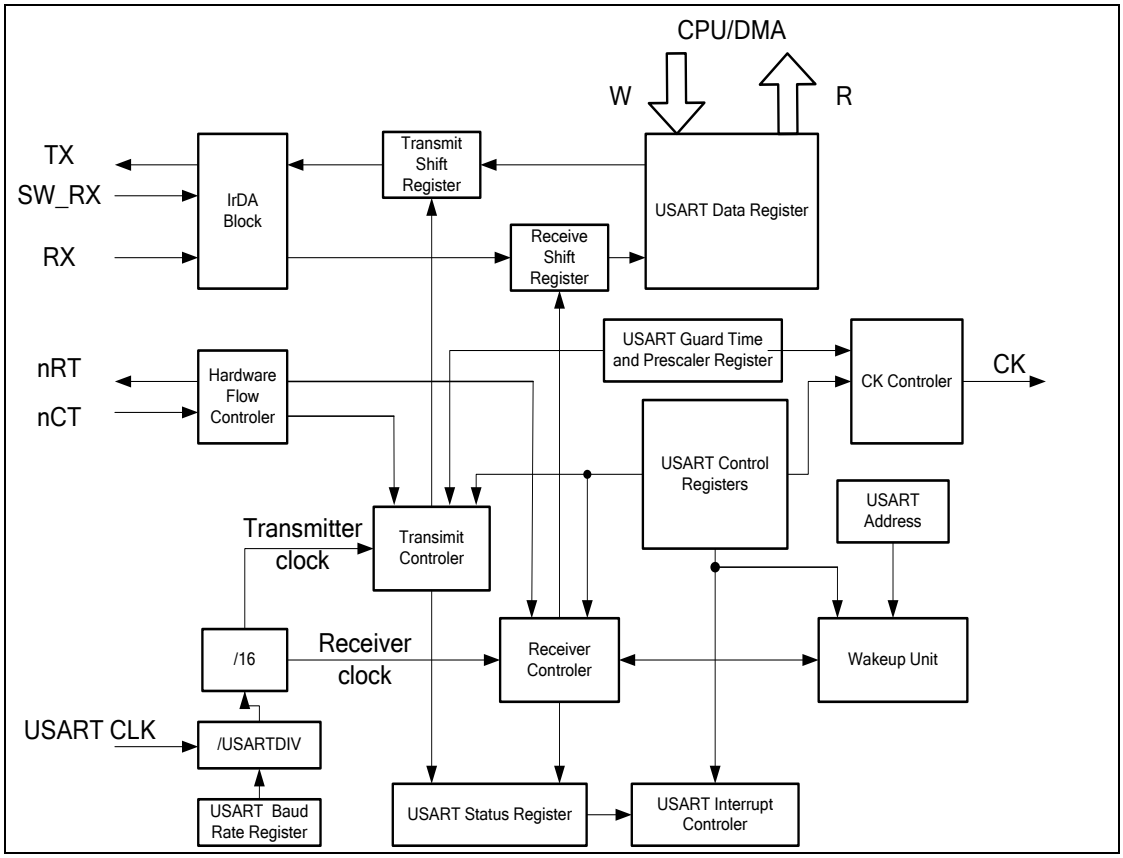
- Smartcard mode
- Synchronous mode
- Hardware Modem operations (CTS/RTS)
- Configurable data polarity

5.4.2. DEMO Principle

Table 5-1 USART important pins description

Pin	Type	Description
RX	Input	Receive Data
TX	Output I/O (single-wire/smartcard mode)	Transmit Data. high level When enabled but nothing to be transmitted
CK	Output	Serial clock for synchronous communication
nCTS	Input	Clear to send in Hardware flow control mode
nRTS	Output	Request to send in Hardware flow control mode

Figure 5-12 USART module block diagram



This DEMO realize print function, GD32F207I_EVAL board get through RS232 connect to computer for communication.

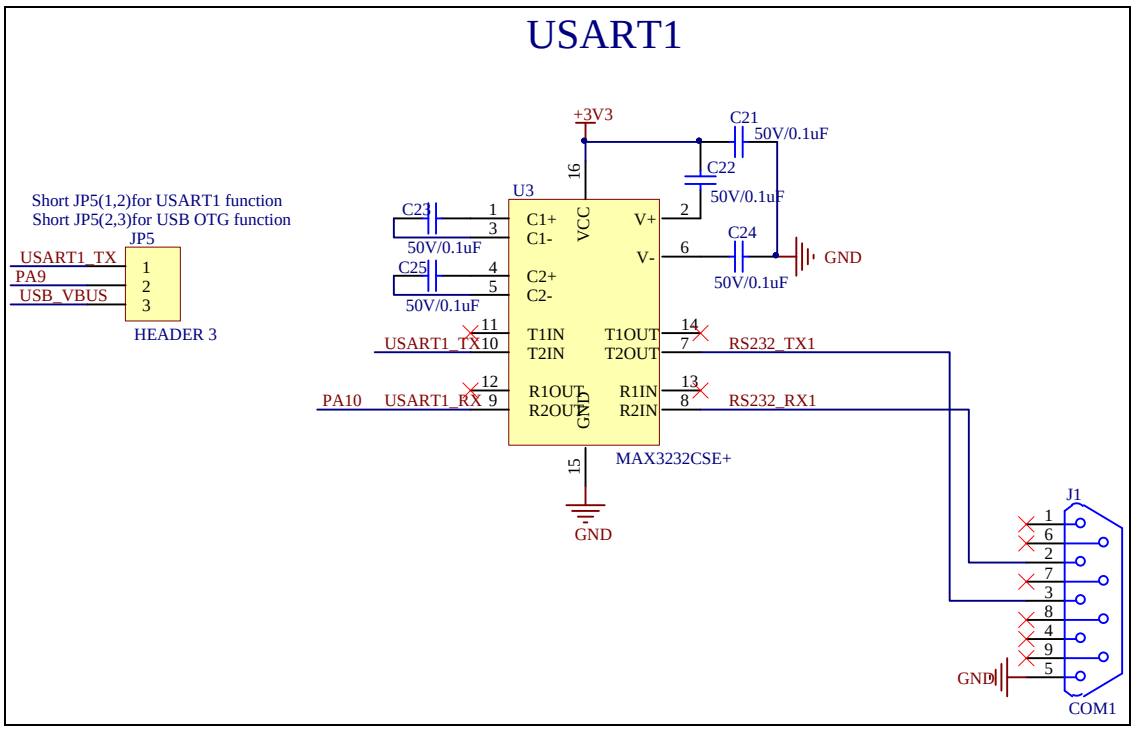
According to the following steps, USART transmit data.

1. Write the WL bit in USART_CTLR1 to set the data bits length
2. Set the stop bits length in USART_CTLR2

3. Set the baud rate in USART_BRR.
4. Set the UEN bit in USART_CTLR1 to enable the USART
5. Set the TEN bit in USART_CTLR1.
6. Wait for the TBE set
7. Write the data to in the USART_DR register
8. Wait until TC=1 to finish

This routine hardware principle diagram is shown as following.

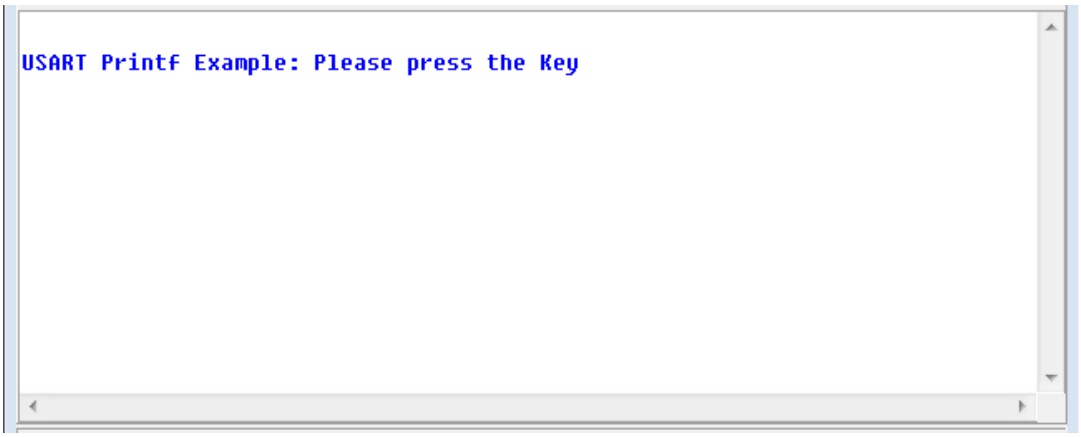
Figure 5-13 Schematic diagram of USART1



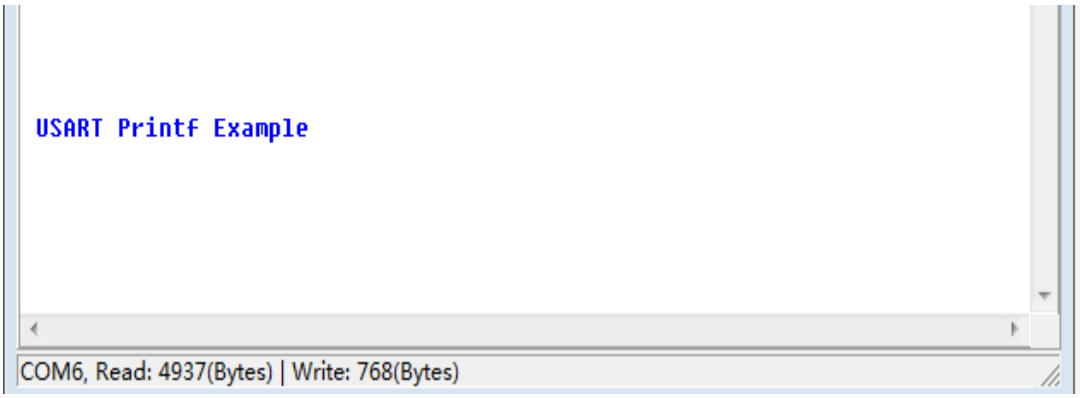
Note: for normal operation of COM1, configure JP5 USART1.

5.4.3. DEMO Implementation Result

After downloading program to board, Information via a serial port output as following.



Press K2 key, serial port output as following.



5.5. USART1_Echo_Interrupt_mode

5.5.1. DEMO Purpose

GD32207I-EVAL-V1.0 board extract USART1 (Universal synchronous asynchronous receiver transmitter). GD32F207 IKT6 hold 8 serial port, this DEMO set USART1 for example, to show GD32F20X series USART interrupt features and using method.

GD32F20X USART main features:

- Full duplex, asynchronous communications
- Half duplex single wire communications
- NRZ standard format (Mark/Space)
- Programmable baud-rate generator allowing speeds up to 7.5 MBits/s when the clock frequency is120 MHz and oversampling is by 16.
- Fully programmable serial interface characteristics:
  - Even, odd or no-parity bit generation/detection
  - A data word length can be 8 or 9 bits
  - 1, 1.5 or 2 stop bit generation
- Configurable data polarity

- Hardware Modem operations (CTS/RTS)
- Configurable multi-buffer communication using centralized DMA
- Separate enable bits for Transmitter and Receiver
- Transfer detection flags:
  - Receive buffer full
  - Transmit buffer empty
  - End of Transmission flags
- Parity control:
  - Transmits parity bit
  - Checks parity of received data byte
- Error detection: Overrun, Noise, Frame and Parity error
- LIN Break generation and detection
- IrDA Support
- Synchronous mode and transmitter clock output for synchronous transmission
- ISO 7816-3 (T=0 and T=1) compliant smart card interface
- Multiprocessor communication
  - Enter into mute mode if address match does not occur
  - Wake up from mute mode by idle line or address mark detection
- Support for ModBus communication
  - Timeout feature
  - CR/LF character recognition
- 12 interrupt sources with flags:
  - CTS changes
  - LIN break detection
  - Transmit data register empty
  - Transmission complete
  - Receive data register full
  - Idle line detected
  - Overrun error
  - Framing error
  - Noise error
  - Parity error
  - Receiver timeout interrupt
  - End of block interrupt

While USART1/2/3/6 is fully implemented, UART4/5/7/8 is only partially implemented with the following features not supported.

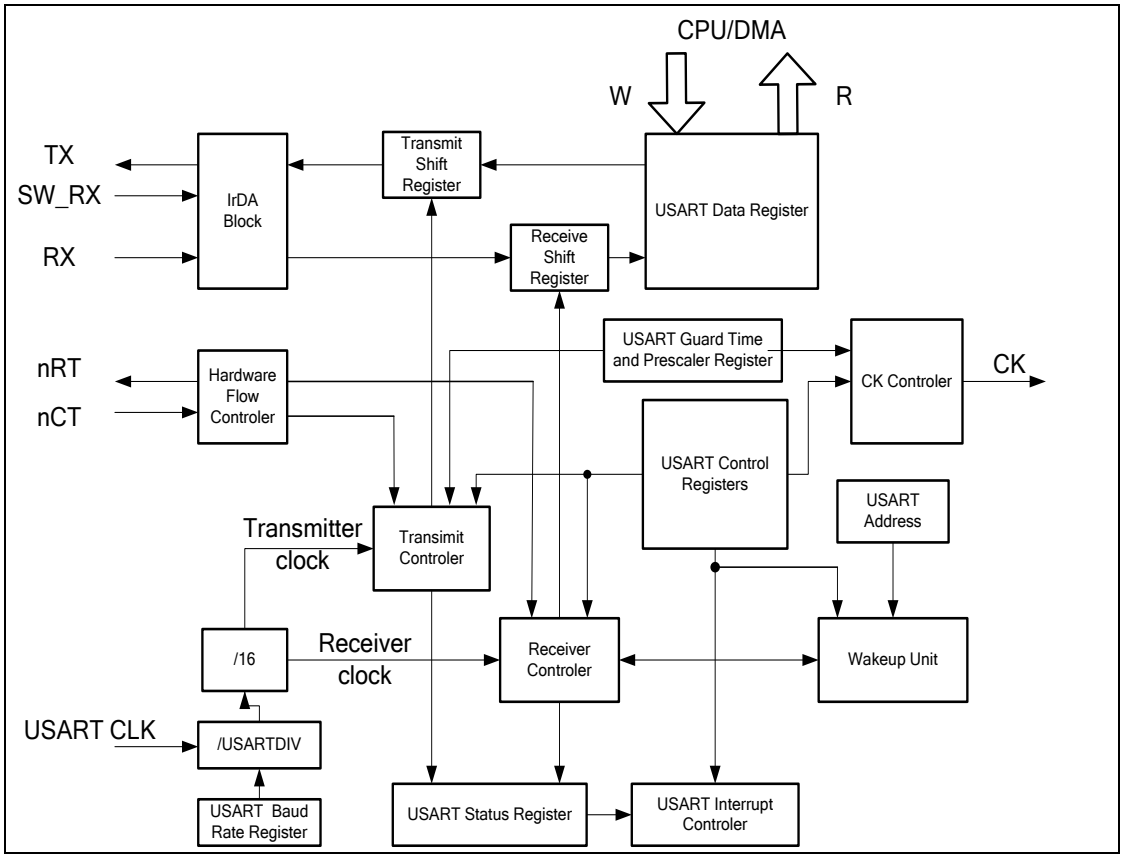
- Smartcard mode
- Synchronous mode
- Hardware Modem operations (CTS/RTS)
- Configurable data polarity

5.5.2. DEMO Principle

Table 5-2 USART important pins description

Pin	Type	Description
RX	Input	Receive Data
TX	Output I/O (single-wire/smartcard mode)	Transmit Data. high level When enabled but nothing to be transmitted
CK	Output	Serial clock for synchronous communication
nCTS	Input	Clear to send in Hardware flow control mode
nRTS	Output	Request to send in Hardware flow control mode

Figure 5-14 USART module block diagram



GD32F20X USART support full duplex interrupt transmission function, this DEMO transmit interrupt and receive data at the same time.

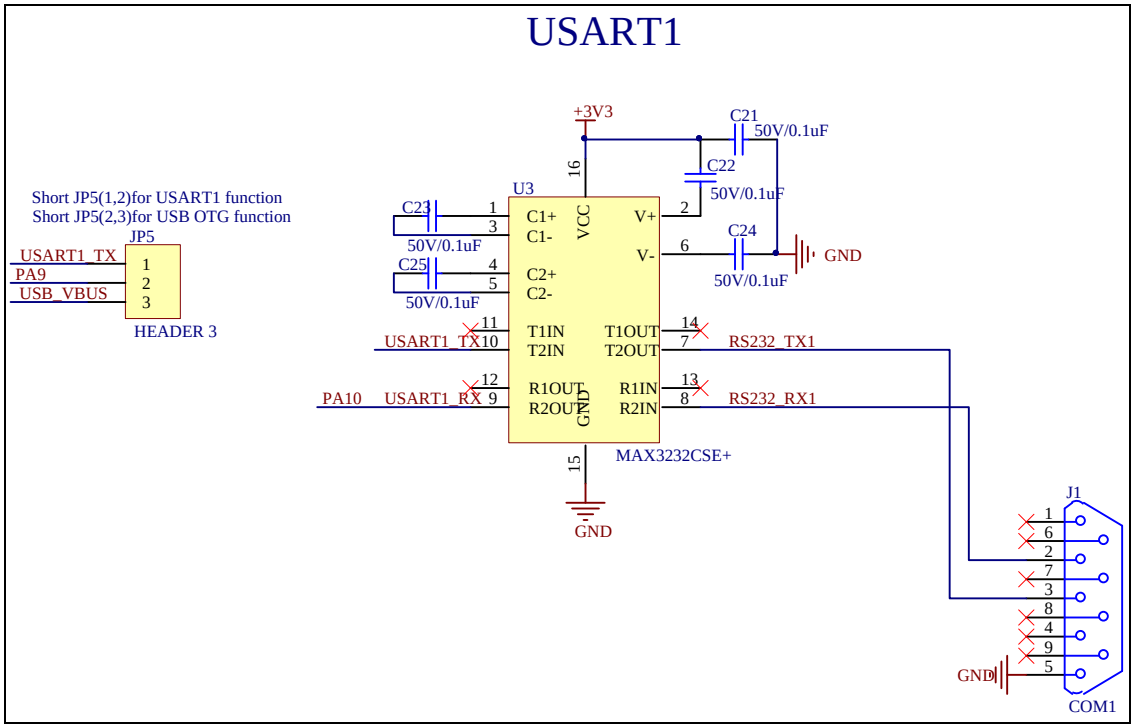
According to the following steps, USART transmit and receive interrupt.

1. Write the WL bit in USART_CTLR1 to set the data bits length
2. Set the stop bits length in USART_CTLR2

3. Set the baud rate in USART_BRR.
4. Set the UEN bit in USART_CTLR1 to enable the USART
5. Set the TEN bit and REN bit in USART_CTLR1.
6. Configure USART1 interrupt, enable USART transmit and receive interrupt.
7. Wait for transmit and receive finish, compare TxBuffer and RxBuffer.

This routine hardware principle diagram is shown as following.

Figure 5-15 Schematic diagram of USART1

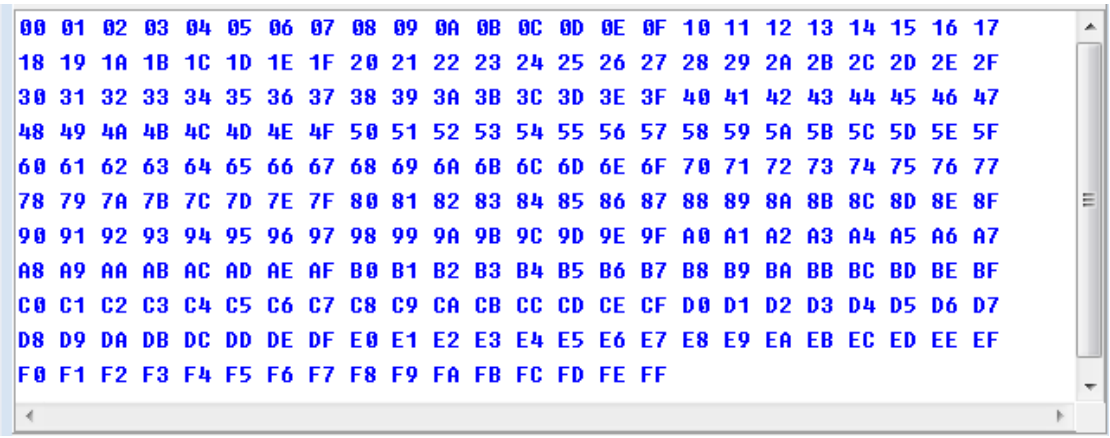


Note: for normal operation of COM1, configure JP5 USART1.

5.5.3. DEMO Implementation Result

This DEMO firstly transmit 256 byte data and wait for receiving then 256 byte data verify. If the transmitted data are same with the received data, LED on the board will be on in turn, otherwise blink meanwhile.

After downloading program to board, Information via a serial port output as following,Then transmitted 256 byte data by the serial assistant software.Observe LED on the board, if data send and data receive is same verify correctly, LED on the board will be on in turn, otherwise blink meanwhile.



5.6. USART1_DMA

5.6.1. DEMO Purpose

GD32207I-EVAL-V1.0 board extract USART1 (Universal synchronous asynchronous receiver transmitter). GD32F207 IKT6 hold 8 serial port, this DEMO set USART1 for example, to show GD32F20X series USART DMA features and using method.

GD32F20X USART main features:

- Full duplex, asynchronous communications
- Half duplex single wire communications
- NRZ standard format (Mark/Space)
- Programmable baud-rate generator allowing speeds up to 7.5 MBits/s when the clock frequency is120 MHz and oversampling is by 16.
- Fully programmable serial interface characteristics:
  - Even, odd or no-parity bit generation/detection
  - A data word length can be 8 or 9 bits
  - 1, 1.5 or 2 stop bit generation
- Configurable data polarity
- Hardware Modem operations (CTS/RTS)
- Configurable multi-buffer communication using centralized DMA
- Separate enable bits for Transmitter and Receiver
- Transfer detection flags:
  - Receive buffer full
  - Transmit buffer empty
  - End of Transmission flags
- Parity control:
  - Transmits parity bit
  - Checks parity of received data byte
- Error detection: Overrun, Noise, Frame and Parity error

- LIN Break generation and detection
- IrDA Support
- Synchronous mode and transmitter clock output for synchronous transmission
- ISO 7816-3 (T=0 and T=1) compliant smart card interface
- Multiprocessor communication
  - Enter into mute mode if address match does not occur
  - Wake up from mute mode by idle line or address mark detection
- Support for ModBus communication
  - Timeout feature
  - CR/LF character recognition
- 12 interrupt sources with flags:
  - CTS changes
  - LIN break detection
  - Transmit data register empty
  - Transmission complete
  - Receive data register full
  - Idle line detected
  - Overrun error
  - Framing error
  - Noise error
  - Parity error
  - Receiver timeout interrupt
  - End of block interrupt

While USART1/2/3/6 is fully implemented, UART4/5/7/8 is only partially implemented with the following features not supported.

- Smartcard mode
- Synchronous mode
- Hardware Modem operations (CTS/RTS)
- Configurable data polarity

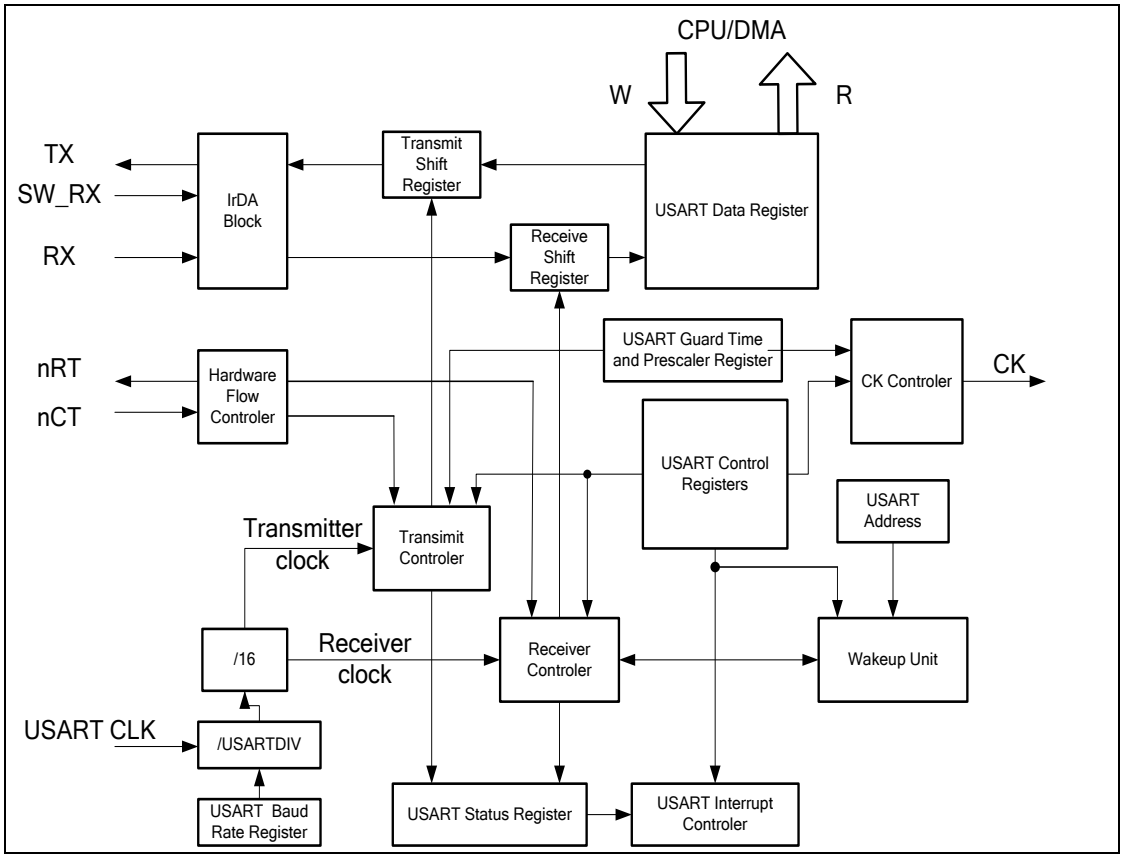
5.6.2. DEMO Principle

Table 5-3 USART important pins description

Pin	Type	Description
RX	Input	Receive Data
TX	Output I/O (single-wire/smartcard mode)	Transmit Data. high level When enabled but nothing to be transmitted
CK	Output	Serial clock for synchronous communication
nCTS	Input	Clear to send in Hardware flow control mode
nRTS	Output	Request to send in Hardware flow

		control mode

Figure 5-16 USART module block diagram



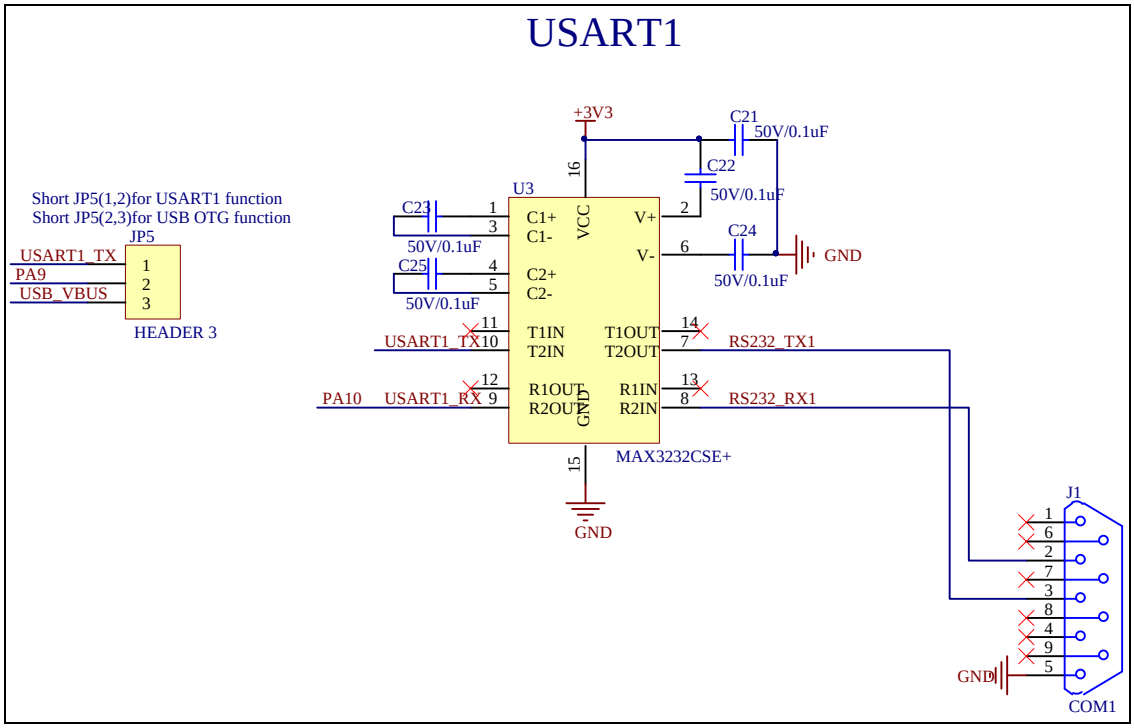
GD32F20X USART support DMA transmit function, this DEMO transmit and receive data in independent DMA channel. This DEMO firstly transmit 256 byte data and wait for receiving then verify.

According to the following steps,

1. Write the WL bit in USART_CTLR1 to set the data bits length
2. Set the stop bits length in USART_CTLR2
3. Set the baud rate in USART_BRR.
4. Set the UEN bit in USART_CTLR1 to enable the USART
5. Set the TEN bit and REN bit in USART_CTLR1.
6. Configure DMA, enable DMA corresponding channel, enable USART DMAtransmit and receive.
7. Wait for transmit and receive finish, compare TxBuffer and RxBuffer.

This routine hardware principle diagram is shown as following.

Figure 5-17 Schematic diagram of USART1



Note: for normal operation of COM1, configure JP5 USART1.

5.6.3. DEMO Implementation Result

This DEMO firstly transmit 256 byte data and wait for receiving then 256 byte data verify. If the transmitted data are same with the received data, LED on the board will be on in turn, otherwise blink meanwhile.

After downloading program to board, Information via a serial port output as following,Then transmitted 256 byte data by the serial assistant software.Observe LED on the board, if data send and data receive is same as well verify correctly, LED on the board will be on in turn, otherwise blink meanwhile.

```
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. I2C read and write EEPROM

5.7.1. DEMO Purpose

GD32207I-EVAL-V1.0 board integrated the I2C (circuit inter-integrated) module, and the module provides an I2C interface which is an industry standard two line serial interface for MCU to communicate with external I2C interface. In this Demo, putting the common AT24C02C-SSHM-T chip with the I2C interface as the access object, through the DEMO, we can have an in-depth understanding of the I2C bus, and then master how to read and write the EEPROM interface I2C.

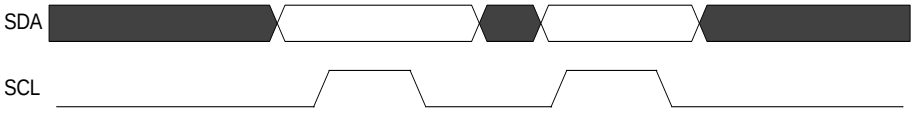
GD32F20X I2C main features:

- Parallel-bus to I2C-bus protocol converter and interface
- Both master and slave functions with the same interface
- Bi-directional data transfer between master and slave
- Supports 7-bit and 10-bit addressing and general call addressing
- Multi-master capability
- Supports Standard Speed (up to 100 kHz) and Fast Speed (up to 400 kHz)
- Configurable SCL stretching in slave mode
- Supports DMA mode
- SMBus 2.0 and PMBus compatible
- 2 Interrupts: one for successful byte transmission and the other for error event
- Optional PEC (packet error checking) generation and check

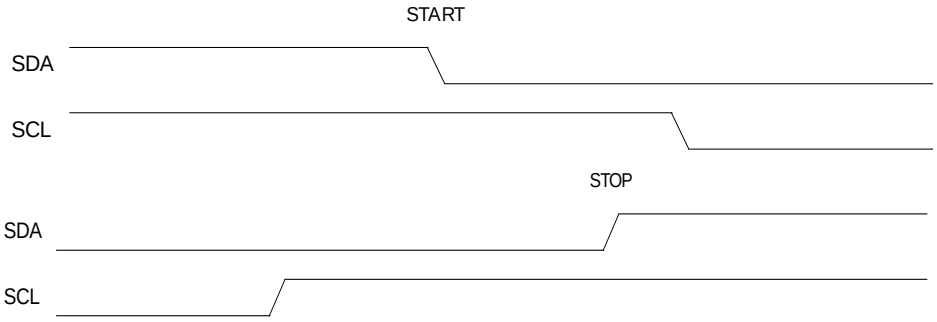
5.7.2. DEMO Principle

AT24C02C-SSHM-T access interface is I2C interface. The I2C module has two external lines, the serial data SDA and serial clock SCL lines. The two wires carry information between the devices connected to the bus. Both SDA and SCL are bidirectional lines. Data on the I2C-bus can be transferred at rates of up to 100 kbit/s in the standard-mode and up to 400 kbit/s in the fast mode.

The data on the SDA line must be stable during the HIGH period of the clock. The HIGH or LOW state of the data line can only change when the clock signal on the SCL line is LOW. One clock pulse is generated for each data bit transferred. As shown in the figure below:



All transactions begin with a START (S) and are terminated by a STOP (P). A HIGH to LOW transition on the SDA line while SCL is HIGH defines a START condition. A LOW to HIGH transition on the SDA line while SCL is HIGH defines a STOP condition. As shown in the figure below:



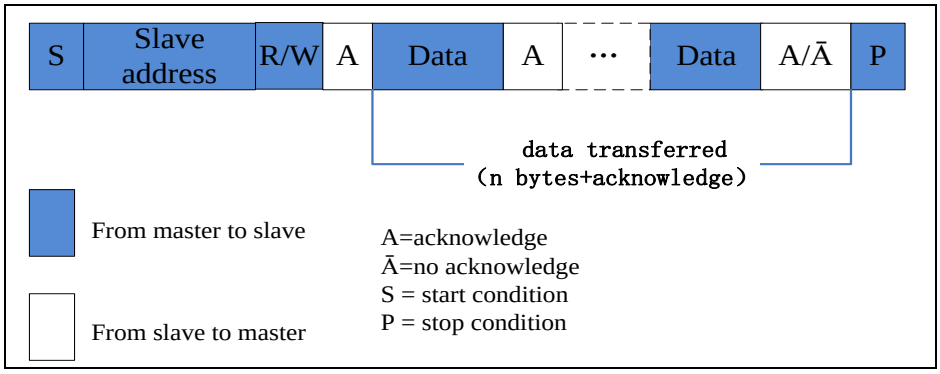
I2C communication flow:

Each I2C device is recognized by a unique address (whether it is a microcontroller, LCD driver, memory or keyboard interface) and can operate as either a transmitter or receiver, depending on the function of the device.

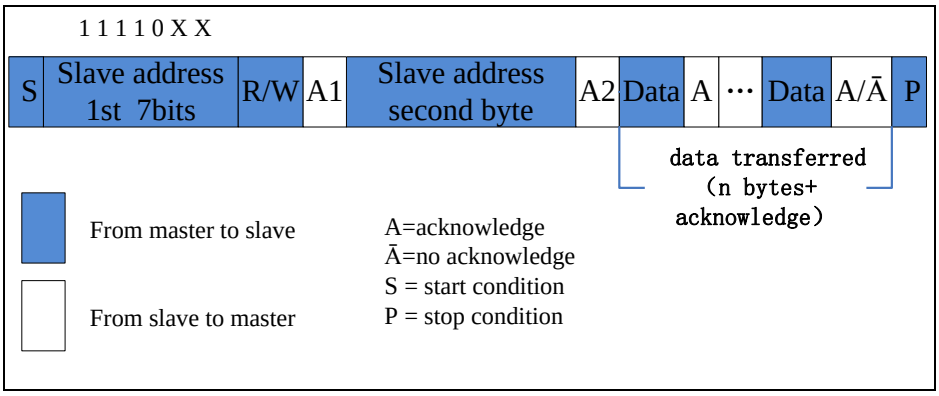
An I2C slave will continue to detect addresses after a start condition on I2C bus and compare the detected address with its own address which is programmable by software. Once the two addresses matches, the I2C slave will send an ACK to the I2C bus and responses to the following command on I2C bus: transmitting or receiving desired data. Additionally, if General Call is enabled by software, an I2C slave always responses to a General Call Address (0x00). The I2C block support both 7-bit and 10-bit addresses.

An I2C master always initiates or end a transfer using Start or Stop condition and it's also responsible for SCL clock generation.

I2C communication flow with 7-bit address:

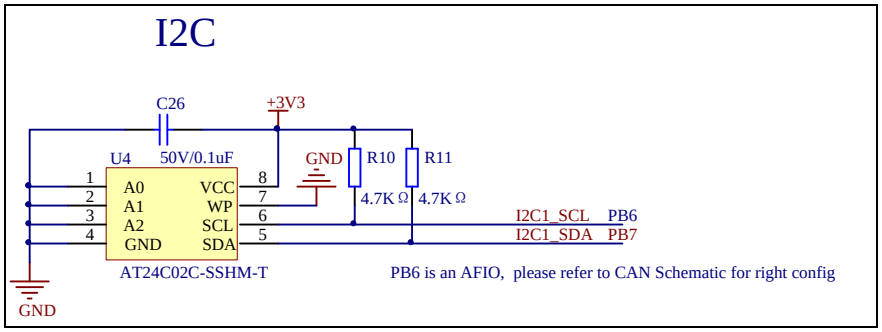


I2C communication flow with 10-bit address:



In this DEMO, the board GD32207I-EVAL-V1.0 comes with EEPROM AT24C02C-SSHM-T, and the chip capacity is 2Kbit. The board through the processor comes with the hardware I2C1 interface and the EEPROM connection. Circuit diagram as follows:

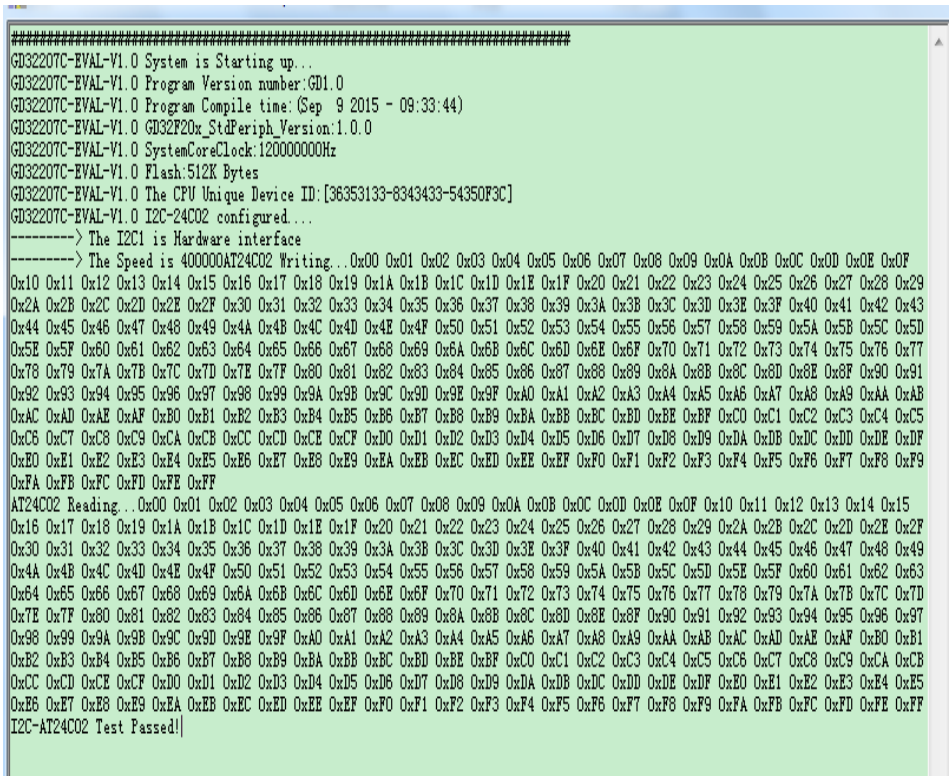
Figure 5-18 Schematic diagram of I2C



In this DEMO, firstly through the I2C interface write 256 bytes of data to EEPROM, and then read our just written into the data. Compare the data written and read the data is consistent. In the experiment the data to read, etc will be printed out through the serial port. Before the experiment using the jumper cap to connect P4(1,2).

5.7.3. DEMO Implementation Result

Download procedures, and under normal circumstances, serial print out the following information:



Firstly print some information on the board, data to written and data to read out,

and then print the results of the written data and read the results are consistent.

Program from address 0x00 sequential writes 256 bytes of data to the EEPROM, then program from address 0x00 order to read 256 bytes of data. To compare the data and read the data read, if the same, serial print out "Test Passed I2C-AT24C02!", while the board of the four LED lights flashing, otherwise serial print out "Read and Write are't Matching. Err:Data", while the four LED fight fully.

5.8. SPI-Flash quad wire flash read and write

5.8.1. DEMO Purpose

This Demo use SPI1 interface of GD32207I-EVAL-V1.0 development board to read and write SPI NOR FLASH at quad SPI mode. The SPI NOR FLASH is a serial FLASH memory chip GD25Q16B which size is 16Mbit ,the chip supports standard SPI and quad SPI operation instructions.

GD32F20X SPI main features:

- Master or slave operation
- Quad wire configuration available in master mode

- Programmable clock bit rate
- Programmable clock polarity and phase
- Separate transmit and receive buffer, 16 bits wide
- Programmable data frame size, 8 or 16 bits
- Programmable data order, transmit MSB-first or LSB-first
- Hardware CRC calculation and transmit automatic CRC error checking
- Full-duplex synchronous transfers on three lines
- Simplex synchronous transfers on two lines
- NSS work in software mode or hardware mode for both master and slave
- SPI bus busy status flag
- Transmission and reception flags with interrupt capability
- Master configuration fault, overrun and CRC error flags with interrupt capability
- Transmission and reception by DMA capability

5.8.2. DEMO Principle

Quad SPI principle

In quad wire configuration, SPI uses 6 pins: MOSI, MISO, IO2, IO3, SCK and NSS.

MOSI: This pin is used to transmit data in quad write mode and receive data in quad read mode.

MISO: This pin is used to transmit data in quad write mode and receive data in quad read mode.

IO2: This pin is used to transmit data in quad write mode and receive data in quad read mode.

IO3: This pin is used to transmit data in quad write mode and receive data in quad read mode.

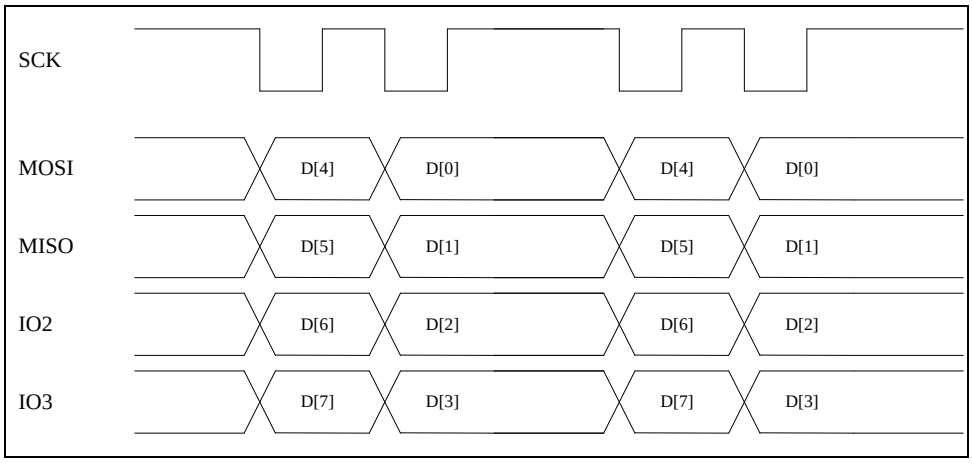
SCK: The configuration and behavior of SCK pin is the same as single wire mode except that there are only 2 clock cycles per data frame in quad wire configuration.

Data frame format: The frame length is fixed to 8 bit in quad wire mode, as shown below.

NSS: when this pin is driven low means external chip is selected.

The following figure is SPI data clock timing in quad wire mode(SCKPL=1,SCKPH=0)

Figure 5-19 SPI data clock timing in quad wire mode(SCKPL=1,SCKPH=0)



SPI FLASH principle：

According to the principle of FLASH storage, FLASH must erase before write. GD25Q16B each sector size is 4K byte and sector is the smallest unit of erase. But when user write data to, flash ,the smallest unit is a page, the page size is 256 bytes. When you read data, you can read the entire FLASH after send reading command. Please refer to the GD25Q16B reference manual for the specific operation instructions.

Hardware design

In order to read and write FLASH,we connect SPI1 pin to each SPI FLASH signal line.

Connected line is show as following :

PB1——SPIFLASH_CS——GD25Q16B_CS

PA5——SPI1_CLK——GD25Q16B_SCLK

PA7——SPI1_MOSI_IO0——GD25Q16B_SI

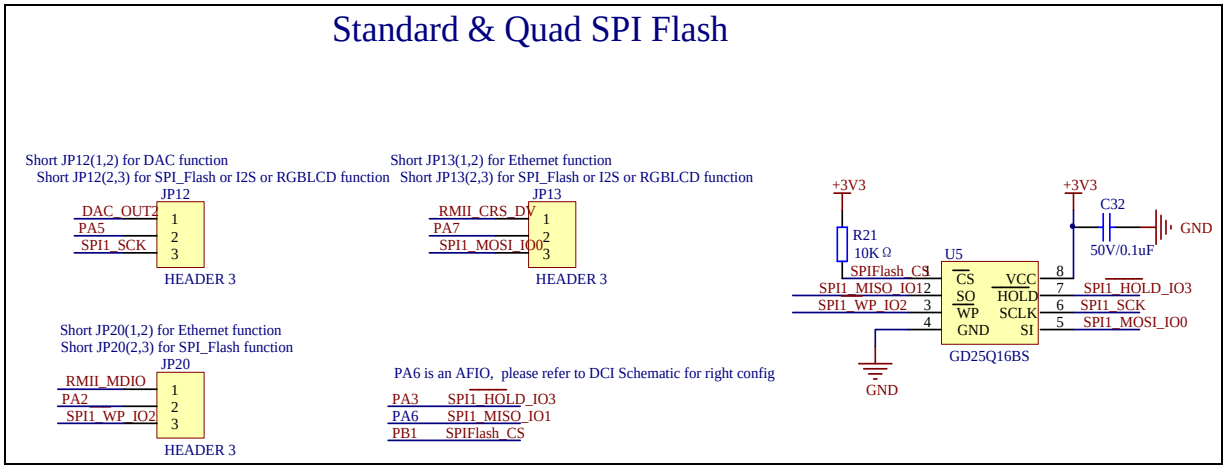
PA6——SPI1_MISO_IO1——GD25Q16B_SO

PA2——SPI1_WP_IO2——GD25Q16B_WP

PA3——SPI1_HOLD_IO3——GD25Q16B_HOLD

The following figure is GD32207I-EVAL-V1.0 development board Hardware connection of SPI FALSH module：

Figure 5-20 Schematic diagram of SPI



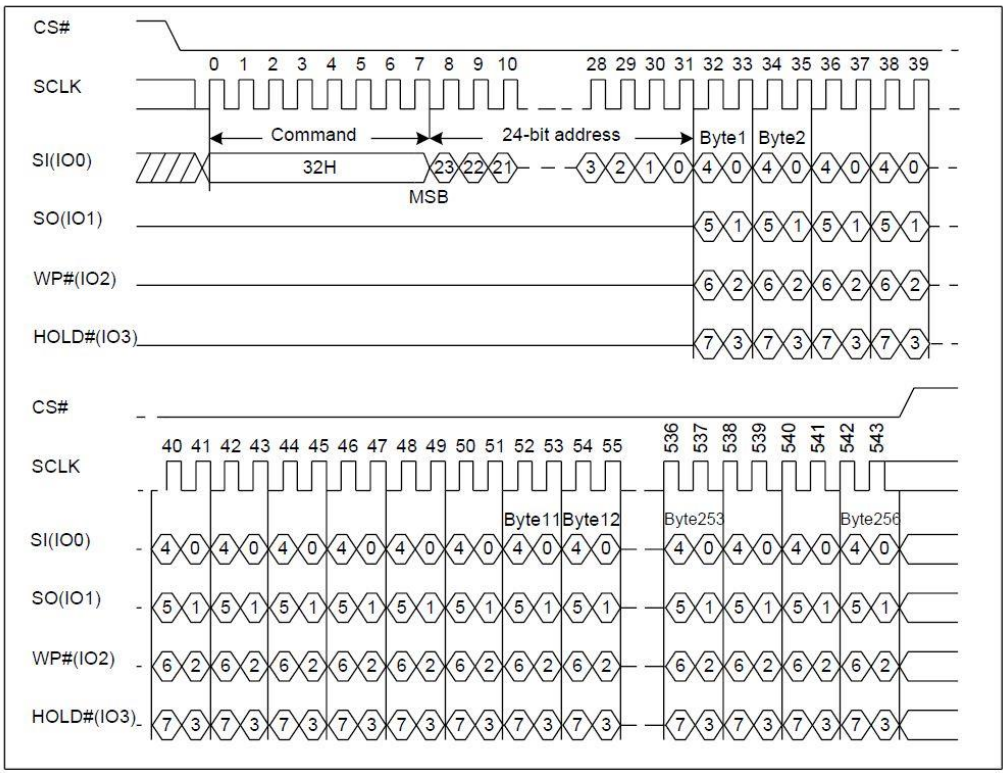
Software design

Based on the firmware library, according to the GD SPI FLASH datasheet, the serial FLASH driver file is added. According to the operation instructions described in the data sheet, to achieve the GD SPI FLASH ID read, erase, data read and write operations. Please refer to the GD25Q16B datasheet for specific instructions.

This Demo use quad SPI interface to read and write SPI NOR FLASH. Firstly, read the ID of SPI_FLASH, and then erase a sector from an address, lastly, use the quad wire mode to write and read 256 bytes from the address.

The following describes how to achieve quad wire write operation according to the clock timing of GD25Q16B datasheet:

Figure 5-21 Quad write operation timing diagram of GD25Q16B



Following is the code how to write 256 bytes to the FLASH:

```
void QSPI_FLASH_PageWrite(uint8_t* pBuffer, uint32_t WriteAddr, uint16_t NumByteToWrite)

{

    /* Enable the FLASH Quad Mode */

    QSPI_FlashQuad_Enable();

    /* Enable the write access to the FLASH */

    SPI_FLASH_WriteEnable();


    /* Select the FLASH: Chip Select low */

    SPI_FLASH_CS_LOW();

    /* Send "Quad Write to Memory " instruction */

    SPI_FLASH_SendByte(QUADWRITE);

    /* Send WriteAddr high nibble address byte to write to */

    SPI_FLASH_SendByte((WriteAddr & 0xFF0000) >> 16);

    /* Send WriteAddr medium nibble address byte to write to */
```

```
    SPI_FLASH_SendByte((WriteAddr & 0xFF00) >> 8);
    /* Send WriteAddr low nibble address byte to write to */
    SPI_FLASH_SendByte(WriteAddr & 0xFF);
    /* Enable the QSPI */
    QSPI_Enable(SPI1,ENABLE);
    /* Enable the QSPI Write Operation*/
    QSPI_Write_Enable(SPI1,ENABLE);
    /* while there is data to be written on the FLASH */
    while (NumByteToWrite--)
    {
        /* Send the current byte */
        SPI_FLASH_SendByte(*pBuffer);
        /* Point on the next byte to be written */
        pBuffer++;
    }

    /* Deselect the FLASH: Chip Select high */
    SPI_FLASH_CS_HIGH();
    /* Disable the QSPI Function */
    QSPI_Enable(SPI1,DISABLE);
    /* Wait the end of Flash writing */
    SPI_FLASH_WaitForWriteEnd();
}
```

Quad wire write operation process:

1. Send command to the FLASH, use the QSPI_FlashQuad_Enable () function to achieve the FLASH into quad mode

2. Pull low chip select signal, send quad write command QUADWRITE (0x32) and following 24 bit starting address that the data wan to write in standard SPI mode

3 .Enable the GD32MCU SPI1 QSPI function, and set the write operation;

4. Until the end of the GD32MCU SPI1 transmission in the quad wire mode, and then chip

select pin driven high, close the SPI1 quad wire SPI function;

5. Achieved a page data write operation after the last data write complete.

Quad wire SPI read operation is similar to the write operation. Please refer to the GD25Qxx.c file. In addition, the read ID command, erase command, read or write register command of FLASH memory chip GD25Q16B is performed in the standard SPI mode, only user read or write data at quad SPI mode.

5.8.3. DEMO Implementation Result

Ensure GD32207I-EVAL-V1.0 development board JP4/JP12/JP13/JP19/JP27 jumper cap jump to SPI, computer serial port line connected to the COM1 port of development board, set the baud rate of serial assistant software to 115200, 8 bits data bit, 1 bit stop bit. Download the program into the development board, through the serial assistant software can observe the operation condition, will display the ID of the flash, 256 bytes data which write to and read from flash. The following is the experimental results.

```
################################################################################
GD32207I-EVAL-V1.0 System is Starting up...
GD32207I-EVAL-V1.0 Program Version number:GD1.0.0000
GD32207I-EVAL-V1.0 Program Compile time:(Sep  8 2015 - 17:48:08)
GD32207I-EVAL-V1.0 SystemCoreClock:120000000Hz
GD32207I-EVAL-V1.0 Flash:65535K
GD32207I-EVAL-V1.0 The CPU Unique Device ID:[FFFFFFFF-FFFFFFFF-FFFFFFFF]
GD32207I-EVAL-V1.0 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 Flash:GD25Q16 Initialize Successfully!
```

5.9. EXMC_NAND Flash

5.9.1. DEMO Purpose

There is a 1Gb NAND Flash (HY27UF081G2A) on the GD32207I-EVAL-V1.0 board. The NAND Flash can be accessed via EXMC module of GD32F207IKT6. This Demo is used to show how to use EXMC NAND Flash controller to access NAND Flash.

GD32F20X EXMC main features:

- Supported external memory:
  - SRAM
  - PSRAM/SQPI-PSRAM

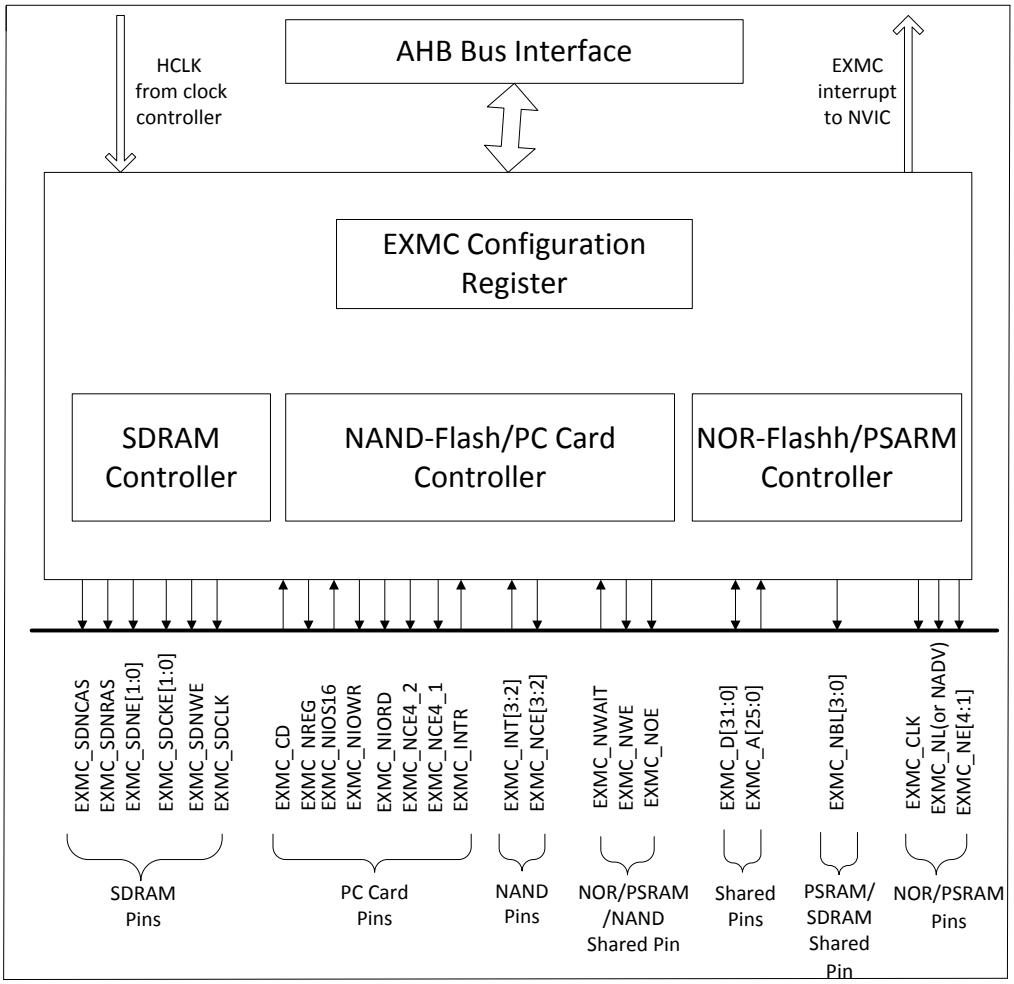
- ROM
- Nor Flash
- 8-bit or 16-bit NAND Flash
- 16-bit PC Card
- Synchronous DRAM(SDRAM)

- Conversion interface between the AHB bus and the external device protocol
- Offer a variety of programmable timing parameters to meet user specific needs
- Each bank has a separate chip-select signal which can be configured independently
- Support independent configuration of reading and writing operation timing for some devices
- Provide ECC calculating hardware module for NAND Flash memory block
- Provide 8-bit or 16-bit or 32-bit data bus
- Support address and data bus multiplexing for NOR Flash and PSRAM
- For some devices, write enable signal and byte select signal can be provided
- Automatic split AHB transaction if AHB bus data size is greater than external memory data size

5.9.2. DEMO Principle

As shown below, EXMC module includes six parts: AHB bus interface, EXMC configuration registers, NOR Flash memory controller, NAND Flash and PC Card controller, SDRAM controller, external device interface. Reference clock of EXMC module is the AHB clock (HCLK).

Figure 5-22 The EXMC block diagram



NAND Flash interface：

Table 5-4 8-bit or 16-bit NAND Flash interface signal

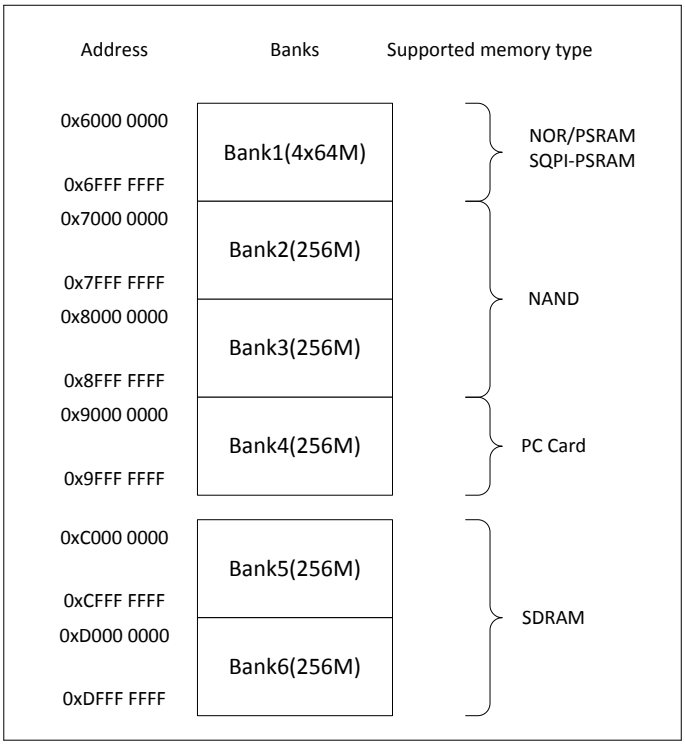
EXMC Pin	Direction	Functional description
A[17]	Output	NAND Flash address latch（ALE）
A[16]	Output	NAND Flash command latch（CLE）
D[7:0]/ D[15:0]	Input /Output	8-bit multiplexed, bidirectional address/data bus
		16-bit multiplexed, bidirectional address/data bus
NCE[x]	Output	Chip select, x = 2, 3
NOE(NRE)	Output	Output enable
NWE	Output	Write enable
NWAIT/INTx	Input	NAND Flash ready/busy input signal to the EXMC, x=2, 3

EXMC provides the conversion interface between AHB bus and external device protocol. 32-bit of AHB read or write accesses can be split into several consecutive 8-bit or 16-bit read or write operations.

EXMC external memory can be divided into many banks and they can support different types of memory. Each bank is 256 Mbytes. The address space of each bank and the supported

memory type are shown in following figure.

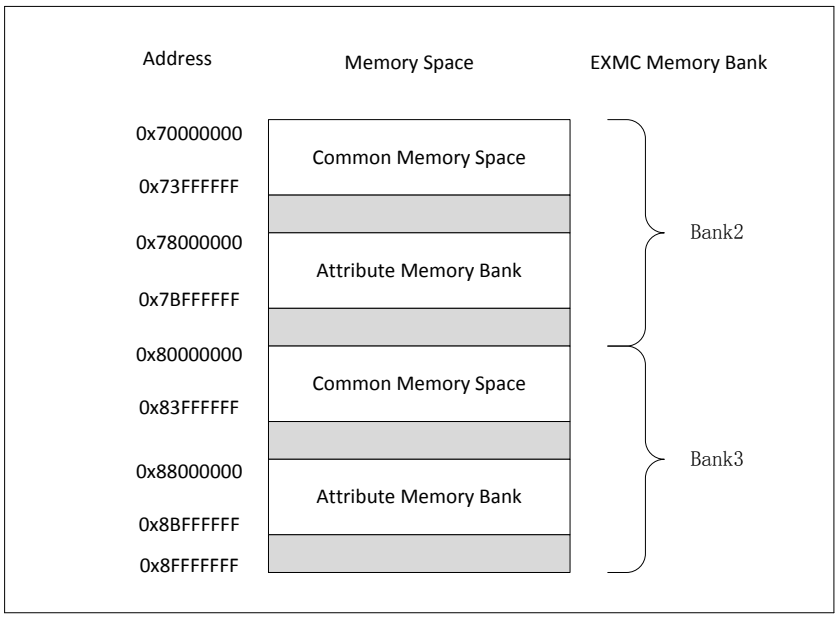
Figure 5-23 EXMC memory banks



You can access NAND Flash through the Bank2 or Bank3, and EXMC module provides dedicated registers to generate the appropriate read and write timing according to user needs and the characteristics of external memory. In addition, EXMC provide ECC computing module for access NAND Flash.

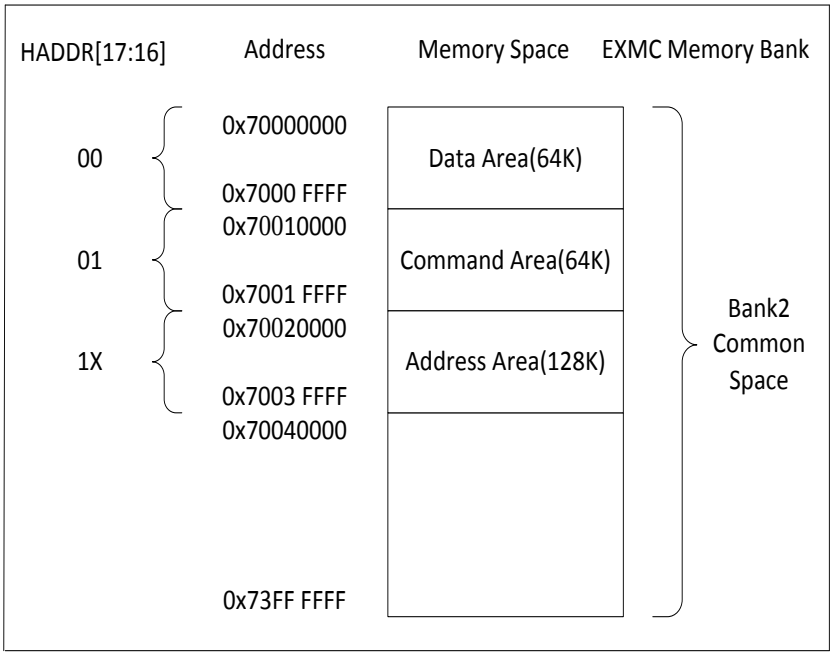
As shown below, both Bank2 and bank3 are divided into two sections, which are attribute memory space and common memory space.

Figure 5-24 NAND address mapping



For NAND Flash, common space and attribute space in the low 256K bytes of bank2 or bank3 space can be divided into three sections. The following figure is a partition diagram of bank2 common space. The attribute space of bank2, the common space and attribute space of bank3 are divided as well.

Figure 5-25 Diagram of bank2 common space



HADDR [17:16] bit are used to select one of the three area. HADDR [17:16]=00 selects the data area, HADDR[17:16]=01 selects the command area, and HADDR[17:16]=1X selects the address area. Application software uses the 3 area to access NAND Flash. Operating

rules are as follows:

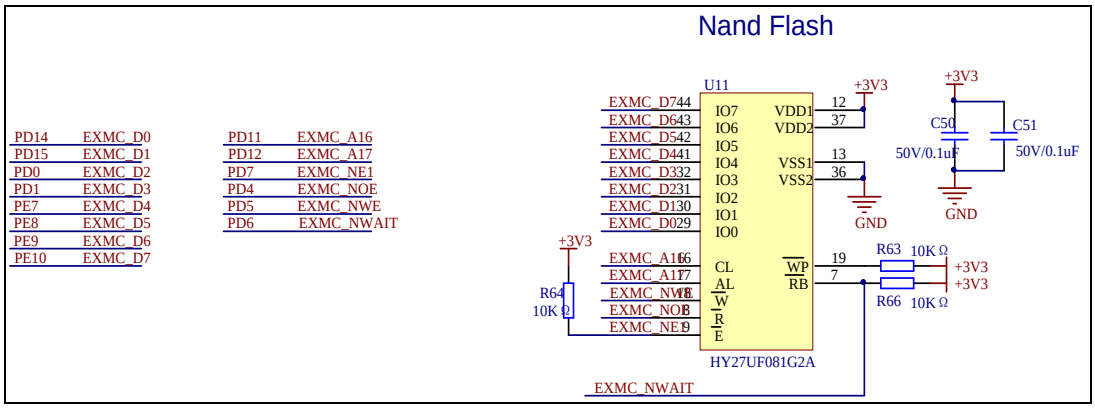
1) Send a command to NAND Flash memory: Software writes commands in the command area.

2) Specified the NAND Flash operation address: Software writes the operation address in the address area. If the number of operation address bytes is excessive, the process of writing operation address should be split into several consecutive writes in the address section.

3) Read or write the data to NAND Flash: The software reads or writes the data from or to NAND Flash data area. Since the NAND Flash memory will increase the operation address automatically, there is no need to increase the address of the data section manually to access consecutive memory locations.

The schematic diagram of 1Gb NAND Flash (HY27UF081G2A) which is on the GD32207I-EVAL-V1.0 board is shown in Figure 5-26. Data width of the NAND Flash is 8bit. Page size is (2K+64 spare) Bytes. Block size is 64 pages, namely, (128K + 4K spare) Bytes. The total storage space of this NAND Flash is (2K+64) x Bytes 64 x Pages 1024 Blocks. It is programmed basically by page, and the Erase operation is done on a block basis.

Figure 5-26 Schematic diagram of EXMC-NAND Flash function



According to NAND Flash access rules, it is required to send command firstly. And different NAND Flash device may have different operating commands .The command sets of HY27UF081G2A are shown below.

Table 5-5 Command sets

FUNCTION	1st CYCLE	2nd CYCLE	3rd CYCLE	4th CYCLE	Acceptable command during busy
READ 1	00H	20H	-	-	
READ FOR COPY-BACK	00H	35H	-	-	
READ ID	90H	-	-	-	
RESET	FFH	-	-	-	YES
PAGE PROGRAM	80H	10H	-	-	
COPY BACK PGM	85H	10H	-	-	

BLOCK ERASE	60H	D0H	-	-	
READ STATUS REGISTER	70H	-	-	-	YES
CACHE PROGRAM	80H	15H	-	-	
RANDOM DATA INPUT	85H	-	-	-	
RAMDOM DATA OUTPUT	05H	E0H	-	-	
CACHE READ START	00H	31H	-	-	
CACHE READ EXIT	34H	-	-	-	

To send the 28 addresses needed to access the 1Gbit NAND Flash, 4 clock cycles (x8 bit) are needed. The address of each cycle is listed as follows:

Table 5-6 Address Cycle Map

	IO0	**IO1**	**IO2**	**IO3**	**IO4**	**IO5**	**IO6**	**IO7**
1st Cycle	A0	A1	A2	A3	A4	A5	A6	A7
2nd Cycle	A8	A9	A10	A11	L(1)	L(1)	L(1)	L(1)
3rd Cycle	A12	A13	A14	A15	A16	A17	A18	A19
4th Cycle	A20	A21	A22	A23	A24	A25	A26	A27

NOTE:

1. L must be set to Low

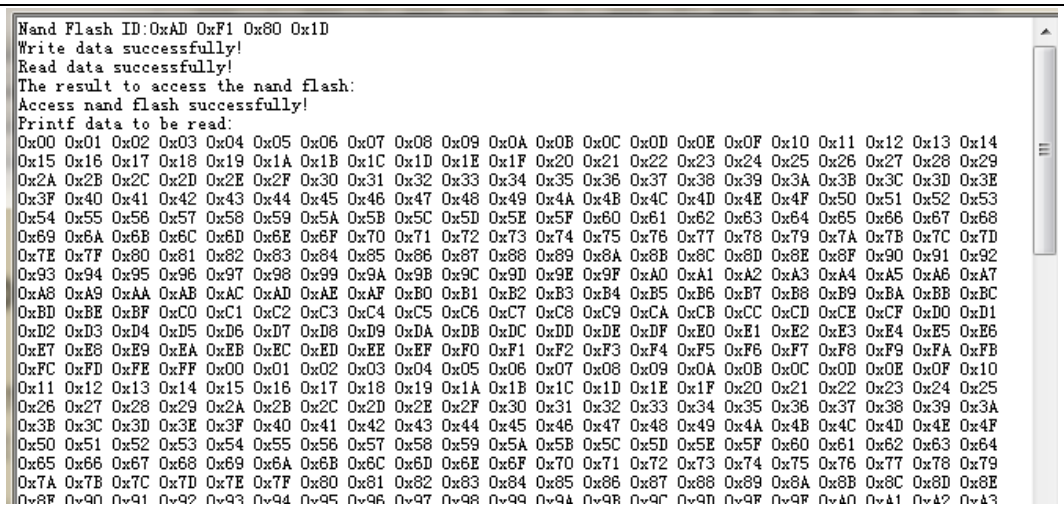
5.9.3. DEMO Implementation Result

The NAND Flash on GD32207I-EVAL-V1.0 board is connected to the Bank2 of EXMC. This Demo achieved the functions of read NAND Flash ID, erase, page write, page read through EXMC module.

After initialization, the program read NAND Flash device ID first, and print the read ID through the serial port. If the ID value read from the NAND Flash is not equal to the true device ID, the error message will be print out. Otherwise, the program will write 1K bytes data starting from a designated address of NAND Flash, then read out 1K bytes data from the designated address for verification. If all the read data and write data are equal, there are four LEDs on the GD32207I-EVAL-V1.0 board will be turned on and the serial port will print out the successful access information and the 1K bytes data read from NAND Flash. If the read and write failure, only LED1 and LED2 will be turned on, and the failure information will be print out.

Put jumper "JP5" to "USART1" and the serial line connect to COM1.The operating result can be view via the serial port.

After you download the program to the development board, if the program is running correctly, the following information will be shown through the serial port.



If the read ID is wrong, you can see the following information through the serial port.

5.10. SDIO TF card test DEMO

5.10.1. DEMO Purpose

The SD/SD I/O /MMC card host interface (SDIO) provides an interface between the AHB peripheral bus and MultiMediaCard (MMC), SD memory cards, SD I/O cards or CE-ATA devices.

GD32F20X SDIO main features:

- Full compliance with MultiMediaCard System Specification Version 4.2(and previous versions). Card supports for three different databus modes: 1-bit (default), 4-bit and 8-bit
- Full compliance with *SD Memory Card Specifications Version 2.0*
- Full compliance with *SD I/O Card Specification Version 2.0:* card support for two different databus modes: 1-bit (default) and 4-bit
- Full support of the CE-ATA features (full compliance with *CE-ATA digital protocol Version 1.1*)
- Data transfer up to 416 Mbit/sec for the 8-bit mode(the maximum frequency of MMC4.2 is 52MHz)
- Data and command output enable signals to control external bidirectional drivers
- Interrupt and DMA request to processor
- Completion Signal enables and disable feature (CE-ATA)

Based on the GD32207I-EVAL-V1.0 Board, this Demo presents how to use the SDIO controller of the MCU to initialize then communication with the SD MEMORY card which includes SD card, mini SD card and micro SD card. TF card is the another name of micro SD card.

5.10.2. DEMO Principle

The followings are the hardware the Demo used:

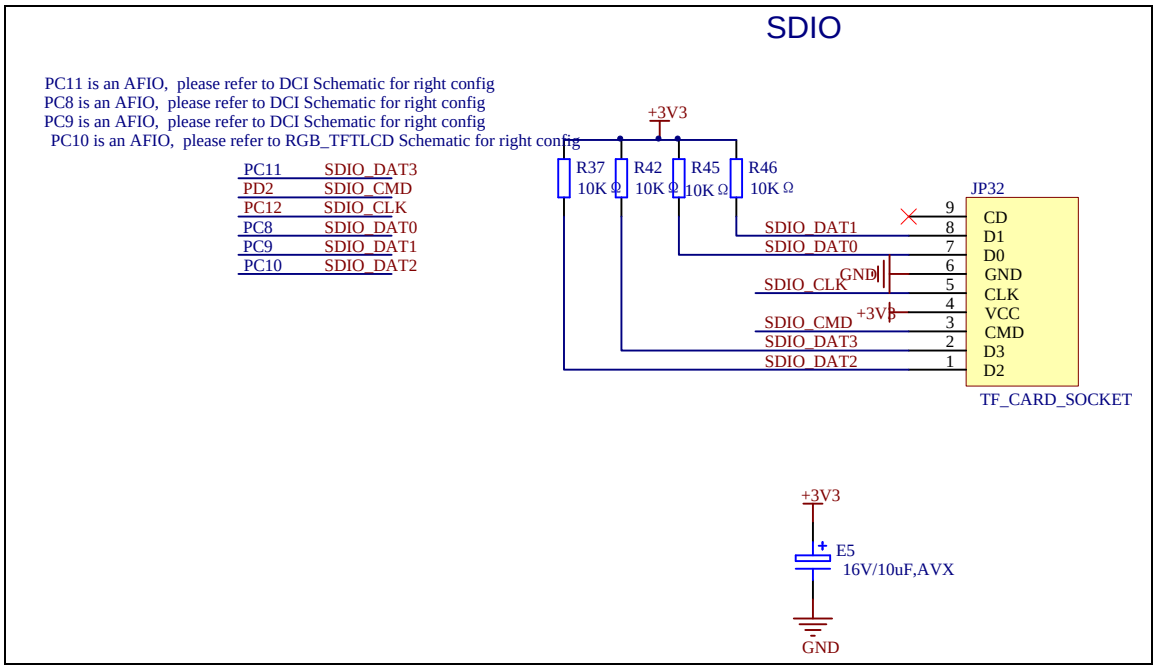
USART1: USART1 is used to print the information of the Demo when the Demo is running.

SDIO: SDIO controller is used to communication with the SDIO MEMORY.

LED: LED shows the state of the running program.

The following picture is the hardware layout of SDIO:

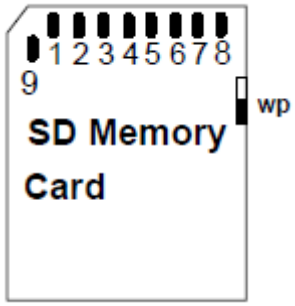
Figure 5-27 Schematic diagram of SDIO



The SD Memory Card system defines two alternative communication protocols: SD mode and SPI mode. The host system (SDIO controller) can choose either one of the modes. The card detects which mode is requested by the CS(dat3) pin output level when the reset command is received, low level means SPI mode.

The following figure shows the general shape and interface contacts of the TF Card.

Figure 5-28 SD Memory Card Shape and Interface



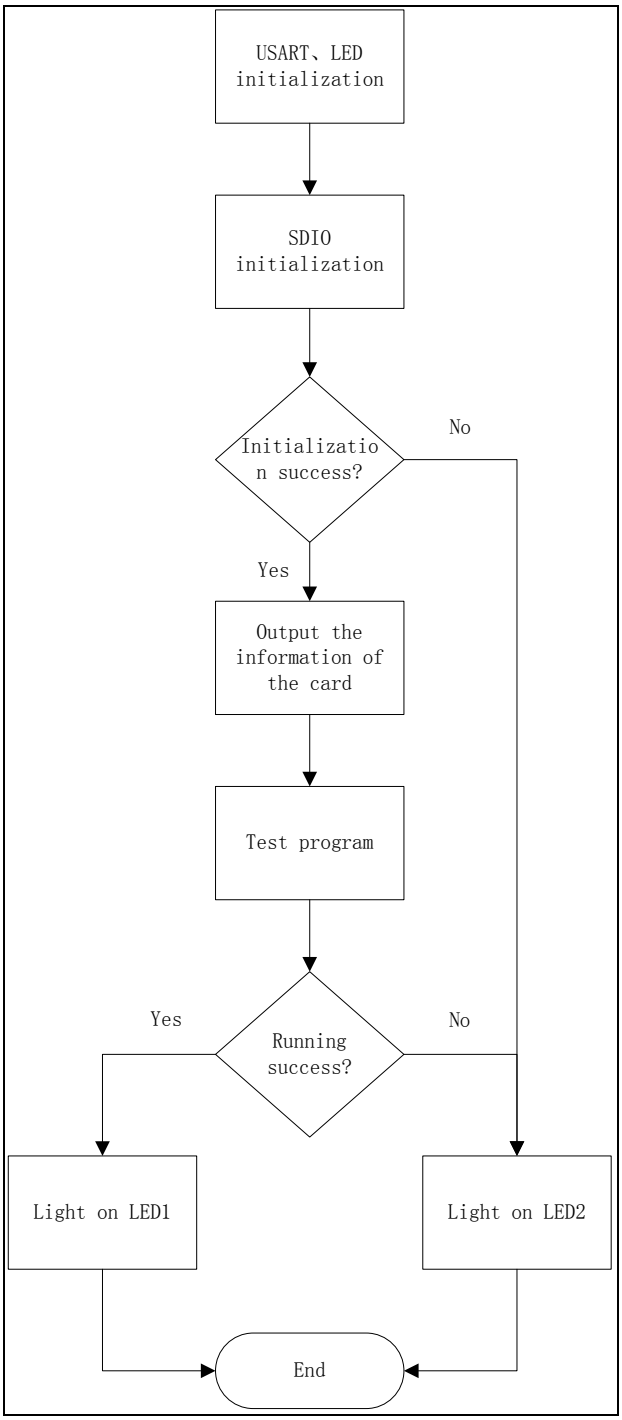
The following table defines the card contacts:

Table 5-7 Memory card alternate-function

Pin	SDIO mode		SPI mode	
	Name	Description	Name	Description
1	CD/DAT3	Card Detect/ Data Line 3	CS	Chip Select
2	CMD	Command/Response	DI	Data In
3	VSS1	Supply voltage ground	VSS	Supply voltage ground
4	VDD	Supply voltage	VDD	Supply voltage
5	CLK	Clock	SCLK	Clock
6	VSS2	Supply voltage ground	VSS2	Supply voltage ground
7	DAT0	Data Line 0	DO	Data Out
8	DAT1	Data Line 1	Reserved	
9	DAT2	Data Line 2	Reserved	

The flow diagram of the Demo shows below :

Figure 5-29 Program flow diagram



Firstly, the program initializes the USART1 and the GPIO of LED and USART1.

Secondly, the program initializes TF card by SDIO controller .If the initialization runs successfully, the controller will get the information of the card, then we can select the SDIO mode (1-bit mode or 4-bit mode) and data transformation method (DMA or polling), otherwise USART1 will output “Card init failed” with the program entering an infinite loop, at the meanwhile lights on LED2.

After initialization, USART1 outputs the information about the TF card, including the version

of the SD protocol the card supports, the card type, the card capacity, the supported command sets of the card and so on.

Finally, testing the basic function of the card, read and write operation, lock and unlock operation, erase operation, users can uncomment “#define DATA_PRINT” to watch the details of the test program through the output of USART1.At last lights on LED1 if the test program is passed.

Jumper settings: JP5 jump to the end of USART1, JP26, JP28, JP29 and JP31 jump to the end of SDIO.

5.10.3. DEMO Implementation Result

Note: 1.This Demo may destroy file system on the card, please backup the content of the card if there exists a kind of file system.

This Demo only supports SDHC card.

Execute the Demo in the folder called “10_SDIO_SDCardTest” successfully, USART1 outputs the follow messages:

```
Card init success

Card information:
Card version 3.0x
SDHC card
Device size is  7565312KB
Block size is 512B
Block count is 15130624

CardComandClasses is: 5b5
Block operation supported
Lock&unlock supported
Erase supported
Application specific operation supported
Switch function supported
```

This indicates a successful initialization, USART1 prints the card information. Then test the basic function of the card, USART1 prints the following messages:

```
Card test:
Block write success
Block read success
The card is locked
Erase failed
The card is unlocked
Erase success
Block read success
Mul block write success
Mul block read success
```

The message “Erase failed” belongs to lock function test, if the card is locked, erase operation can’t be achieved.

5.11. RTC_Calendar

5.11.1. DEMO Purpose

GD32207I-EVAL-V1.0 development board integrated RTC (clock Real-time) real-time clock. If the battery has been installed, the accuracy of the current date and time can be guaranteed when the system is reset or power down RTC is essentially an independent timer, usually used for calendar clocks. This Demo is used to demonstrate the function and usage of the RTC module in the GD32207I-EVAL-V1.0 development board.

GD32F20X RTC main features:

- 32-bit programmable counter for counting elapsed time
- Programmable prescaler:
  - Division factor up to 2^{20}
- Separate clocks:
  - PCLK1 clock
  - RTC clock (must be at least 4 times slower than the PCLK1 clock)
- RTC clock source:
  - HSE clock divided by 128
  - LSE oscillator clock
  - LSI oscillator clock
- Maskable interrupt source:
  - Alarm interrupt
  - Seconds interrupt
  - Overflow interrupt

5.11.2. DEMO Principle

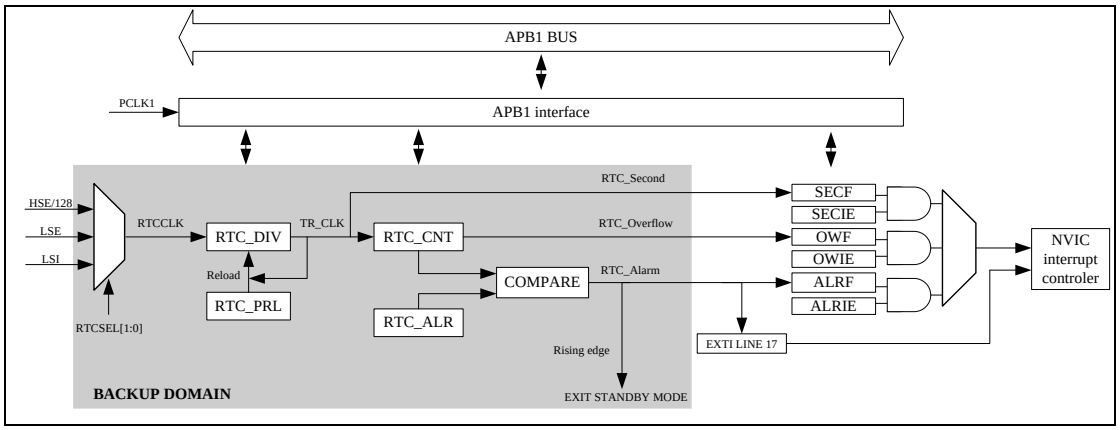
The RTC includes two major units, APB1 Interface and RTC Core.

APB1 Interface is connect with the APB1 bus. It includes a set of 16-bit registers, can be read or write from APB1 bus. In order to interface with the APB1 bus, the APB1 interface is clocked by the APB1 bus clock.

RTC Core includes two major blocks. One is the RTC prescaler block, which generates the RTC time base TR_CLK. TR_CLK can be programmed to have a period of up to 1 second. RTC prescaler block includes a 20-bit programmable divider (RTC Prescaler). If Second

Interrupt is enabled in the RTC_CTLR register, the RTC will generate an interrupt in every TR_CLK period.Another block is a 32-bit programmable counter, which can be initialized with the value of current system time. If alarm interrupt is enabled in the RTC_CTLR register, the RTC will generate an alarm interrupt when the system time equals programmable date (stored in the RTC_ALRMR register),

Figure 5-30 Block diagram of RTC



The APB1 interface and the RTC_CTLR register are reset by system reset. The RTC Core (Prescaler , Divider, Counter and Alarm) is reset only by a Backup domain reset.

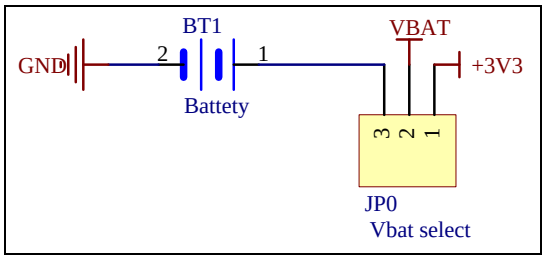
Steps to enable access to the Backup registers and the RTC after reset are as follows:

1. Set the PWREN and BKPEN bits in the RCC_APB1CCR register to enable the power and backup interface clocks;

2. Enable access to the Backup registers and RTC by setting the DBP bit in the Power Control Register (PWR_CTLR).

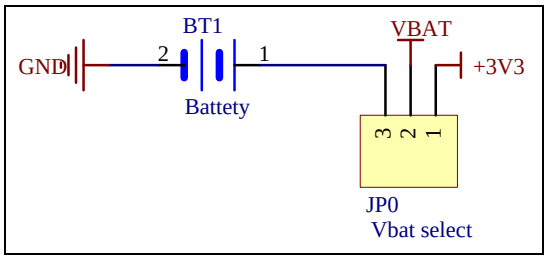
The major hardware of RTC on GD32207I-EVAL-V1.0 development board is integrated within the processor, a LES(40KHz) has been integrated within GD32F207, In order to ensure the accuracy of RTC, increase the 32.768KHz crystal drive circuit, see below.

Figure 5-31 Schematic diagram of RTC



GD32F20X VBAT adopt button battery and 3.3 V hybrid power supply, When the power is supplied by the external power supply (3.3 V),BT1 doesn't work , the power is supplied by the BT1 when the external power supply is disconnected. This hybrid power supply method can guarantee the continuous operation and precision of RTC. VBAT power supply circuit as shown below.

Figure 5-32 Schematic diagram of VBAT power supply



Jumper: battery is not installed, JP0 will jump to +3.3V using an external power supply. When installing the battery, the JP0 jumps to the Bat to use the battery power supply.

5.11.3. DEMO Implementation Result

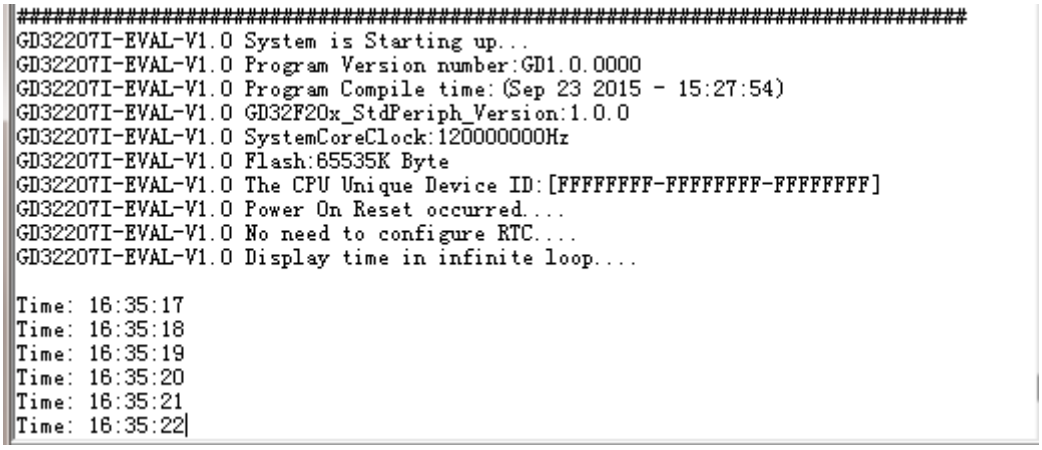
Download the program to the development board, serial port output information, as shown in the following figure. If the development board run the program for the first time, serial port output following information "GD32207I-EVAL RTC not yet configured....", that requires the user to set up time.

```
################################################################################
GD32207I-EVAL-V1.0 System is Starting up...
GD32207I-EVAL-V1.0 Program Version number:GD1.0.0000
GD32207I-EVAL-V1.0 Program Compile time:(Sep 23 2015 - 15:27:54)
GD32207I-EVAL-V1.0 GD32F20x_StdPeriph_Version:1.0.0
GD32207I-EVAL-V1.0 SystemCoreClock:120000000Hz
GD32207I-EVAL-V1.0 Flash:65535K Byte
GD32207I-EVAL-V1.0 The CPU Unique Device ID:[FFFFFFFF-FFFFFFFF-FFFFFFFF]
GD32207I-EVAL-V1.0 RTC not yet configured....
GD32207I-EVAL-V1.0 RTC configured....
==============Time Settings=====================================
  Please Set Hours
```

According to the serial port output information prompt, setting time, serial port will print out the current time every second, as shown below.

```
################################################################################
GD32207I-EVAL-V1.0 System is Starting up...
GD32207I-EVAL-V1.0 Program Version number:GD1.0.0000
GD32207I-EVAL-V1.0 Program Compile time:(Sep 23 2015 - 15:27:54)
GD32207I-EVAL-V1.0 GD32F20x_StdPeriph_Version:1.0.0
GD32207I-EVAL-V1.0 SystemCoreClock:120000000Hz
GD32207I-EVAL-V1.0 Flash:65535K Byte
GD32207I-EVAL-V1.0 The CPU Unique Device ID:[FFFFFFFF-FFFFFFFF-FFFFFFFF]
GD32207I-EVAL-V1.0 RTC not yet configured....
GD32207I-EVAL-V1.0 RTC configured....
==============Time Settings=====================================
  Please Set Hours:  16
  Please Set Minutes:  34
  Please Set Seconds:  0
GD32207I-EVAL-V1.0 Display time in infinite loop....
Time: 16:34:00
Time: 16:34:01
Time: 16:34:02
Time: 16:34:03
```

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 or battery power restart, as shown below, serial port output following information "GD32207I-EVAL No need to configured RTC....", serial port continue printing time information.



5.12. ADC analog digital conversion, including internal temperature sensor and reference voltage

5.12.1. DEMO Purpose

Analog to digital converter, is a module used to convert the analog signal into digital signal. This Demo will introduce the function and application method of ADC (analog to digital converter) module on the GD32207I-EVAL-V1.0.

GD32F20X ADC main features:

The 12-bit ADC is a successive approximation analog-to-digital converter. The analog watchdog allows the application to detect if the input voltage goes outside the user-defined higher or lower thresholds. A/D conversion of the various channels can be performed in single, continuous, scan or discontinuous mode. The output of the ADC is stored in a left-aligned or right-aligned 16-bit data register. An on-chip hardware oversample scheme improves performances while off-loading the related computational burden from the MCU

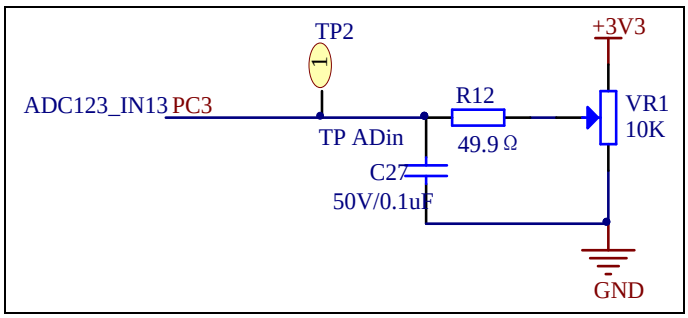
- High performance
  - —12-bit, 10-bit, 8-bit or 6-bit configurable resolution
  - —ADC sampling rate: 2 MSPs for 12-bit resolution, 2.3 MSPs for 10-bit resolution, Faster sampling rate can be obtained by lowering the resolution
  - —Self-calibration
  - —Programmable sampling time
  - —Data alignment with built-in data coherency
  - —DMA support
- Analog input channels
  - —24 external analog inputs
  - —1 channel for internal temperature sensor (VSENSE)
  - —1 channel for internal reference voltage (VREFINT)

- Start-of-conversion can be initiated
  - —By software
  - —By hardware triggers
- Conversion modes
  - —Converts a single channel or scan a sequence of channels.
  - —Single mode converts selected inputs once per trigger.
  - —Continuous mode converts selected inputs continuously
  - —Discontinuous mode
  - —Dual mode(the device with two or more ADCs)
- Analog watchdog
- Interrupt generation at the end of regular and inserted group conversions, and in case of analog watchdog
- Oversampler
  - —16-bit data register
  - —Oversampling ratio adjustable from 2 to 256x
  - —Programmable data shift up to 8-bits
- ADC supply requirements: 2.6V to 3.6V, and typical power supply voltage is 3.3V
- ADC input range: VREF- ≤VIN ≤VREF+

5.12.2. DEMO Principle

Analog to digital converter module(ADC),The 12-bit ADC is a successive approximation analog-to-digital converter. It has up to 18 channels, 16 internal channels and 2 external signal sources, A/D conversion of the various channels can be performed in single, continuous, scan or discontinuous mode. The conversion result of the ADC is stored left-aligned or right-aligned in a 16-bit data register.ADC input clock must not exceed 28MHz. In this Demo, DMA is used to transfer data continuously,PC3 (channel 13) acquisition and development board can adjust the potentiometer voltage, Internal temperature sensor (channel 16) and internal reference voltage (channel 17).Jumper cap (JP5) should be in 1 and 2.

Figure 5-33 Schematic diagram of ADC



The Potentiometer connects with PC3 on the GD32207I-EVAL-V1.0 development board, The external analog signal can be obtained from the 3.3V voltage divider resistance by the

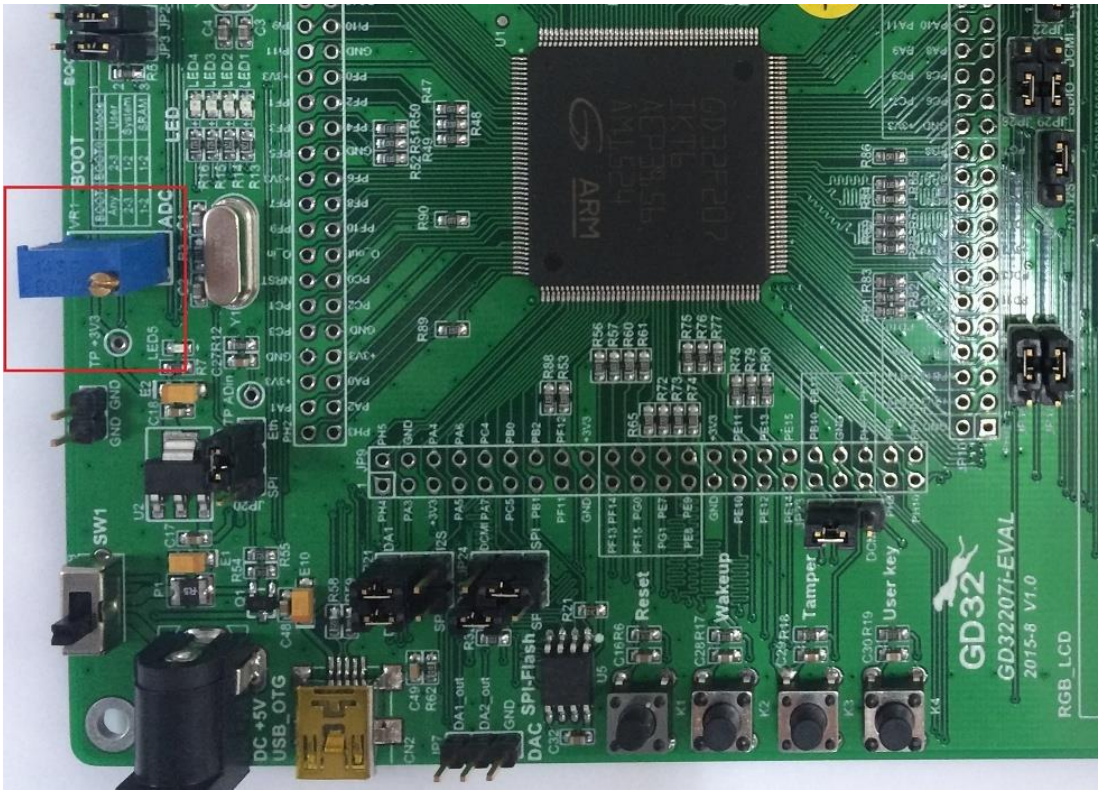
potentiometer. VR1 is the potentiometer of 10K, According to the principle of voltage divider, the output voltage is：Vout = Vcc*(Rdown/VR1). where Vcc provides the potentiometer voltage, it is 3.3V. Rdown is the lower part resistors be sampled by ADC, which is related to the current position of the slider. R12 and C27 form a RC filter circuit, as a filter for the output of the potentiometer, thus making the input of ADC is more stable. Capacitor values can't be too high, a high value will make the circuit unable to work as a filter, while a low value turn to the result that, the ability to resist interference will be weak and also be unable to meet the requirements.

5.12.3. DEMO Implementation Result

Download the program to the development board, the implementation of the results is That we can check the sampled value by live watch In debug mode. Also it can be seen through the serial port printing out the data.

```
12_ADC_Temperature_Sensor_REFINT_Channel
ADC_Value 1.105
Temperature 18
VREF_Value 3.348
```

The position of the Potentiometer and ADC peripheral circuit is shown in the red region of the graph.



5.13. DAC digital analog conversion

5.13.1. DEMO Purpose

This Demo will introduce the function and application method of DAC (digital to analog converter) module on the GD32207I-EVAL-V1.0.

GD32F20X DAC main features:

The 12-bit DAC module is a voltage output digital-to-analog converter. The DAC can be configured in 8 or 12 bit mode and may be used in conjunction with the DMA controller. The DAC has two output channels, each with its own converter. In dual DAC channel mode, conversions could be done independently or simultaneously when both channels are grouped together for synchronous update operation. An input reference pin VREF+(shared with ADC) is available for better resolution.

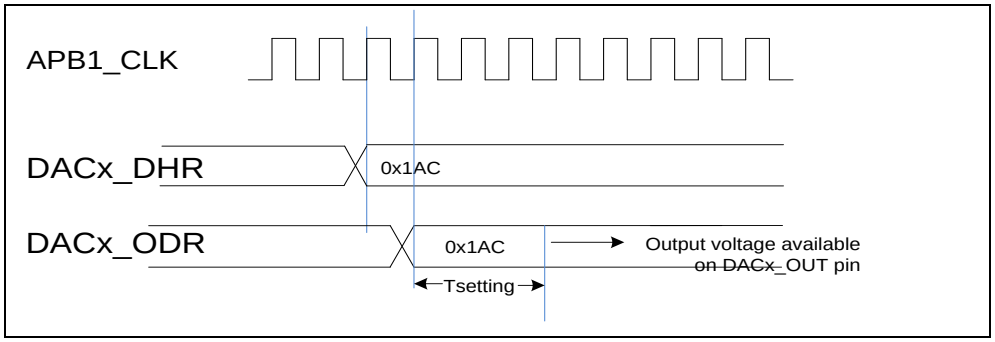
- 12-bit mode .Left or right data alignment
- Two DAC converters: one output channel each
- DMA capability for each channel
- Noise-wave generation
- Conversion update synchronously
- Triangular-wave generation
- Dual DAC independent or simultaneous conversions
- Conversion trigged by external triggers
- Input voltage reference, VREF+

5.13.2. DEMO Principle

The DACx_ODR can't be written directly and any data to transfer to the DAC channel must be performed by loading the DACx_DHR register (write on DACx_R12DHR,

DACx_R8DHR,DACD_R12DHR，DACD_L12DHR ,DACD_R8DHR).If no hardware trigger is selected (DTENx bit in DAC_CTLR register is reset),Data stored into the DACx_DHR register are automatically transferred to the DACx_ODR register after one APB1 clock cycle. However, if a hardware trigger is selected (DTENx bit in DAC_CTLR register is set) and a trigger occurs, the transfer is performed three APB1 clock cycles later. When DACx_ODR is loaded with the DACx_DHR contents, the analog output voltage becomes available after a time of Tsetting that depends on the power supply voltage and the analog output load.

Figure 5-34 Timing diagram for conversion with trigger disabled DTENx = 0



DAC output voltage:

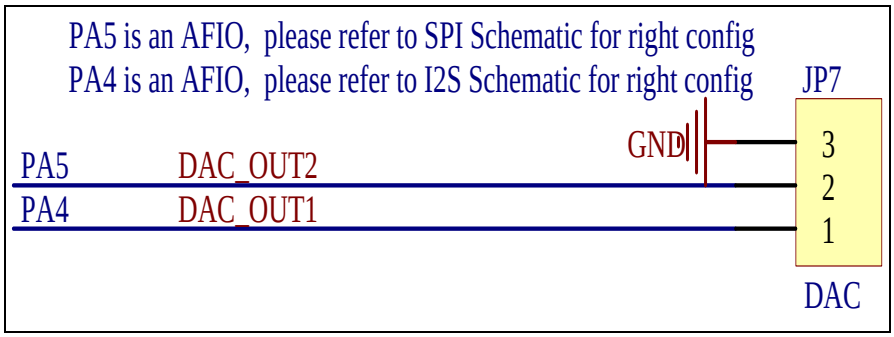
The analog output voltages on the DAC channel pin are determined by the following equation:

DAC output = VREF*DACx_ODR /4096

Digital inputs are converted to output voltages on a linear conversion between 0 and VREF+.

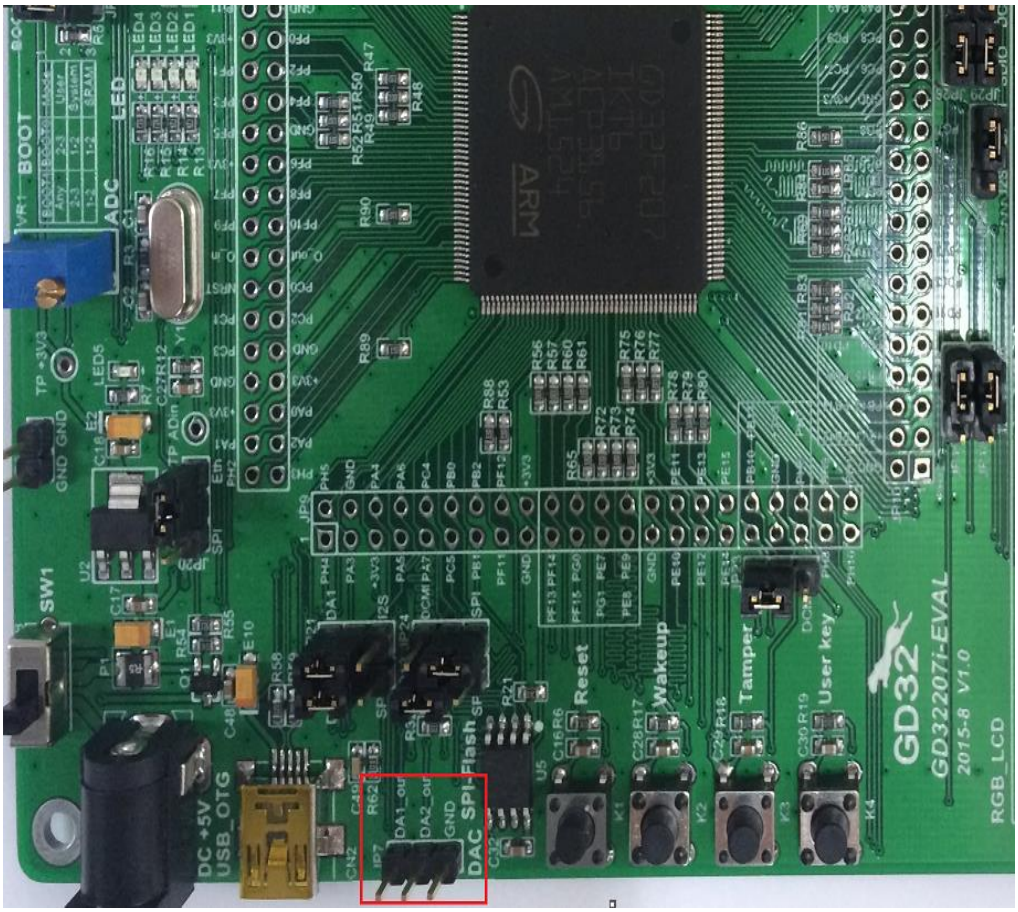
According to the DAC1 and DAC2 set of data, converted into the corresponding voltage value. Jumper cap (JP12) should be in 1 and 2. Jumper cap (JP21) should be in 1 and 2.

Figure 5-35 Schematic diagram of DAC



5.13.3. DEMO Implementation Result

Download the program to the development board, the implementation of the results is that we can get corresponding voltage value on DA1 (JP7) and DA2 (JP7) by a amultimeter or an oscilloscope. The voltage value of DA1 (JP7) is 1.655V, the voltage value of DA2 (JP7) is 0.413V. The position of the DAC peripheral circuit is shown in the red region of the graph.



5.14. Controller Area Network (bxCAN)

5.14.1. DEMO Purpose

GD32207I EVAL-V1.0 development board integrates the CAN (Controller Area Network) bus controller, which is a common industrial control bus. In this chapter, CAN bus controller follows the CAN bus protocol of 2.0 A and 2.0 B.

CAN bus controller can handle the data transmission and reception on the bus, and the application will get the desired data frame after being filtered. There are 28 filters in GD32F205 and GD32F207 series products. The application can send the data to the bus by 3sending boxes, and get the data from bus through 2 FIFOs which is of 3-worddepth. Also, CAN bus controller supports Time triggered CAN communication (Time-trigger communication). In this Demo, the function and application method of the CAN module on GD32207I EVAL-V1.0 board will be showed.

GD32F20X CAN main features:

- Supports CAN protocol version 2.0 A, B
- Baud rates up to 1 Mbit/s

- Supports the time-triggered communication
- Maskable interrupts

Transmission

- 3 transmit mailboxes
- Prioritization of messages
- Time Stamp on SOF transmission

Reception

- 2 receive FIFOs with 3 message deep
- 14 scalable/configurable identifier filter banks in GD32F203
- 28 scalable/configurable identifier filter banks in GD32F205 and GD32F207
- FIFO lock

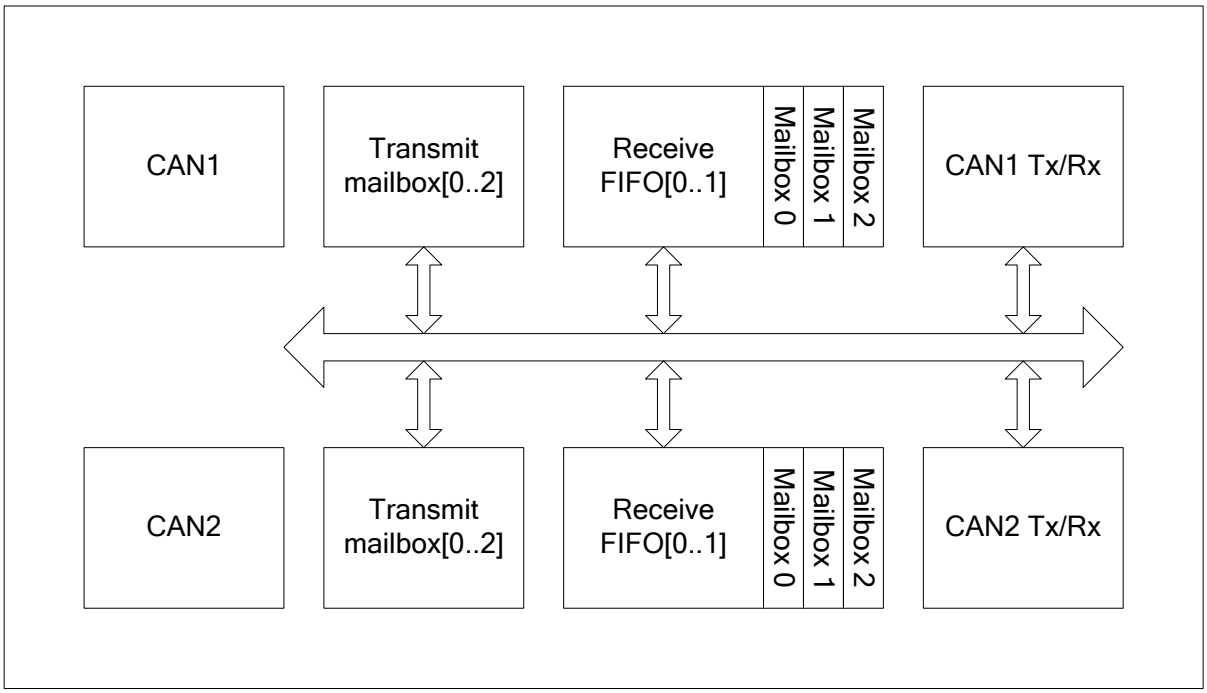
Time-triggered communication

- Disable retransmission automatically
- 16-bit free timer
- Time Stamp on SOF reception
- Time Stamp sent in last two data bytes

5.14.2. DEMO Principle

Figure below shows the CAN block diagram.

Figure 5-36 CAN module block diagram



The bxCAN interface has four communication modes:

- Silent communication mode
- Loopback communication mode
- Loopback and silent communication mode
- Normal communication mode

Silent communication mode

Silent communication mode means reception available and transmission disable.

The Rx pin of the bxCAN can get the signal from the network and the Tx pin always holds logical one.

When the SCM bit in CAN_BTR register is set, the bxCAN enters the silent communication mode. When it is cleared, the bxCAN leaves silent communication mode.

Silent communication mode is useful on monitoring the network messages.

Loopback communication mode

Loopback communication mode means the sending messages are transformed into the reception FIFOs.

Set LCM bit in CAN_BTR register to enter loopback communication mode or clear it to leave.

Loopback communication mode is useful on self-test.

Loopback and silent communication mode

Loopback and silent communication mode means the RX and TX pins are disconnected from the CAN network while the sending messages are transformed into the reception FIFOs.

Set LCM and SCM bit in CAN_BTR register to enter loopback and silent communication mode or clear them to leave.

Loopback and silent communication mode is useful on self-test. The TX pin holds logical one. The RX pin holds high impedance state.

Normal communication mode

Normal communication mode is the default communication mode unless the LCM or SCM bit in CAN_BTR register is set.

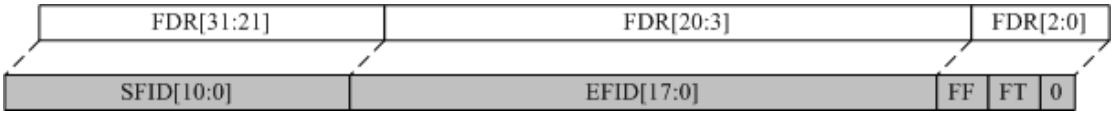
Scale

In GD32F203, the filter consists of 14 banks: bank0 to bank13. In GD32F205 and GD32F207, the filter consists of 28 banks: bank0 to bank27. Each bank has two 32-bit registers: CAN_FDR0 and CAN_FDR1.

Each filter bank can be configured 32-bit or 16-bit.

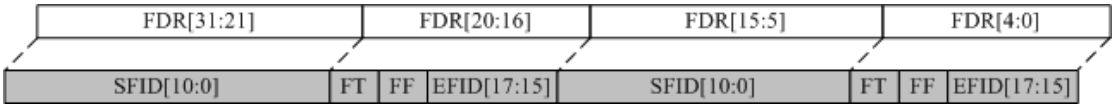
32-bit: SFID[10:0], EFID[17:0], FF and FT bits. As shown in figure below.

Figure 5-37 32-bit filter



16-bit: SFID [10:0], FT, FF and EFID[17:15] bits. As shown in figure below.

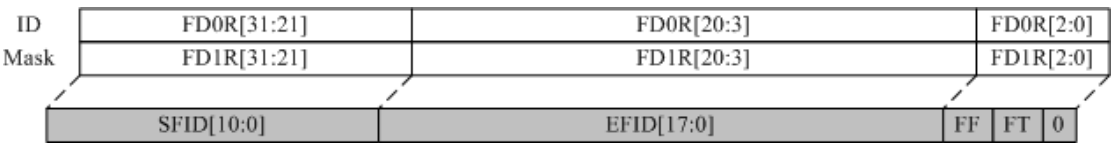
Figure 5-38 16-bit filter



Mask mode

In mask mode the identifier registers are associated with mask registers specifying which bits of the identifier are handled as “must match” or as “don’t care”. 32-bit mask mode example is shown in figure below.

Figure 5-39 32-bit mask mode filter

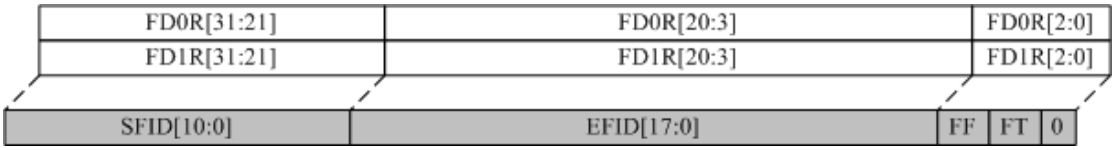


List mode

The filter consists of frame identifiers. The filter can decide whether a frame will be discarded or not. When one frame arrived, the filter will check which member can match the identifier of the frame.

32-bit list mode example is shown in figure below.

Figure 5-40 32-bit list mode filter



Filter number

Each filter within a filter bank is numbered from 0 to a maximum dependent on the mode and the scale of each of the filter banks.

For example, there are two filter banks. Bank 0 is configured as 32-bit mask mode. Bank 1 is configured as 16-bit list mode. The filter number is shown in figure below.

Figure 5-41 32-bit filter number

Filter Bank	*Filter Data Register*	*Filter Number*
0	F0D0R-32bit-ID	0
	F0D1R-32bit-Mask	
1	F1D0R-32bit-ID	1
	F1D1R-32bit-ID	2

Associated FIFO

28 banks can associate to FIFO0 or FIFO1. If the bank associated FIFO0, the frames passed through the bank will fill the FIFO0.

Active

The filter bank needs to be configured activation if the application wants the bank working and while filters not used by the application should be left deactivated.

Filtering index

Each filter number corresponds to a filtering rule. When the frame from the CAN bus passes the filters, a filter number must associate with the frame. The filter number is called filtering index. It stores in the CAN_RFMPR.FI when the frame is read by the application.

Each FIFO numbers the filters within the banks associated with the FIFO itself whatever the bank is active or not.

The example about filtering index is shown in figure below.

Figure 5-42 Filtering index

Filter Bank	FIFO0	Active	Filter Number
0	F0D0R-32bit-ID	Yes	0
	F0D1R-32bit-Mask		
1	F1D0R-32bit-ID	Yes	1
	F1D1R-32bit-ID		2
3	F3D0R[15:0]-16bit-ID	No	3
	F3D0R[31:16]-16bit-Mask		
	F3D1R[15:0]-16bit-ID		4
	F3D1R[31:16]-16bit-Mask		
7	F7D0R[15:0]-16bit-ID	No	5
	F7D0R[31:16]-16bit-ID		6
	F7D1R[15:0]-16bit-ID		7
	F7D1R[31:16]-16bit-ID		8
8	F8D0R[15:0]-16bit-ID	Yes	9
	F8D0R[31:16]-16bit-ID		10
	F8D1R[15:0]-16bit-ID		11
	F8D1R[31:16]-16bit-ID		12
9	F9D0R[15:0]-16bit-ID	Yes	13
	F9D0R[31:16]-16bit-Mask		
	F9D1R[15:0]-16bit-ID		14
	F9D1R[31:16]-16bit-Mask		
12	F12D0R-32bit-ID	Yes	15
	F12D1R-32bit-Mask		

Filter Bank	FIFO1	Active	Filter Number
2	F2D0R[15:0]-16bit-ID	Yes	0
	F2D0R[31:16]-16bit-Mask		
	F2D1R[15:0]-16bit-ID		1
	F2D1R[31:16]-16bit-Mask		
4	F4D0R-32bit-ID	No	2
	F4D1R-32bit-Mask		
5	F5D0R-32bit-ID	No	3
	F5D1R-32bit-ID		4
6	F6D0R[15:0]-16bit-ID	Yes	5
	F6D0R[31:16]-16bit-ID		6
	F6D1R[15:0]-16bit-ID		7
	F6D1R[31:16]-16bit-ID		8
10	F10D0R[15:0]-16bit-ID	No	9
	F10D0R[31:16]-16bit-Mask		
	F10D1R[15:0]-16bit-ID		10
	F10D1R[31:16]-16bit-Mask		
11	F11D0R[15:0]-16bit-ID	No	11
	F11D0R[31:16]-16bit-ID		12
	F11D1R[15:0]-16bit-ID		13
	F11D1R[31:16]-16bit-ID		14
13	F13D0R-32bit-ID	Yes	15
	F13D1R-32bit-ID		16

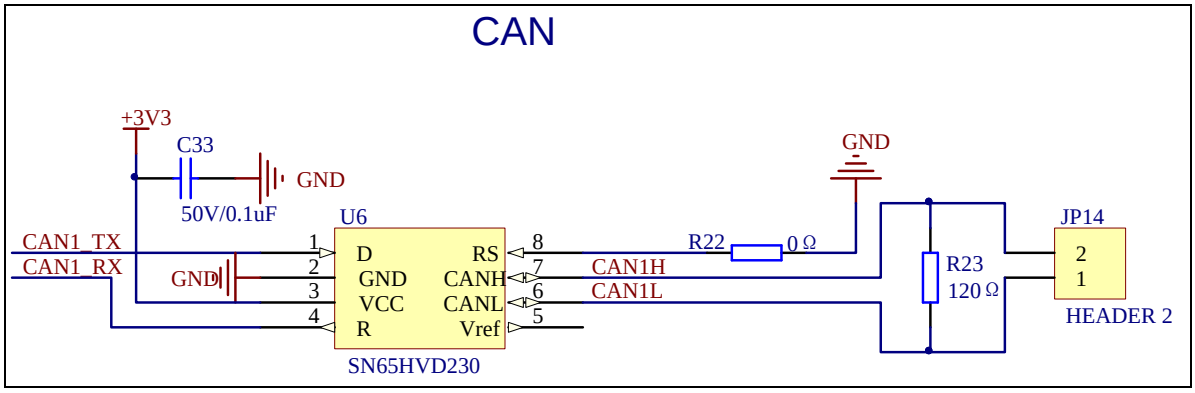
Priority

The filters have the priority:

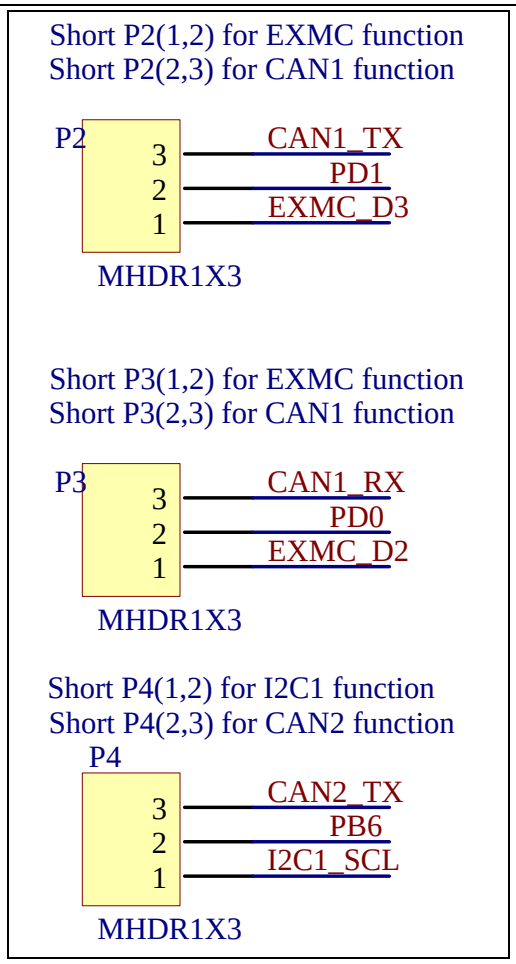
1. 32-bit mode is higher than 16-bit mode.
2. List mode is higher than mask mode.
3. Smaller filter index value has the higher priority.

The CAN module on GD32207I-EVAL-V1.0 development board has addedaCAN transceiverSN65HVD230 for data sending and receiving on CAN bus.

Figure 5-43 Schematic diagram of CAN function

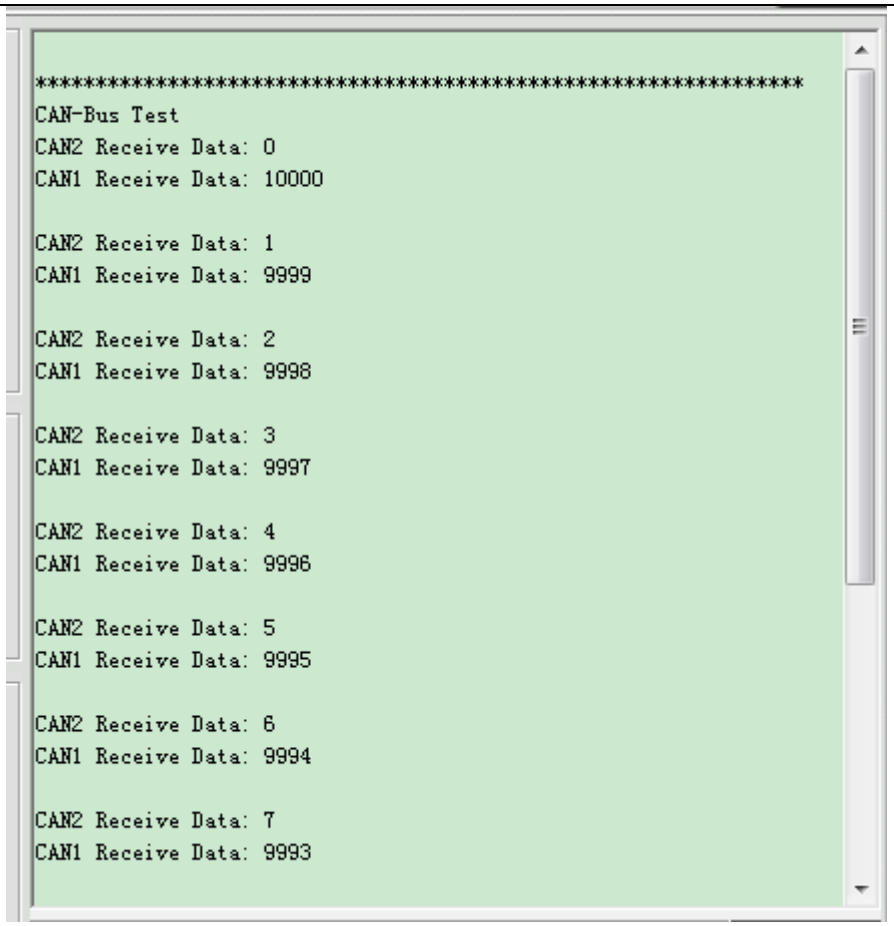


Jumper Settings: P2 / P3 / P4 should be configured to the Correct position.



5.14.3. DEMO Implementation Result

Let the signal pins of CAN1 and that of CAN2 connect together, L connects with L, H connects with H, and download the program to the development board, then the following information will be showed through the serial output.



The result shows that CAN2 has successfully received the data from CAN1, and also CAN1 has successfully received data from CAN2. CAN1 transmits data from 0, and do self-increment each time, while CAN2 transmit data from 10,000, and doself-decrement each time.

5.15. Cryptographic Acceleration Unit

5.15.1. DEMO Purpose

GD32207I-EVAL-V1.0 development board integrated CAU（Cryptographic Acceleration Unit），The Cryptographic Acceleration Unit supports acceleration of DES, Triple-DES or AES (128, 192, or 256) algorithms. The DES/TDES supports Electronic codebook (ECB) or Cipher block chaining (CBC) mode. The AES supports Electronic codebook (ECB), Cipher block chaining (CBC) mode or Counter mode (CTR) mode. This Demo is used to demonstrate the function and usage of the CAU module in the GD32207I-EVAL-V1.0 development board.

GD32F20X CAU main features:

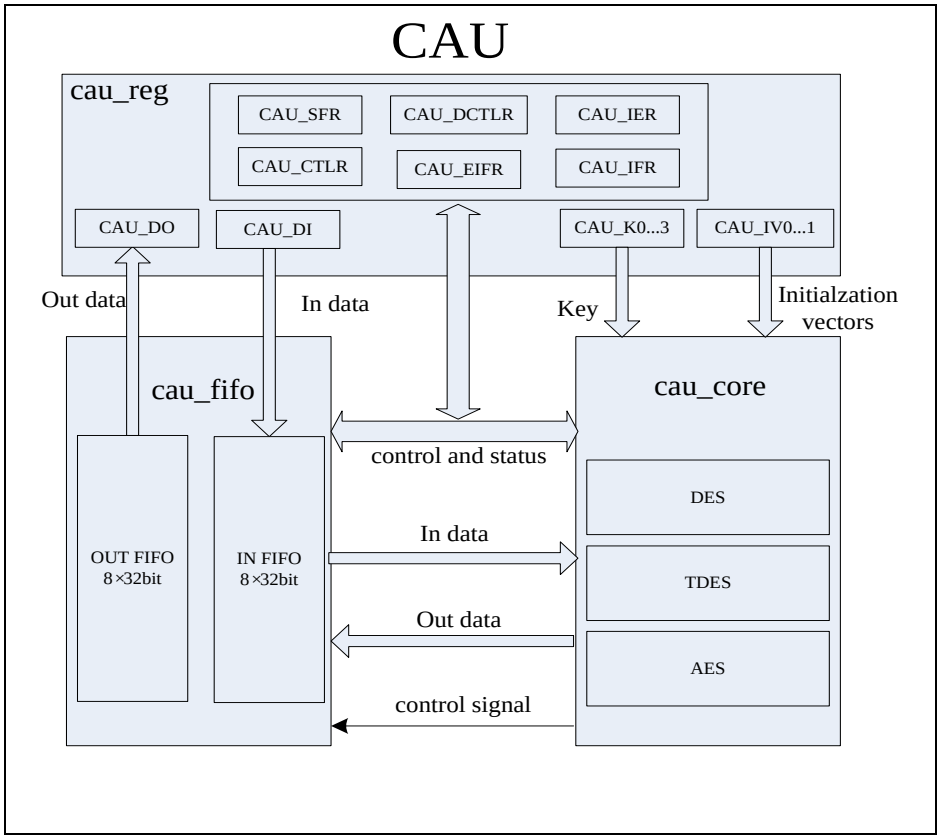
- Supports DES, Triple-DES or AES algorithm
- AES supports 128bits-key, 192bits-key or 256 bits-key

- AES supports Electronic codebook (ECB), Cipher block chaining (CBC) mode or Counter mode (CTR) mode.
- DES/TDES supports Electronic codebook (ECB) or Cipher block chaining (CBC) mode.
- TDES supports 64bits-key, 128bits-key or 192bits-key
- IN FIFO and OUT FIFO store 8x32bits data.
- Supports DMA transfer for IN FIFO or OUT FIFO
- Supports half-word swapping, byte swapping or bit swapping for IN/OUT data

5.15.2. DEMO Principle

The block diagram of the Cryptographic processor is shown as follows.

Figure 5-44 Block Diagram



The cau_reg is the register of CAU processor, which configured by AHB bus. It includes CAU_CTLR, CAU_SFR, CAU_DCTLR, CAU_DI, CAU_DO, CAU_IER, CAU_IFR, CAU_EIFR, CAU_K0…K3, CAU_IV0…1.

The cryptographic fifo have two 32x8 bits fifos to store in/out data. The IN FIFO is written by cau reg, when the AHB bus writes the CAU_DI register Then the data is read to calculation by CAU processing core. It store the input data, which is plaintext when encryption and ciphertext when decryption. The OUT FIFO is written by CAU processing core to store processing result, and read by cau_reg to CAU_DO register. It stores the output data, which

is plaintext in deccryption or ciphertext in enryption.

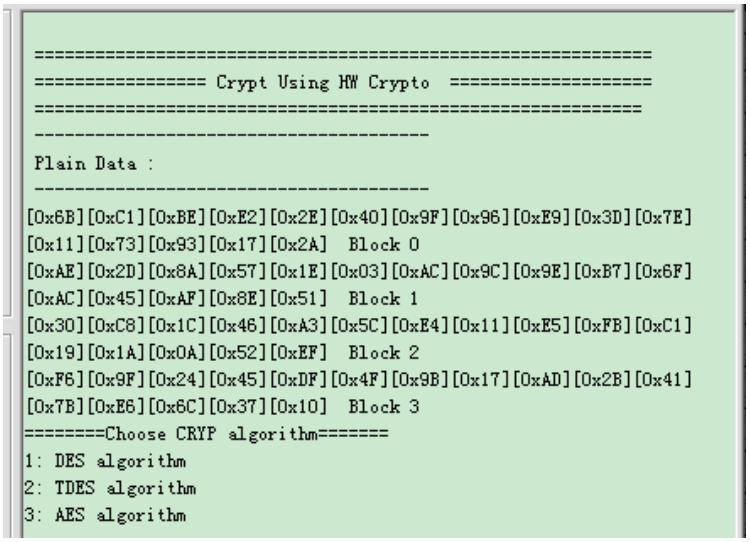
The cau_core module is the CAU processing core to process AES/DES/TDES algorithm. The CAU processing core gets input data from IN FIFO, and control register from cau_reg. After calculation, the processing core send calculation result to OUT FIFO.

The CAU processing core include AES processing core, which process AES algorithm and DES/TDES processing core, which process DES/TEDS algorithm.

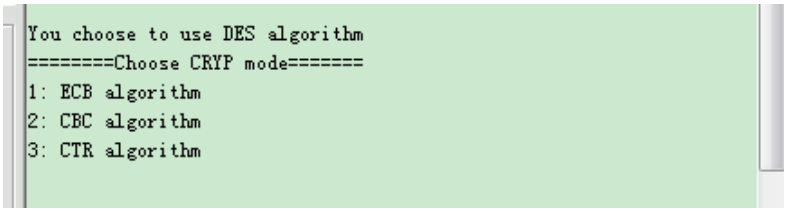
GD32207I-EVAL-V1.0 development board CAU hardware circuits are integrated within the processor, no external circuit is needed.

5.15.3. DEMO Implementation Result

Download the program to the development board, serial port output information, as shown in the following figure.



Plaintext data value, and the encryption algorithm can be selected are shown. After the user setting the algorithm according to the serial output information indicating, serial port will print out selectable mode, as shown below.



After selection mode, the program starts encryption and decryption operations, the results through the serial port to print, and return to the start for user to select a different algorithm and mode repeat Demo. As shown below。

5.16. Hash Acceleration Unit

5.16.1. DEMO Purpose

GD32207I-EVAL-V1.0 development board integrated HAU（Hash Acceleration Unit）,The HASH Acceleration Unit supports acceleration of SHA-1, SHA-224, SHA-256, MD5 algorithm and the HMAC (keyed-hash message authentication code) algorithm, which called the SHA-1, SHA-224, SHA-256 or MD5 hash function to calculate key, message, digest three times. This Demo is used to demonstrate the function and usage of the HAU module in the GD32207I-EVAL-V1.0 development board.

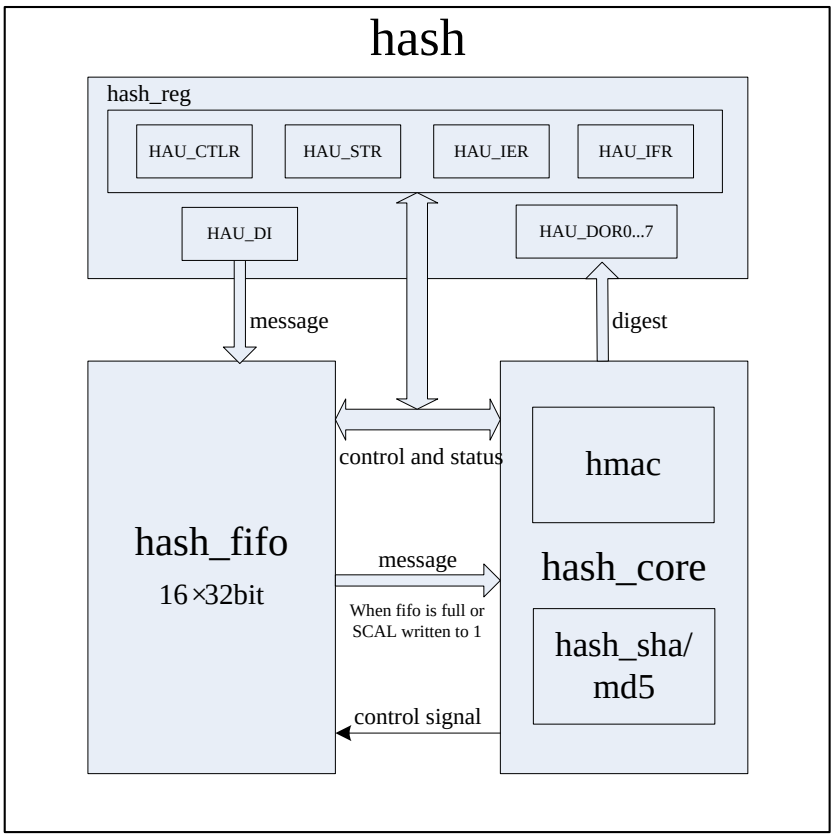
GD32F20X HAU main features:

- Supports SHA-1, SHA-224 and SHA-256 algorithms, compliant with FIPS PUB 180-2 (Federal Information Processing Standards Publication 180-2)
- Supports MD5 compliant with IETF RFC 1321 (Internet Engineering Task Force Request For Comments number 1321)
- Supports HMAC (keyed-hash message authentication code) algorithm
- Automatic swapping to comply with the big-endian or little-endian for SHA-1, SHA-224 and SHA-256 algorithms.
- Automatic padding to fit modulo 512.
- Support DMA mode for input data flow.

5.16.2. DEMO Principle

The block diagram of the hash processor shown as follows.

Figure 5-45 Block Diagram



The hash_reg contains the register of hash processor, which configured by AHB bus. The HAU_IER is interrupt enable register Setting the mask bit to 1 enables the interrupts. The HAU_IFR is status register These two registers generator interrupt signal to CPU NVIC. The HAU_CTLR is HASH control registers. The HAU_STR is HASH start registers. The HAU_DOR0…DOR7 are HASH digest registers which store hash digest.

The hash FIFO is a 32X16 bits FIFO to store message. When write to the HASH_DI register, the data is written to hash_fifo. If FIFO is full then hash_core module is ready to calculate the hash algorithm, the receive data is read to hash_core.

The hash_core module is the calculation module to process hash algorithm. The hash_core module get message from hash_fifo module, and control register from hash_reg module. After calculation, the hash_core send digest and flag to hash_reg module.

GD32207I-EVAL-V1.0 development board HAU hardware circuits are integrated within the processor, no external circuit is needed.

5.16.3. DEMO Implementation Result

Download the program to the development board, serial port output information, as shown in the following figure.

```
========Choose HASH algorithm=======
1: SHA1 algorithm
2: SHA224 algorithm
3: SHA256 algorithm
4: MD5 algorithm
```

The user can enter the corresponding algorithm number based on the needs, the serial output as shown below

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

The user continue to enter the corresponding serial number, select the mode, after the serial output message and key data, and the output of the algorithm user selected, mode and the digest of the calculation.

```
You choose to use HASH mode

 ====================================
 ====    HASH Example    ====
 ====================================
 ----------------------------------------
 Text to be Hashed (254 bits):

 ----------------------------------------
The GD32 F2 series is the result of a perfect symbiosis of the real-time control capabilities of an MCU and the
signal  processing performance of a DSP, and thus complements the GD32 portfolio with a new class of devices,
digital signal controllers (DSC).

 ----------------------------------------

 HMAC Key ( 248 bits):

 ----------------------------------------

The hash processor is a fully compliant implementation of the secure hash algorithm (SHA-1), the MD5 (message-
digest algorithm 5) hashalgorithm and the HMAC (keyed-hash message authentication code) algorithm suitable for a
variety of applications.
 ----------------------------------------
 Algorithm: SHA1   Mode: HASH
 SHA1 Message Digest (160 bits):

 ----------------------------------------
 H0 = [0x749190ea]
 H1 = [0xec3511f6]
 H2 = [0x04a2dc76]
 H3 = [0x58132a09]
 H4 = [0x8a8770cc]

 Example restarted...
```

5.17. Random number generator (RNG)

5.17.1. DEMO Purpose

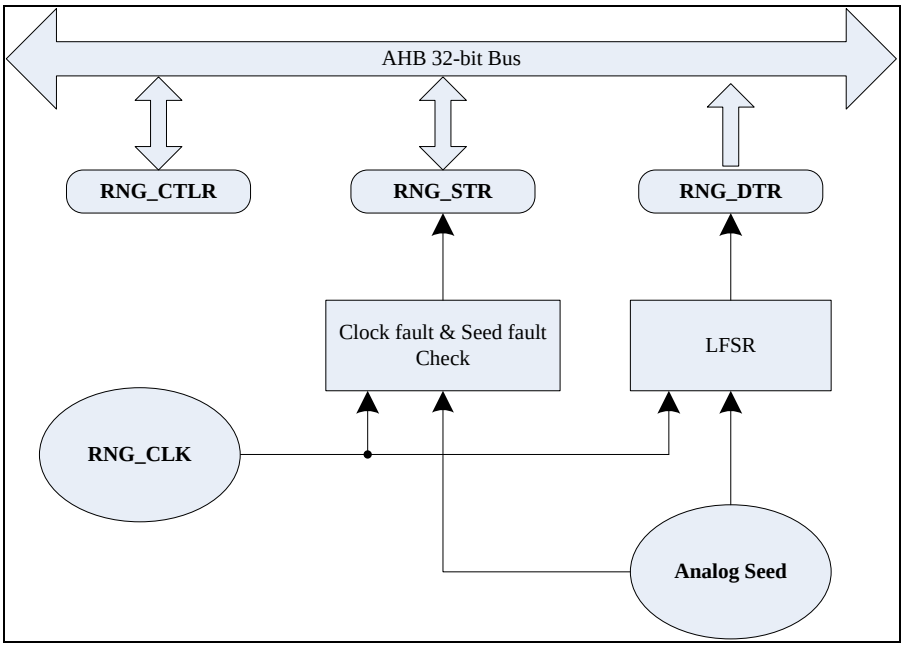
GD32207I-EVAL-V1.0 development board integrated RNG(Random number generator),The random number generator (RNG) module can generate a 32-bit value using continuous analog noise. This Demo is used to demonstrate the function and usage of the RNG module in the GD32207I-EVAL-V1.0 development board.

GD32F20X RNG main features:

- About 40 period PLL clock consumed between two consecutive random numbers
- Disable RNG module will reduce the chip power consumption
- 32-bit random value seed is generated from analog noise

5.17.2. DEMO Principle

Figure 5-46 RNG Block Diagram



The random number seed comes from analog circuit. The analog seed signal output to a linear feedback shift register (LFSR) and in that block will generate a 32-bit width random number.

The analog seed is generated by several ring oscillators whose outputs are XORed. The LFSR is driven by a configurable PLLCLK clock, so that the quality of the random number is independent of the HCLK frequency.

The 32-bit value of LFSR will transfer into RNG_DTR register after a significant number of

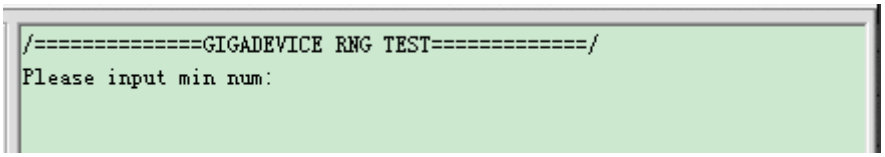
seeds have been entered into the LFSR.

At the same time, the analog seed and PLLCLK clock are monitored. When the analog seed occurs fault or the PLLCLK clock occurs fault, the corresponding status bit in RNG_STR will assert and an interrupt is generated if the interrupt enable control bit is enabled.

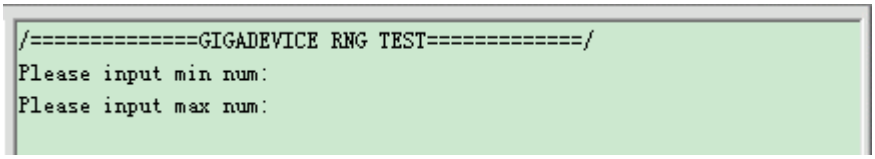
GD32207I-EVAL-V1.0 development board RNG hardware circuits are integrated within the processor, no external circuit is needed.

5.17.3. DEMO Implementation Result

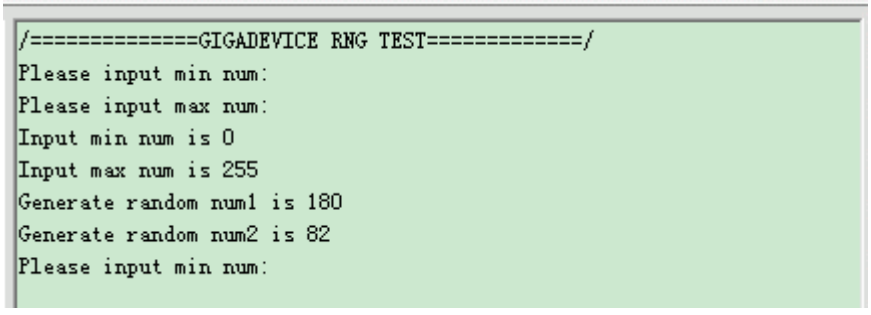
Download the program to the development board, serial port output information, as shown in the following figure.



The user can enter any number between 0x00 to 0xff in hexadecimal form as the minimum value of random numbers generated by the serial port. After entering the complete diagram is shown below.



Then the user can enter any number in the same range as the maximum value to generate random numbers, the program will show the minimum and maximum values just entered, randomly generates and show two random numbers based on the input range by serial output.



5.18. TLDI_without_GUI

5.18.1. DEMO Purpose

GD32207I-EVAL-V1.0 board integrate TLDI(TFT display interface). The TFT(LCD) display interface provides a parallel digital RGB (Red, Green, Blue) and signals for horizontal,

vertical synchronization, Pixel Clock and Data Enable as output to interface directly to a variety of LCD(Liquid Crystal Display) and TFT(Thin Film Transistor) panels. This Demo show TLDI features and using method.

GD32F20X TLDI main features:

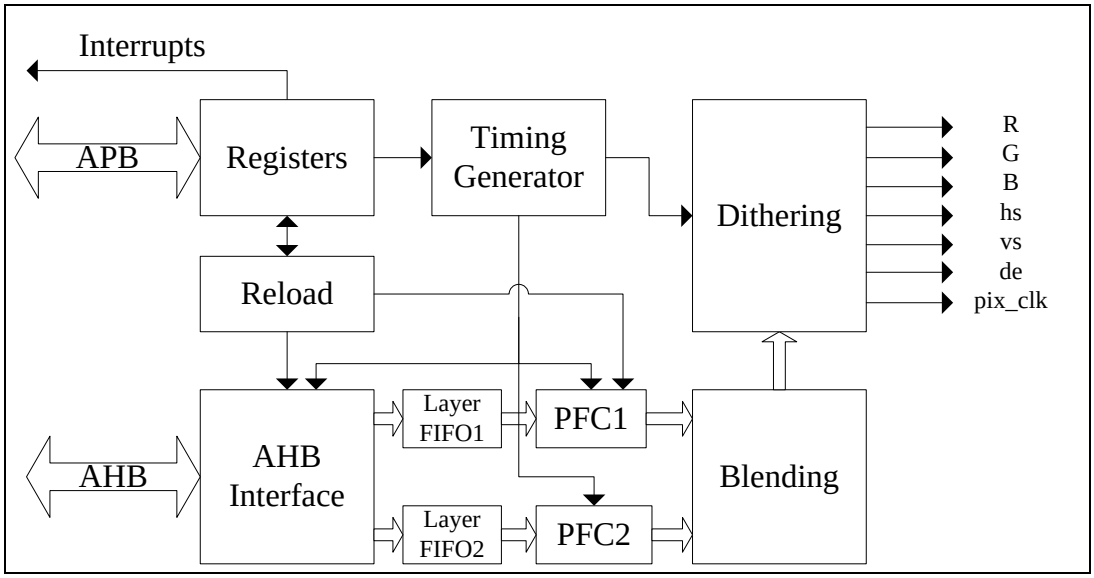
- 24-bit RGB Parallel Pixel Output; 8 bits-per-pixel (RGB888)
- 2 display layers with dedicated FIFO (64x32-bit)
- Color Look-Up Table (CLUT) up to 256 color (256x24-bit) per layer
- Supports up to SVGA (800x600) resolution
- Programmable timings for different display panels
- Programmable Background color
- Programmable polarity for HSync, VSync and Data Enable
- Up to 8 Input color formats selectable per layer
- Pseudo-random dithering output for low bits per channel
- Flexible blending between two layers using alpha value (per pixel or constant)
- Color Keying (transparency color)
- Programmable Window position and size
- Image size up to 800x600
- Pixel Clock as fast as HCLK when one layer enable in ARGB format.

5.18.2. DEMO Principle

TLDI provides a 24-bit RGB Parallel display interface, which is shown in table below.

Direction	Name	Width	Description
O	LCD_HSYNC	1	Horizontal Synchronous
O	LCD_VSYNC	1	Vertical Synchronous
O	LCD_DE	1	Data Enable
O	LCD_CLK	1	Pixel Clock
O	LCD_R[7:0]	8	Pixel Red Data
O	LCD_G[7:0]	8	Pixel Green Data
O	LCD_B[7:0]	8	Pixel Blue Data

Figure 5-47 TLDI module block diagram



The TLDI contains these modules: Layer FIFO: One FIFO 64x32 bit per layer, PFC: Pixel Format Convertor performing the pixel format conversion from the selected input pixel format of a layer to words, AHB interface: For data transfer from memories to the FIFO, Blending, Dithering unit and Timings Generator. Figure above shows the block diagram of the TLDI module.

Different LCD has its specific synchronous time sequence, window effective area configure method is as follows:

Layer_WindowRightPos= (Offset_X + Hsync + HBP);

Layer_WindowLeftPos= (Offset_X + Hsync + HBP + Window_Width - 1);

Layer_WindowBottomPos= (Offset_Y + Vsync + VBP);

Layer_WindowTopPos= (Offset_Y + Vsync + VBP + Window_Heigh - 1);

Window_Width and Window_Heigh should be configured depending on picture size displaying.

This Demo configure first layer all below.

TLDI_Layer_InitStruct.Layer_WindowRightPos = 42;

TLDI_Layer_InitStruct.Layer_WindowLeftPos = (480 + 42 - 1);

TLDI_Layer_InitStruct.Layer_WindowBottomPos = 12;

TLDI_Layer_InitStruct.Layer_WindowTopPos = (272+12 - 1);

The picture will be displayed as background.

This Demo configure second layer all below.

TLDI_Layer_InitStruct.Layer_WindowRightPos =110;

TLDI_Layer_InitStruct.Layer_WindowLeftPos = (140+ 110-1);

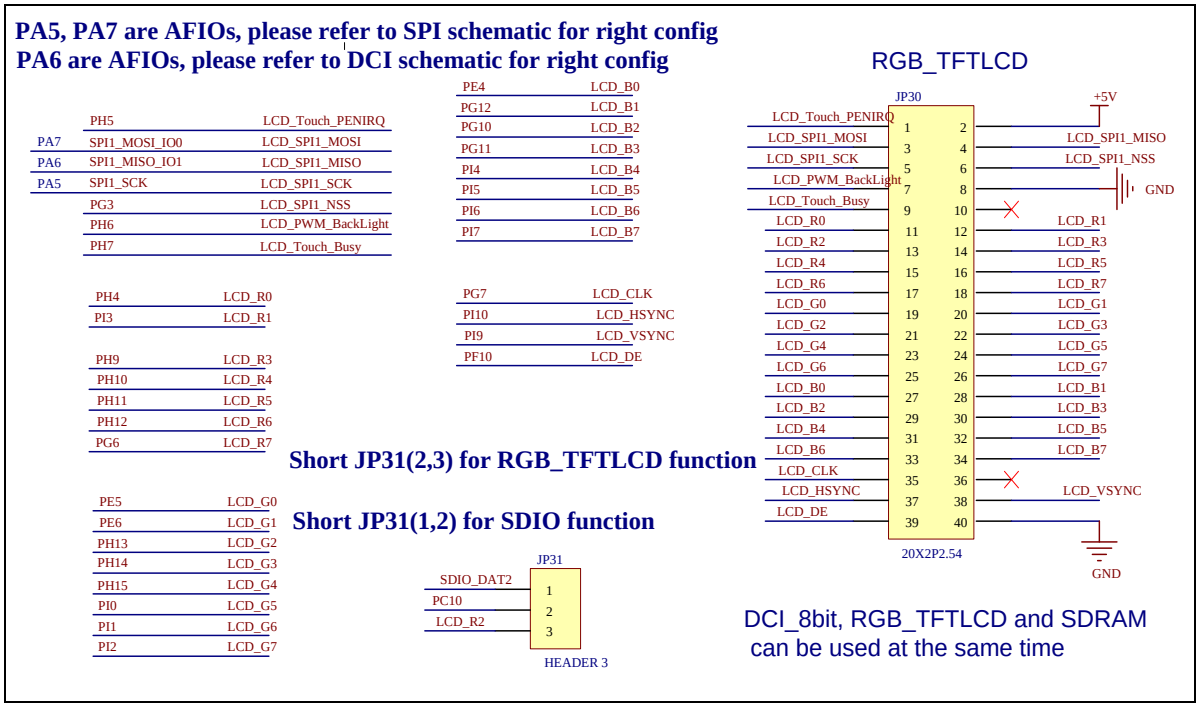
TLDI_Layer_InitStruct.Layer_WindowBottomPos = 100;

TLDI_Layer_InitStruct.Layer_WindowTopPos = (60+ 100-1);

The window effective area is consistent with the size of the dynamic images. User can depend need to change the related parameters, such as transparency and default RGB values. Specific can consult routines.

This example hardware principle diagram is shown as following.

Figure 5-48 Schematic diagram of TLDI RGB-LCD function



note：Configure JP31 LCD_R2。

5.18.3. DEMO Implementation Result

After downloading program to board, LCD appear a running cheetah on the background of GD logo，output as following.

5.19. TAMPER and Waveform Detection

5.19.1. DEMO Purpose

GD32207I-EVAL-V1.0 development board integrates TAMPER and Waveform Detection function, it includes two TAMPER sources and two Waveform Detection, and it can be controlled by the BKP register in Backup domain. The Backup registers are located in the Backup domain that remains powered-on by V_{BAT} even if V_{DD} power is shut down, they are forty two 16-bit (84 bytes) registers for data protection of user application data, and the wake-up action from Standby mode or system reset are not affect these registers.

GD32F20X BKP main features:

- 84 bytes Backup registers which can keep data under power saving mode. If tamper event is detected, Backup registers will be reset
- The active level of Tamper source (PC13 and PI8) can be configured
- RTC Clock Calibration register provides RTC alarm and second output selection, and sets the calibration value
- Tamper interrupt event register (BKP_TIER) can control tamper detection and waveform detection with interrupt or event capability
- Two square waveform detection on PC13->PI8 or PC14->PC15

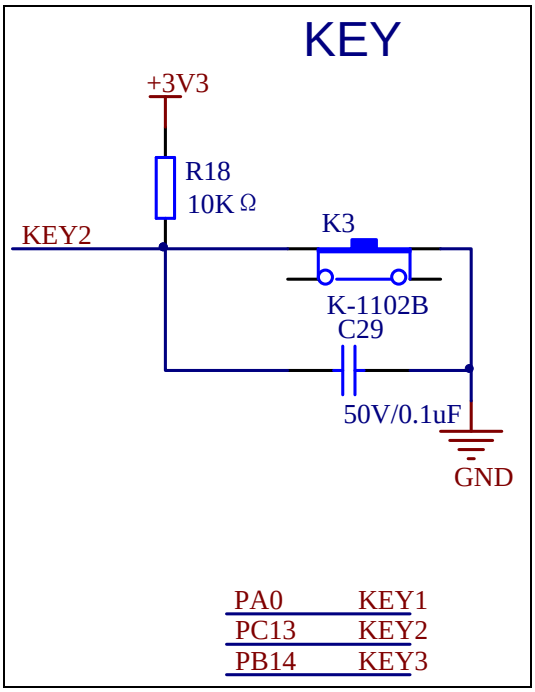
5.19.2. DEMO principle

In order to protect the important user data, the MCU provides the tamper detection function, and it can be independently enabled on TAMPER_1 pin (PC13)/ TAMPER_2 pin (PI8) by setting corresponding TPE_1/ TPE_2 bit in the BKP_TPCR1/ BKP_TPCR2 register. To prevent the tamper event from losing, the edge detection is logically ANDed with the TPE_1/

TPE_2 bit, used for tamper detection signal. So the tamper detection configuration should be set before enable TAMPER_1/ TAMPER_2 pin. When the tamper event is detected, the corresponding TEF_1/ TEF_2 bit in the BKP_TIER register will be set. Tamper event can generate an interrupt if tamper interrupt is enabled. Any tamper event will reset all Backup data registers.

In this Demo, the TAMPER1 (PC13) is selected. It used the K3 trigger, and the detection active level is set to low.

Figure 5-49 Schematic diagram of KEY



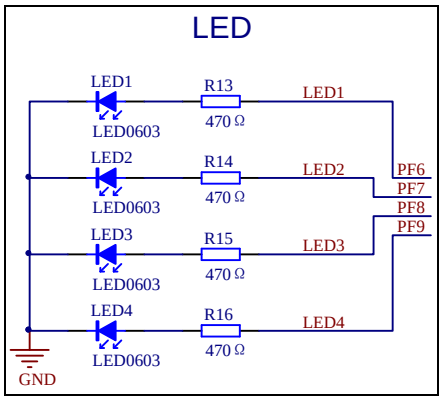
There are two waveform detect. Send a square waveform on PC13 if TPM_1 bit is set or on PC14 if TPM_2 bit is set. Receive and check square waveform on PI8 if TPM_1 bit set or on PC15 if TPM_2 bit is set. When the check is wrong, the corresponding TEF_2 bit in the BKP_TIER register will be set. Tamper event can generate an interrupt if tamper interrupt is enabled. Any tamper event will reset all Backup data registers.

In this Demo, the second Waveform detection is selected. Since PC14 and PC15 are used in this program, the LSI is used instead of LSE when the RTC clock source is configured.

Demo program has two functions: TAMPER1 and the second Waveform Detection, it is selected by main.h file, and by means of conditional compilation in program. TAMPER1 using PC13 to trigger TAMPER interrupt to clear the BKP data, and turn on LED. The second Waveform using PC14 and PC15 were receive and check square waveform, if the detection is not successful, it will trigger TAMPER interrupt to clear the BKP data, and turn on LED.

This Demo uses the LEDs and the corresponding GPIO pins as shown below:

Figure 5-50 Schematic diagram of LED



Jumper: Battery is not installed, JP0 will jump to +3.3V using an external power supply. When installing the battery, the JP0 jumps to the Bat to use the battery power supply.

5.19.3. DEMO Implementation Result

After downloading the program to the development board, the data will be written to the backup data register, if the write successfully, LED1 on, otherwise, LED2 on. If there is TAMPER or Waveform detection error is triggered, LED3 on, LED4 off, otherwise, LED3 and LED4 are all off.

5.20. USB OTG_FS virtual mouse

5.20.1. DEMO Purpose

GD32207I-EVAL-V1.0 board support USB Host and USB Device with its own USB OTG FS (FS, Full Speed, 12Mbps).This Demo has been realized a USB HID Device by using USB OTG FS to simulate a USB Mouse.

5.20.2. DEMO Principle

USB, Universal Serial BUS, Is an external bus standard used to regulate the connection and communication between the computer and the external device.USB interface supports device plug and play and hot plugging functions.

USB's development has experienced a number of versions of USB1.0/1.1/2.0/3.0/3.1。At present, the most used is USB 2.0, and the USB 3.0 is also becoming more and more popular. GD32207I's conforms to USB 2.0 specification.

Standard USB is composed of four lines, in addition to VCC/GND, the other are D+ and D-, the two data lines, which use the differential voltage mode for data transmission. On the USB host, D+ and D- are all connected the 15K resistance to ground, so the D- and D+ are low when there is no device to connect. And on the USB peripheral, if it is full-speed device, it will

connect a 1.5K resistor to VCC on the D+, and if low-speed device, the connection will be on the D-. When the device is connected to the host, the host can judge whether there is a device to access, and to determine the device is full-speed or low-speed.

USB OTG is USB On-The-Go, means USB is in the process. USB OTG makes USB get rid of the limitation of the original master-slave architecture, and realizes the transmission mode of the end to end. USB OTG standard is fully compatible with the USB2.0 standard, and add limited host ability and the role exchanging function, which allows the device to operate at times as a host and at times as a peripheral (OTG dual role function). OTG dual role function device is fully in line with the USB2.0 standard and can provide limited host ability. It supports the host negotiation protocol (HNP) and the session request protocol (SRP). In OTG, the initial host is called the A-device, the initial peripheral is known as the B-device. It mainly use cable connection mode to determine the initial role，the mini-AB receptacle which use is added the ID pin to identify different cable ends. Mini-A plug ID pin is connected to ground and mini-B plug ID pin is in floating. When the OTG device detected the ID pin is connected to ground, the default device is an A-device (the host) and the detection of the floating ID pin device is considered to be a B-Device (the peripheral). Once the system is connected, the OTG host and peripherals can realize the role exchange by using HNP protocol. A-device as the default host will provide VBus power, reset bus, enumerate and configure B-device in the detection of the device is connected. Session request protocol (SRP) allows the B device to open the VBus power and start a session on the A device. An OTG session can be determined by the time of the A- device to provide VBus power (Note: the A device is always powered for VBus, even is set as a peripheral). It can also be used to shut down the VBus power supply to save power, which is very important in battery powered products. Next, turn to the GD32207I USB OTG controller.

The USB OTG FS of the GD32207I is a dual-role device (DRD) controller that supports both device and host functions and is fully compliant with the On-The-Go Supplement to the USB 2.0 Specification. It can also be configured as a host-only or device-only controller, fully compliant with the USB 2.0 Specification. In host mode, the OTG_FS supports full-speed (FS, 12M bits/s) and low-speed (LS, 1.5M bits/s) transfers whereas in device mode, it only supports full-speed (FS, 12M bits/s) transfers. The OTG_FS supports both HNP and SRP.

The main features of GD32207I OTG FS include three categories: general, host-mode and device-mode features.

General features

- Fully compliant with the On-The-Go Supplement to the USB 2.0 Specification
- In PHY, it includes fully support for the optional protocol detailed in the On-The-Go Supplement Rev 1.3 specification
  - Supports the insertion A-B type device identification (USB ID line)
  - Supports Host Negotiation Protocol (HNP) and Session Request Protocol (SRP)
  - Allows host to turn V_{BUS} off to conserve battery power in OTG applications

  - Supports V_{BUS} level detection with internal comparators
  - Supports dynamic switching between host and device roles
- It can be configured by software to operate as:
  - USB OTG_FS dual role device (host or device)
  - USB FS/LS host (A-device)
  - USB FS Peripheral (B-device)
- It supports SOF (at FS) and keep-alive (at LS) by
  - SOF pulse PAD connectivity
  - SOF pulse internal connection to TIMER2
  - Configurable framing period
  - Configurable end of frame (EOF) interrupt
- It provides a dedicated RAM of 1.25 KB with advanced FIFO control for flexible and efficient use of RAM:
  - Configurable partitioning of RAM space into different FIFOs
  - Each FIFO can hold multiple packets
  - Dynamic and contiguous memory allocation
- It guarantees max USB bandwidth for up to 1 frame via hardware and needs no system intervention
- It provides power saving features during USB suspend:
  - Stop system
  - Switch off clock domains internal to the digital core, PHY and FIFO power management
- It supports suspend and resume

Host-mode features

- Needs an external charge pump or 5V power for V_{BUS} voltage generation
- Provides one port which is able to deliver a minimum of 100mA for a configured or un-configured device, and optionally, up to 500mA for a configured device
- Up to 8 host channels: each channel is dynamically configured in control, interrupt, bulk or isochronous transfer type
- Built-in hardware scheduler, it holds two hardware queues:
  - Up to 8 periodic transfer (interrupt or isochronous) requests in the periodic

hardware queue

– Up to 8 non-periodic transfer (control or bulk) requests in the non-periodic hardware queue

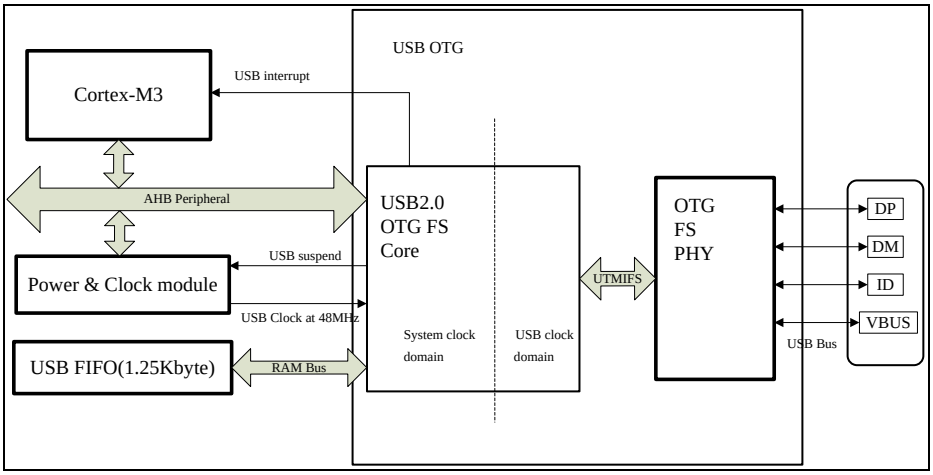
- In order to use the USB data RAM efficiently, it is allocated as a shared Rx FIFO, a periodic Tx FIFO and a non-periodic Tx FIFO to manage

Device-mode features

- Draws 100mA or less from the bus before configuration
- Can draw up to 500mA from the bus after successful negotiation with the host
- 1 bidirectional control endpoint (endpoint 0)
- 3 IN endpoints and 3 OUT endpoints configurable to support bulk, interrupt or isochronous transfers
- Management of a shared Rx FIFO and up to 4 dedicated Tx FIFOs (one for each active IN endpoint) for efficient usage of the USB data RAM and less load on the application
- Support the soft disconnect feature
- Can be bus-powered or self-powered

GD32207I USB OTG FS block diagram as shown in Figure：

Figure 5-51 USB OTG FS block diagram



The GD32207I access the OTG FS USB function module through the AHB bus (AHB frequency must be greater than 16MHz). USB 48MHz clock is get from the PLL by frequency division.

About other GD32207I USB OTG FS introduction, please refer to the “ARM Cortex-M3 32-bit MCU user manual”, the twenty-third chapter, here is no longer a detailed introduction.

In order to use the GD32207I USB normally, you have to write USB driver, and the entire USB communication process is very complicated, need to understand the entire USB

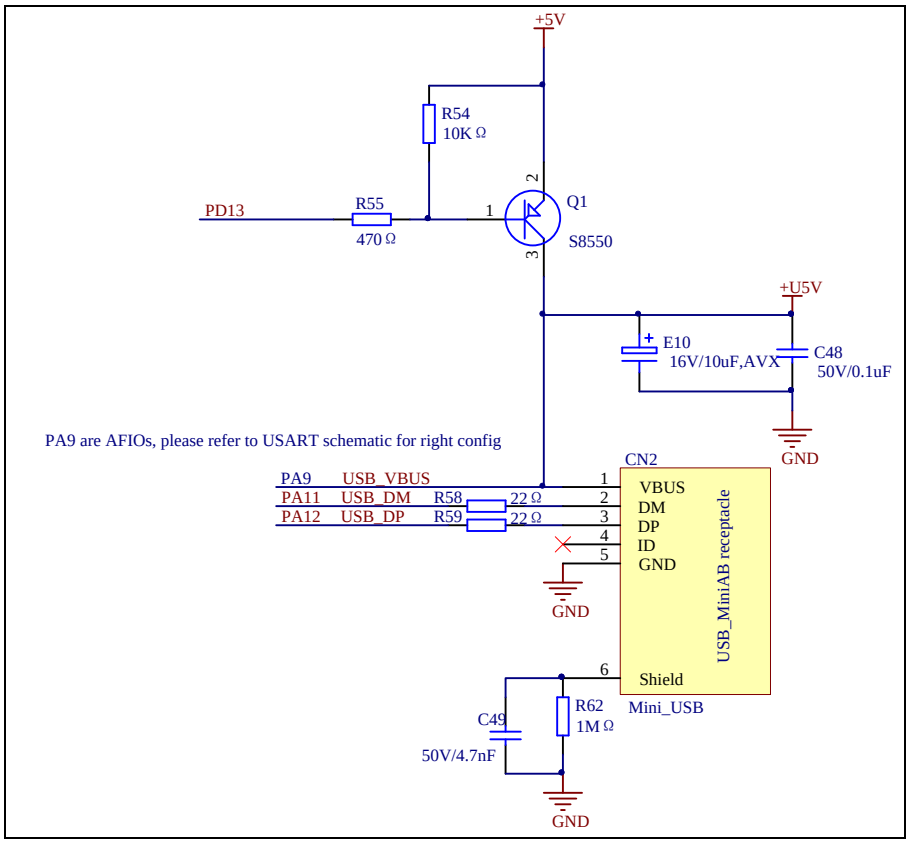
protocol stack and the specific device class protocol. It can't be described in detail here, for more details please refer to the USB 2.0 protocol standards. Of course, GD provides a complete USB OTG FS driver (including the host and the device), through this library can easily achieve the functions of the various USB Demo, without the need to understand the entire drive USB, greatly reducing the development time and effort. This drive library can be downloaded from the official website of GD.

This Demo mainly implements a USB mouse. USB mouse is very widely used HID devices currently. HID, the Human Interface Device, is the device which computer directly interacting with human, including keyboard, mouse and joystick, etc… However, the HID device does not necessarily have a human interface, as long as it conforms to the HID class specification.

USB host understands the USB device through a series of descriptors, for some type of device class, there will be some special descriptors. For HID devices, the most important special descriptor is the report descriptor. For information about the HID device class, please refer to the HID 1.11 protocol.

The GD32207I USB OTG FS interface schematic diagram is shown in the following diagram, in the device mode it support from the USB interface to take electricity, in the host mode it control USB device power supply through the PD13.

Figure 5-52 Schematic diagram of USB



The USB OTG is only used as a device in this Demo, so it just takes electricity from the USB interface.

Jumper settings: JP5 whether jump to the end of the USB, according to the USB OTG global configuration register NOVBUSSENS bit to decide: if NOVBUSSENS bit is 1, then JP5 no need to jump to the end of the USB. At this time USB VBUS wire not connected to the PA9; if NOVBUSSENS bit is 0, then jp5 need to jump to the end of the USB. At this time the USB VBUS wire connected to the PA9. The role of the NOVBUSSENS bit, please refer to the user manual.

Due to the Demo using the LCD print the USB running log information, so also are LCD related jumper settings, and the cache of the LCD using SDRAM, so also SDRAM related jumper settings. Please refer to the related sections about these 2 modules in this manual.

5.20.3. DEMO implementation result

After the Demo compiled and download, PC can detect the HID device, because the Windows comes with a born USB mouse driver, so there is no need to manually install the driver. Next, you can find a new added USB mouse device in the computer equipment manager. At the same time, you can see log about device enumeration on the LCD.
When tested, the results are as follows:
When the Wakeup key is pressed on the GD32207I-EVAL-V1.0 board, the mouse pointer moves towards left on the computer screen;
When the Tamper key is pressed on the GD32207I-EVAL-V1.0 board, the mouse pointer moves towards right on the computer screen;
When the User key is pressed on the GD32207I-EVAL-V1.0 board, the mouse pointer moves towards up on the computer screen.

5.21. USB OTG_FS virtual U disk

5.21.1. DEMO Purpose

GD32207I-EVAL-V1.0 board support USB Host and USB Device with its own USB OTG FS (FS, Full Speed, 12Mbps).This Demo has been realized a USB MSC Device by using USB OTG FS to simulate a U disk.

5.21.2. DEMO Principle

USB, Universal Serial BUS, Is an external bus standard used to regulate the connection and communication between the computer and the external device.USB interface supports device plug and play and hot plugging functions.

USB's development has experienced a number of versions of USB1.0/1.1/2.0/3.0/3.1。At present, the most used is USB 2.0, and the USB 3.0 is also becoming more and more popular. GD32207I's conforms to USB 2.0 specification.

Standard USB is composed of four lines, in addition to VCC/GND, the other are D+ and D-,

the two data lines, which use the differential voltage mode for data transmission. On the USB host, D+ and D- are all connected the 15K resistance to ground, so the D- and D+ are low when there is no device to connect. And on the USB peripheral, if it is full-speed device, it will connect a 1.5K resistor to VCC on the D+, and if low-speed device, the connection will be on the D-. When the device is connected to the host, the host can judge whether there is a device to access, and to determine the device is full-speed or low-speed.

USB OTG is USB On-The-Go, means USB is in the process. USB OTG makes USB get rid of the limitation of the original master-slave architecture, and realizes the transmission mode of the end to end. USB OTG standard is fully compatible with the USB2.0 standard, and add limited host ability and the role exchanging function, which allows the device to operate at times as a host and at times as a peripheral (OTG dual role function). OTG dual role function device is fully in line with the USB2.0 standard and can provide limited host ability. It supports the host negotiation protocol (HNP) and the session request protocol (SRP). In OTG, the initial host is called the A-device, the initial peripheral is known as the B-device. It mainly use cable connection mode to determine the initial role，the mini-AB receptacle which use is added the ID pin to identify different cable ends. Mini-A plug ID pin is connected to ground and mini-B plug ID pin is in floating. When the OTG device detected the ID pin is connected to ground, the default device is an A-device (the host) and the detection of the floating ID pin device is considered to be a B-Device (the peripheral). Once the system is connected, the OTG host and peripherals can realize the role exchange by using HNP protocol. A-device as the default host will provide VBus power, reset bus, enumerate and configure B-device in the detection of the device is connected. Session request protocol (SRP) allows the B device to open the VBus power and start a session on the A device. An OTG session can be determined by the time of the A- device to provide VBus power (Note: the A device is always powered for VBus, even is set as a peripheral). It can also be used to shut down the VBus power supply to save power, which is very important in battery powered products. Next, turn to the GD32207I USB OTG controller.

The USB OTG FS of the GD32207I is a dual-role device (DRD) controller that supports both device and host functions and is fully compliant with the On-The-Go Supplement to the USB 2.0 Specification. It can also be configured as a host-only or device-only controller, fully compliant with the USB 2.0 Specification. In host mode, the OTG_FS supports full-speed (FS, 12M bits/s) and low-speed (LS, 1.5M bits/s) transfers. Whereas in device mode, it only supports full-speed (FS, 12M bits/s) transfers. The OTG_FS supports both HNP and SRP.

The main features of GD32207I OTG FS include three categories: general, host-mode and device-mode features.

General features

- Fully compliant with the On-The-Go Supplement to the USB 2.0 Specification
- In PHY, it includes fully support for the optional protocol detailed in the On-The-Go Supplement Rev 1.3 specification
  - Supports the insertion A-B type device identification (USB ID line)

  - Supports Host Negotiation Protocol (HNP) and Session Request Protocol (SRP)
  - Allows host to turn V_{BUS} off to conserve battery power in OTG applications
  - Supports V_{BUS} level detection with internal comparators
  - Supports dynamic switching between host and device roles
- It can be configured by software to operate as:
  - USB OTG_FS dual role device (host or device)
  - USB FS/LS host (A-device)
  - USB FS Peripheral (B-device)
- It supports SOF (at FS) and keep-alive (at LS) by
  - SOF pulse PAD connectivity
  - SOF pulse internal connection to TIMER2
  - Configurable framing period
  - Configurable end of frame (EOF) interrupt
- It provides a dedicated RAM of 1.25 KB with advanced FIFO control for flexible and efficient use of RAM:
  - Configurable partitioning of RAM space into different FIFOs
  - Each FIFO can hold multiple packets
  - Dynamic and contiguous memory allocation
- It guarantees max USB bandwidth for up to 1 frame via hardware and needs no system intervention
- It provides power saving features during USB suspend:
  - Stop system
  - Switch off clock domains internal to the digital core, PHY and FIFO power management
- It supports suspend and resume

Host-mode features

- Needs an external charge pump or 5V power for V_{BUS} voltage generation
- Provides one port which is able to deliver a minimum of 100mA for a configured or un-configured device, and optionally, up to 500mA for a configured device
- Up to 8 host channels: each channel is dynamically configured in control, interrupt, bulk or isochronous transfer type

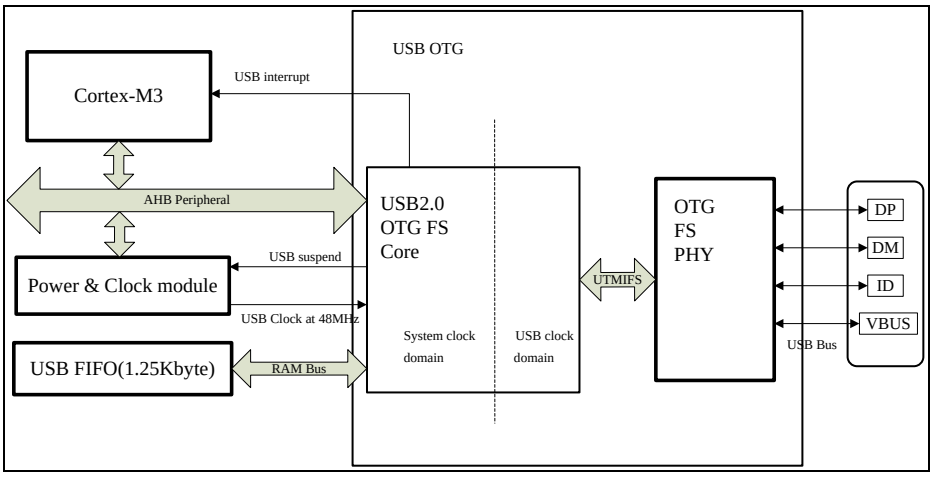
- Built-in hardware scheduler, it holds two hardware queues:
  - Up to 8 periodic transfer (interrupt or isochronous) requests in the periodic hardware queue
  - Up to 8 non-periodic transfer (control or bulk) requests in the non-periodic hardware queue
- In order to use the USB data RAM efficiently, it is allocated as a shared Rx FIFO, a periodic Tx FIFO and a non-periodic Tx FIFO to manage

Device-mode features

- Draws 100mA or less from the bus before configuration
- Can draw up to 500mA from the bus after successful negotiation with the host
- 1 bidirectional control endpoint (endpoint 0)
- 3 IN endpoints and 3 OUT endpoints configurable to support bulk, interrupt or isochronous transfers
- Management of a shared Rx FIFO and up to 4 dedicated Tx FIFOs (one for each active IN endpoint) for efficient usage of the USB data RAM and less load on the application
- Support the soft disconnect feature
- Can be bus-powered or self-powered

GD32207I USB OTG FS block diagram as shown in Figure：

Figure 5-53 USB OTG FS block diagram



The GD32207I access the OTG FS USB function module through the AHB bus (AHB frequency must be greater than 16MHz). USB 48MHz clock is get from the PLL by frequency division.

About other GD32207I USB OTG FS introduction, please refer to the “ARM Cortex-M3 32-bit MCU user manual”, the twenty-third chapter, here is no longer a detailed introduction.

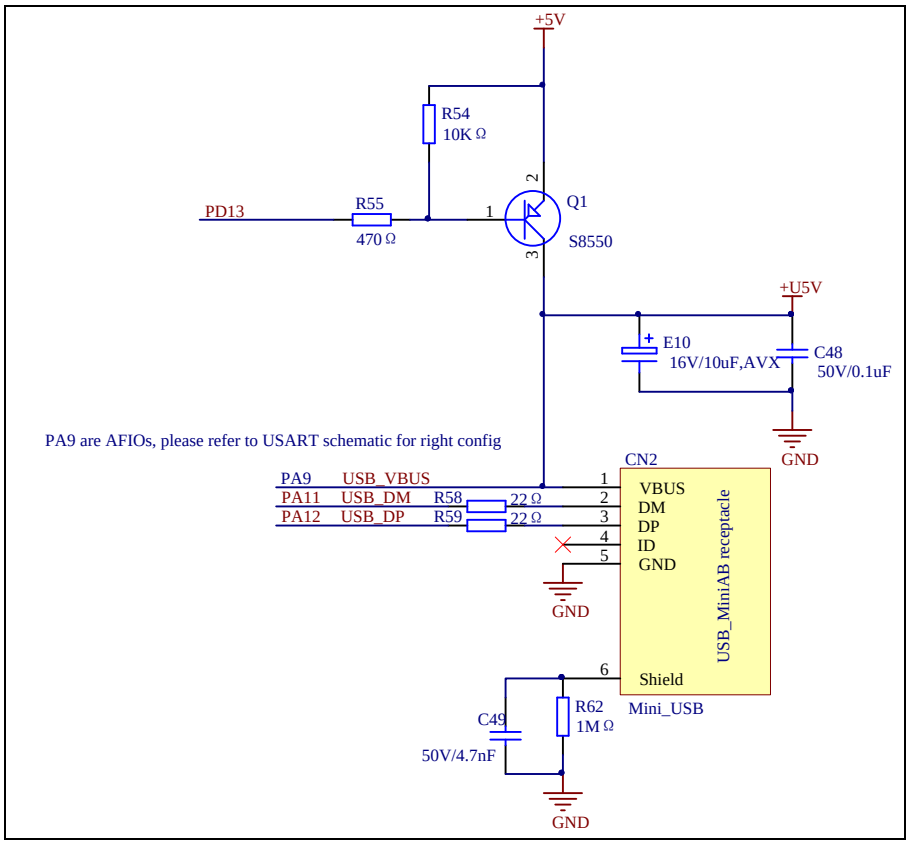
In order to use the GD32207I USB normally, you have to write USB driver, and the entire USB communication process is very complicated, need to understand the entire USB protocol stack and the specific device class protocol. It can’t be described in detail here, for more details please refer to the USB 2.0 protocol standards. Of course, GD provides a complete USB OTG FS driver (including the host and the device), through this library can easily achieve the functions of the various USB Demo, without the need to understand the entire drive USB, greatly reducing the development time and effort. This drive library can be downloaded from the official website of GD.

This Demo mainly implements a U disk. U disk is very widely used removable MSC devices currently. 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 and internal Flash as the storage medium. For more details of the MSC protocol please refer to the MSC protocol standard by yourself.

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.

The GD32207I USB OTG FS interface schematic diagram is shown in the following diagram, in the device mode it support from the USB interface to take electricity, in the host mode it control USB device power supply through the PD13.

Figure 5-54 Schematic diagram of USB



The USB OTG is only used as a device in this Demo, so it just takes electricity from the USB interface.

Jumper settings: JP5 whether jump to the end of the USB, according to the USB OTG global configuration register NOVBUSSENS bit to decide: if NOVBUSSENS bit is 1, then JP5 no need to jump to the end of the USB. At this time USB VBUS wire not connected to the PA9; if NOVBUSSENS bit is 0, then jp5 need to jump to the end of the USB. At this time the USB VBUS wire connected to the PA9. The role of the NOVBUSSENS bit, please refer to the user manual.

Due to the Demo using the LCD print the USB running log information, so also are LCD related jumper settings, and the cache of the LCD using SDRAM, so also SDRAM related jumper settings. Please refer to the related sections about these 2 modules in this manual.

5.21.3. DEMO Implementation Result

After the Demo compiled and download, user need connect the device to the PC with USB cable. Next, PC can detect the MSC device, because the Windows comes with a born USB MSC driver, so there is no need to manually install the driver. After Demo running, you can see log information about device enumeration on the LCD. When you open the computer equipment manager, you will find a USB large capacity storage device is in the universal serial bus controller, and there are 2 more disk drives.

Then, after opening the resource manager, you will see more of the 2 disks.

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

5.22. USB OTG_FS virtual ComPort（VCP）

5.22.1. DEMO Purpose

GD32207I-EVAL-V1.0 board support USB Host and USB Device with its own USB OTG FS (FS, Full Speed, 12Mbps).This Demo has been realized a USB CDC Device by using USB OTG FS to simulate a virtual serial port.

5.22.2. DEMO Principle

USB, Universal Serial BUS, Is an external bus standard used to regulate the connection and communication between the computer and the external device.USB interface supports device plug and play and hot plugging functions.

USB's development has experienced a number of versions of USB1.0/1.1/2.0/3.0/3.1。At present, the most used is USB 2.0, and the USB 3.0 is also becoming more and more popular. GD32207I's conforms to USB 2.0 specification.

Standard USB is composed of four lines, in addition to VCC/GND, the other are D+ and D-, the two data lines, which use the differential voltage mode for data transmission. On the USB host, D+ and D- are all connected the 15K resistance to ground, so the D- and D+ all low when there is no device to connect. And on the USB peripheral, if it is full-speed device, it will connect a 1.5K resistor to VCC on the D+, and if low-speed device, the connection will be on the D-. When the device is connected to the host, the host can judge whether there is a device to access, and to determine the device is full-speed or low-speed.

USB OTG is USB On-The-Go, means USB is in the process. USB OTG makes USB get rid of the limitation of the original master-slave architecture, and realizes the transmission mode of the end to end. USB OTG standard is fully compatible with the USB2.0 standard, and add limited host ability and the role exchanging function, which allows the device to operate at times as a host and at times as a peripheral (OTG dual role function). OTG dual role function device is fully in line with the USB2.0 standard and can provide limited host ability. It supports the host negotiation protocol (HNP) and the session request protocol (SRP). In OTG, the initial host is called the A-device, the initial peripheral is known as the B-device. It mainly use cable connection mode to determine the initial role，the mini-AB receptacle which use is added the ID pin to identify different cable ends. Mini-A plug ID pin is connected to ground and mini-B plug ID pin is in floating. When the OTG device detected the ID pin is connected to ground, the default device is an A-device (the host) and the detection of the floating ID pin device is considered to be a B-Device (the peripheral). Once the system is connected, the

OTG host and peripherals can realize the role exchange by using HNP protocol. A-device as the default host will provide VBus power, reset bus, enumerate and configure B-device in the detection of the device is connected. Session request protocol (SRP) allows the B device to open the VBus power and start a session on the A device. An OTG session can be determined by the time of the A- device to provide VBus power (Note: the A device is always powered for VBus, even is set as a peripheral). It can also be used to shut down the VBus power supply to save power, which is very important in battery powered products. Next, turn to the GD32207I USB OTG controller.

The USB OTG FS of the GD32207I is a dual-role device (DRD) controller that supports both device and host functions and is fully compliant with the On-The-Go Supplement to the USB 2.0 Specification. It can also be configured as a host-only or device-only controller, fully compliant with the USB 2.0 Specification. In host mode, the OTG_FS supports full-speed (FS, 12M bits/s) and low-speed (LS, 1.5M bits/s) transfers. Whereas in device mode, it only supports full-speed (FS, 12M bits/s) transfers. The OTG_FS supports both HNP and SRP.

The main features of GD32207I OTG FS include three categories: general, host-mode and device-mode features.

General features

- Fully compliant with the On-The-Go Supplement to the USB 2.0 Specification
- In PHY, it includes fully support for the optional protocol detailed in the On-The-Go Supplement Rev 1.3 specification
  - Supports the insertion A-B type device identification (USB ID line)
  - Supports Host Negotiation Protocol (HNP) and Session Request Protocol (SRP)
  - Allows host to turn V_{BUS} off to conserve battery power in OTG applications
  - Supports V_{BUS} level detection with internal comparators
  - Supports dynamic switching between host and device roles
- It can be configured by software to operate as:
  - USB OTG_FS dual role device (host or device)
  - USB FS/LS host (A-device)
  - USB FS Peripheral (B-device)
- It supports SOF (at FS) and keep-alive (at LS) by
  - SOF pulse PAD connectivity
  - SOF pulse internal connection to TIMER2
  - Configurable framing period
  - Configurable end of frame (EOF) interrupt

- It provides a dedicated RAM of 1.25 KB with advanced FIFO control for flexible and efficient use of RAM:
  - Configurable partitioning of RAM space into different FIFOs
  - Each FIFO can hold multiple packets
  - Dynamic and contiguous memory allocation
- It guarantees max USB bandwidth for up to 1 frame via hardware and needs no system intervention
- It provides power saving features during USB suspend:
  - Stop system
  - Switch off clock domains internal to the digital core, PHY and FIFO power management
- It supports suspend and resume

Host-mode features

- Needs an external charge pump or 5V power for V_{BUS} voltage generation
- Provides one port which is able to deliver a minimum of 100mA for a configured or un-configured device, and optionally, up to 500mA for a configured device
- Up to 8 host channels: each channel is dynamically configured in control, interrupt, bulk or isochronous transfer type
- Built-in hardware scheduler, it holds two hardware queues:
  - Up to 8 periodic transfer (interrupt or isochronous) requests in the periodic hardware queue
  - Up to 8 non-periodic transfer (control or bulk) requests in the non-periodic hardware queue
- In order to use the USB data RAM efficiently, it is allocated as a shared Rx FIFO, a periodic Tx FIFO and a non-periodic Tx FIFO to manage

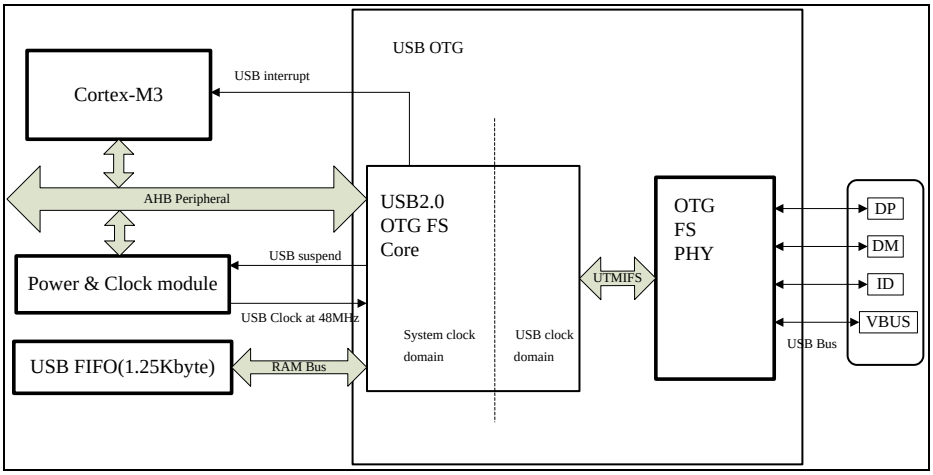
Device-mode features

- Draws 100mA or less from the bus before configuration
- Can draw up to 500mA from the bus after successful negotiation with the host
- 1 bidirectional control endpoint (endpoint 0)
- 3 IN endpoints and 3 OUT endpoints configurable to support bulk, interrupt or isochronous transfers
- Management of a shared Rx FIFO and up to 4 dedicated Tx FIFOs (one for each active IN endpoint) for efficient usage of the USB data RAM and less load on the application

- Support the soft disconnect feature
- Can be bus-powered or self-powered

GD32207I USB OTG FS block diagram as shown in Figure：

Figure 5-55 USB OTG FS block diagram



The GD32207I access the OTG FS USB function module through the AHB bus (AHB frequency must be greater than 16MHz). USB 48MHz clock is get from the PLL by frequency division.

About other GD32207I USB OTG FS introduction, please refer to the “ARM Cortex-M3 32-bit MCU user manual”, the twenty-third chapter, here is no longer a detailed introduction.

In order to use the GD32207I USB normally, you have to write USB driver, and the entire USB communication process is very complicated, need to understand the entire USB protocol stack and the specific device class protocol. It can’t be described in detail here, for more details please refer to the USB 2.0 protocol standards. Of course, GD provides a complete USB OTG FS driver (including the host and the device), through this library can easily achieve the functions of the various USB Demo, without the need to understand the entire drive USB, greatly reducing the development time and effort. This drive library can be downloaded from the official website of GD.

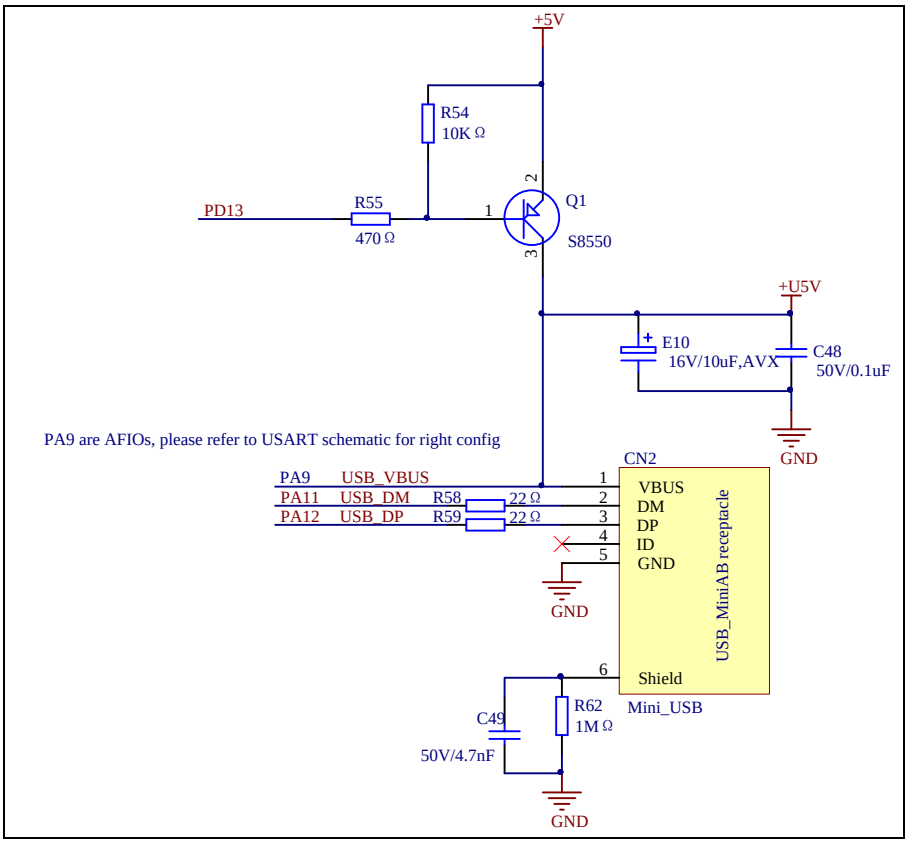
This Demo mainly implements a virtual serial port, is also a virtual COM port (UART). The COM port is not so common today, but much industrial software is still using this classic port. COM port belonging to the CDC, the Communication Device Class, is a type of dedicated USB sub class which is defined by USB organization to be used by a variety of communications equipment (telecommunications equipment and medium speed network communication equipment) using, mainly includes analog telephone and modem, digital telephone (including mobile phones) and virtual COM port. The content of the CDC protocol please refer to the standard of the protocol.

Usually a CDC class is made up of two interfaces subclasses: Communication Interface Class and Data Interface Class. Application is mainly through the communication interface to manage and control the device, and through the data interface to transfer data. These two

interfaces have different numbers and types of Endpoints. However, for the purposes of this Demo, the communication interface is not used, but in order to be compatible with windows it remains to be added to this interface.

The GD32207I USB OTG FS interface schematic diagram is shown in the following diagram, in the device mode it support from the USB interface to take electricity, in the host mode it control USB device power supply through the PD13.

Figure 5-56 Schematic diagram of USB



The USB OTG is only used as a device in this Demo, so it just takes electricity from the USB interface.

Jumper settings: JP5 whether jump to the end of the USB, according to the USB OTG global configuration register NOVBUSSENS bit to decide: if NOVBUSSENS bit is 1, then JP5 no need to jump to the end of the USB. At this time USB VBUS wire not connected to the PA9; if NOVBUSSENS bit is 0, then jp5 need to jump to the end of the USB. At this time the USB VBUS wire connected to the PA9. The role of the NOVBUSSENS bit, please refer to the user manual.

Due to the Demo using the LCD print the USB running log information, so also are LCD related jumper settings, and the cache of the LCD using SDRAM, so also SDRAM related jumper settings. Please refer to the related sections about these 2 modules in this manual.

5.22.3. DEMO Implementation Result

This Demo actually implements a USB-USART bridge, which can realize the communication between PC (not through the RS232 port). PC application in communication is the Windows super terminal. The actual USART is Usart1, so it must be configured with the USB OTG global configuration register NOVBUSSENS bit is 1 so that the VBUS USB line is not connected to the PA9. So you can jump JP5 to the Usart1 end, the PA9 to the Usart1 Tx port.

After the Demo compiled and download, you need to install the appropriate driver (GDUSB2Ser.inf), the driver can be downloaded from the official website of GD. The installation method can refer to the installation documentation in the VCP Driver GD folder after the decompression. After the installation is successful, you can find a new serial port in the computer equipment manager.

Next, the actual Usart1 is required to connect the development board to PC, and then open the PC super terminal, super terminal port selection will appear in two ports: COM14 (virtual serial port) and COM8 (real development board serial port). So, this time need to open 2 super terminals for communication, as follows:

Special note: how to modify the VID and PID

If users want to develop their own driver, they can modify the VID and PID in the Demo, and they need to modify the driver VID and PID in GDUSB2Ser.inf at the same time to maintain

the definition of PID and VID.

When modifying, the following two sections are found in the GDUSB2Ser.inf file, and then the 018A and 28E9 are required to be replaced by your own VID and PID.

[Standard.NTx86]

%GD32VCP.DeviceDesc%=GD32VCP_Device, USB\VID_28E9&PID_018A

[Standard.NTamd64]

%GD32VCP.DeviceDesc%=GD32VCP_Device, USB\VID_28E9&PID_018A

5.23. USB OTG_FS MSC host

5.23.1. DEMO Purpose

GD32207I-EVAL-V1.0 board support USB Host and USB Device with its own USB OTG FS (FS, Full Speed, 12Mbps).This Demo has been realized a USB MSC Host (the host) which can access USB MSC Device (the slave).

5.23.2. DEMO Principle

USB, Universal Serial BUS, Is an external bus standard used to regulate the connection and communication between the computer and the external device.USB interface supports device plug and play and hot plugging functions.

USB's development has experienced a number of versions of USB1.0/1.1/2.0/3.0/3.1。At present, the most used is USB 2.0, and the USB 3.0 is also becoming more and more popular. GD32207I's conforms to USB 2.0 specification.

Standard USB is composed of four lines, in addition to VCC/GND, the other are D+ and D-, the two data lines, which use the differential voltage mode for data transmission. On the USB host, D+ and D- are all connected the 15K resistance to ground, so the D- and D+ all low when there is no device to connect. And on the USB peripheral, if it is full-speed device, it will connect a 1.5K resistor to VCC on the D+, and if low-speed device, the connection will be on the D-. When the device is connected to the host, the host can judge whether there is a device to access, and to determine the device is full-speed or low-speed.

USB OTG is USB On-The-Go, means USB is in the process. USB OTG makes USB get rid of the limitation of the original master-slave architecture, and realizes the transmission mode of the end to end. USB OTG standard is fully compatible with the USB2.0 standard, and add limited host ability and the role exchanging function, which allows the device to operate at times as a host and at times as a peripheral (OTG dual role function). OTG dual role function device is fully in line with the USB2.0 standard and can provide limited host ability. It supports the host negotiation protocol (HNP) and the session request protocol (SRP). In OTG, the

initial host called the A-device, the initial peripheral known as the B-device. It mainly use cable connection mode to determine the initial role，the mini-AB receptacle which use is added the ID pin to identify different cable ends. Mini-A plug ID pin is connected to ground and mini-B plug ID pin is in floating. When the OTG device detected the ID pin is connected to ground, the default device is an A-device (the host) and the detection of the floating ID pin device is considered to be a B-Device (the peripheral). Once the system is connected, the OTG host and peripherals can realize the role exchange by using HNP protocol. A-device as the default host will provide VBus power, reset bus, enumerate and configure B-device in the detection of the device is connected. Session request protocol (SRP) allows the B device to open the VBus power and start a session on the A device. An OTG session can be determined by the time of the A- device to provide VBus power (Note: the A device is always powered for VBus, even is set as a peripheral). It can also be used to shut down the VBus power supply to save power, which is very important in battery powered products. Next, turn to the GD32207I USB OTG controller.

The USB OTG FS of the GD32207I is a dual-role device (DRD) controller that supports both device and host functions and is fully compliant with the On-The-Go Supplement to the USB 2.0 Specification. It can also be configured as a host-only or device-only controller, fully compliant with the USB 2.0 Specification. In host mode, the OTG_FS supports full-speed (FS, 12M bits/s) and low-speed (LS, 1.5M bits/s) transfers. Whereas in device mode, it only supports full-speed (FS, 12M bits/s) transfers. The OTG_FS supports both HNP and SRP.

The main features of GD32207I OTG FS include three categories: general, host-mode and device-mode features.

General features

- Fully compliant with the On-The-Go Supplement to the USB 2.0 Specification
- In PHY, it includes fully support for the optional protocol detailed in the On-The-Go Supplement Rev 1.3 specification
  - Supports the insertion A-B type device identification (USB ID line)
  - Supports Host Negotiation Protocol (HNP) and Session Request Protocol (SRP)
  - Allows host to turn V_{BUS} off to conserve battery power in OTG applications
  - Supports V_{BUS} level detection with internal comparators
  - Supports dynamic switching between host and device roles
- It can be configured by software to operate as:
  - USB OTG_FS dual role device (host or device)
  - USB FS/LS host (A-device)
  - USB FS Peripheral (B-device)
- It supports SOF (at FS) and keep-alive (at LS) by

  - SOF pulse PAD connectivity
  - SOF pulse internal connection to TIMER2
  - Configurable framing period
  - Configurable end of frame (EOF) interrupt
- It provides a dedicated RAM of 1.25 KB with advanced FIFO control for flexible and efficient use of RAM:
  - Configurable partitioning of RAM space into different FIFOs
  - Each FIFO can hold multiple packets
  - Dynamic and contiguous memory allocation
- It guarantees max USB bandwidth for up to 1 frame via hardware and needs no system intervention
- It provides power saving features during USB suspend:
  - Stop system
  - Switch off clock domains internal to the digital core, PHY and FIFO power management
- It supports suspend and resume

Host-mode features

- Needs an external charge pump or 5V power for V_{BUS} voltage generation
- Provides one port which is able to deliver a minimum of 100mA for a configured or un-configured device, and optionally, up to 500mA for a configured device
- Up to 8 host channels: each channel is dynamically configured in control, interrupt, bulk or isochronous transfer type
- Built-in hardware scheduler, it holds two hardware queues:
  - Up to 8 periodic transfer (interrupt or isochronous) requests in the periodic hardware queue
  - Up to 8 non-periodic transfer (control or bulk) requests in the non-periodic hardware queue
- In order to use the USB data RAM efficiently, it is allocated as a shared Rx FIFO, a periodic Tx FIFO and a non-periodic Tx FIFO to manage

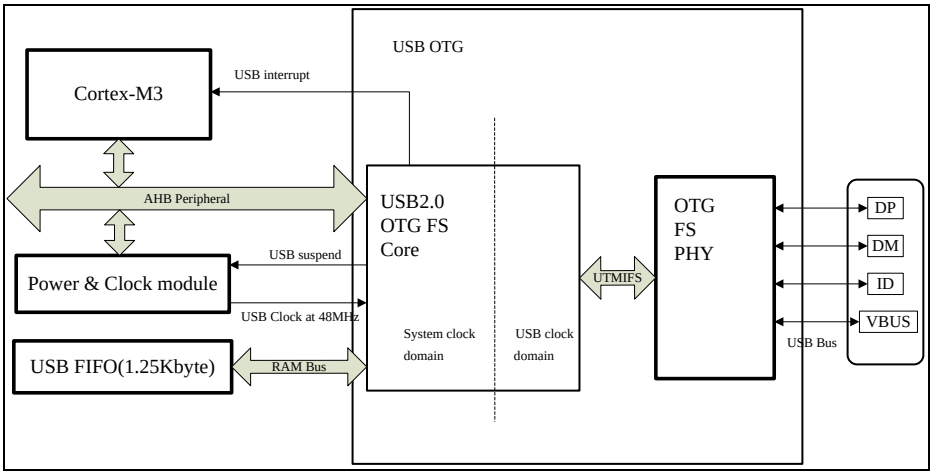
Device-mode features

- Draws 100mA or less from the bus before configuration
- Can draw up to 500mA from the bus after successful negotiation with the host

- 1 bidirectional control endpoint (endpoint 0)
- 3 IN endpoints and 3 OUT endpoints configurable to support bulk, interrupt or isochronous transfers
- Management of a shared Rx FIFO and up to 4 dedicated Tx FIFOs (one for each active IN endpoint) for efficient usage of the USB data RAM and less load on the application
- Support the soft disconnect feature
- Can be bus-powered or self-powered

GD32207I USB OTG FS block diagram as shown in Figure：

Figure 5-57 USB OTG FS block diagram



The GD32207I access the OTG FS USB function module through the AHB bus (AHB frequency must be greater than 16MHz). USB 48MHz clock is get from the PLL by frequency division.

About other GD32207I USB OTG FS introduction, please refer to the “ARM Cortex-M3 32-bit MCU user manual”, the twenty-third chapter, here is no longer a detailed introduction.

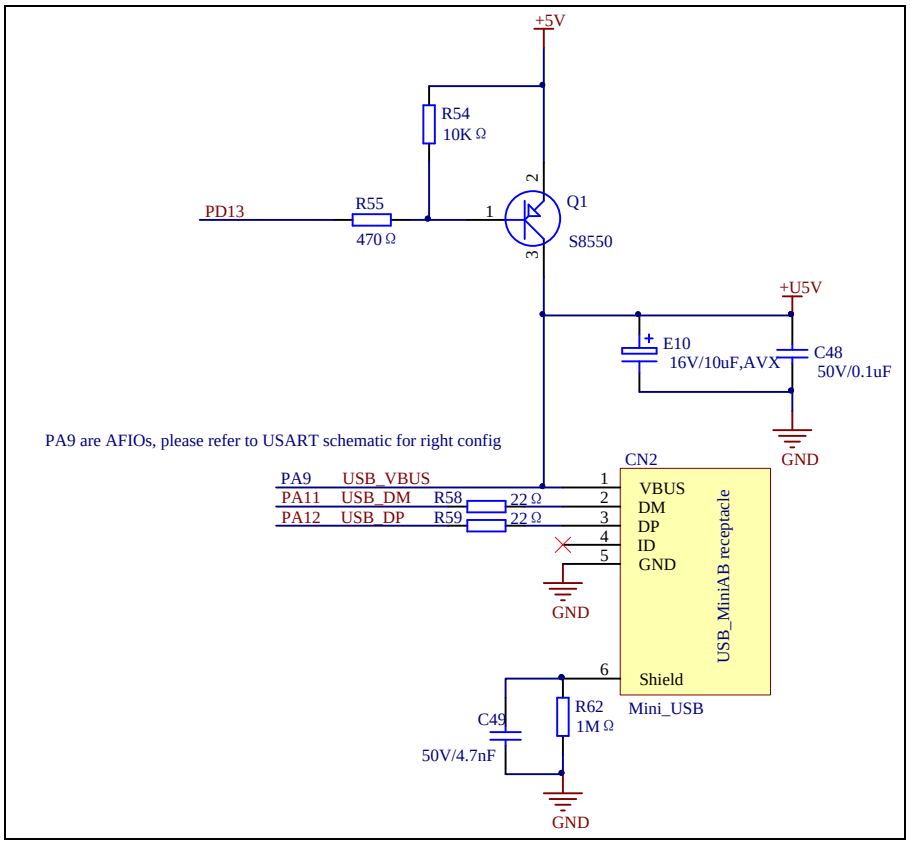
In order to use the GD32207I USB normally, you have to write USB driver, and the entire USB communication process is very complicated, need to understand the entire USB protocol stack and the specific device class protocol. It can’t be described in detail here, for more details please refer to the USB 2.0 protocol standards. Of course, GD provides a complete USB OTG FS driver (including the host and the device), through this library can easily achieve the functions of the various USB Demo, without the need to understand the entire drive USB, greatly reducing the development time and effort. This drive library can be downloaded from the official website of GD.

This Demo mainly implements a USB Host, the ability of which is relatively limited. The USB host can only detect USB devices, enumerate the USB MSC devices, and it can only communicate with the USB MSC class devices. So, this Demo has been realized a dedicated USB Host.

The USB OTG Host is usually not required to install the device driver, because it acts as a driving role itself. In order to communicate with USB MSC Device, it requires the support of the MSC protocol, and the ability to send a MSC class device request. These have been implemented in Demo. In order to access the contents of USB MSC Device, FATFS (Allocation Table File System File) was added in the experiment. For more details please refer to the relevant file.

The GD32207I USB OTG FS interface schematic diagram is shown in the following diagram, in the device mode it support from the USB interface to take electricity, in the host mode it control USB device power supply through the PD13.

Figure 5-58 Schematic diagram of USB



The USB OTG is only used as a host in this Demo, so it need control the VBUS power supply to USB device through the PD13.

Jumper settings: JP5 don't need jump to the end of the USB, because in host mode, USB OTG VBUS need not connect to PA9 all the time.

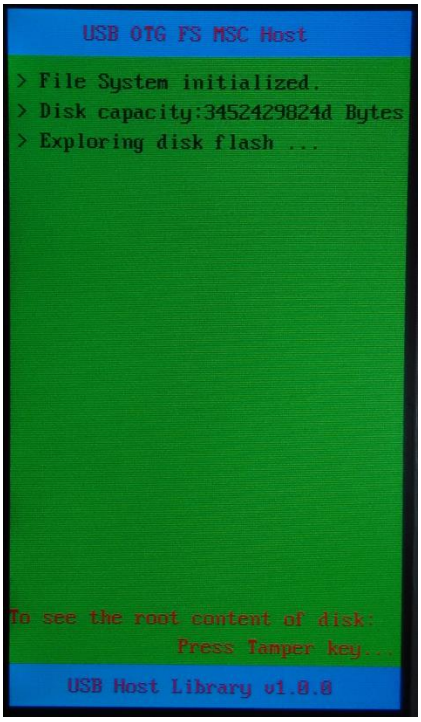
Due to the Demo using the LCD print the USB running log information, so also are LCD related jumper settings, and the cache of the LCD using SDRAM, so also SDRAM related jumper settings. Please refer to the related sections about these 2 modules in this manual.

5.23.3. DEMO Implementation Result

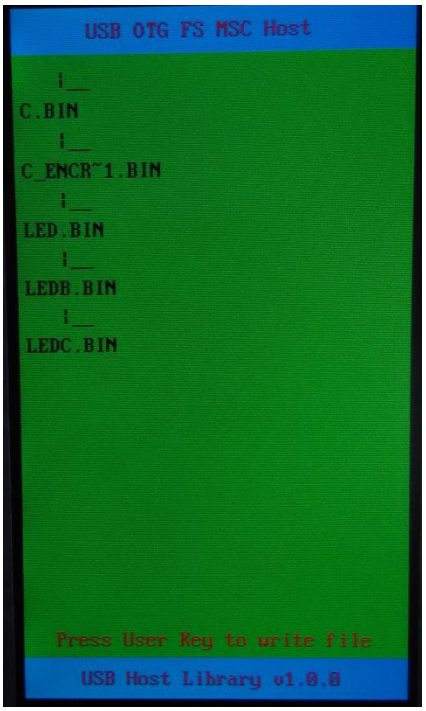
After Demo compiled and download, use the DC power supply to the development board. Then connect a U disk to the development board through the USB adapter. Next, you can operate according to the print information on the LCD, the results are as follows:

1. After peripheral is connected, LCD will display the device information from the host enumerating:

2. After pressing User key, the application will initialize file system and LCD will display MSC device capacity:



3. After pressing Tamper key, LCD will display the file content in the MSC device, showing the level of the folder is 2:



4. After pressing User key, application will write GD32.txt file to the MSC device:

Last operation is to show picture in the MSC device, but this operation has not been realized.

5.24. ETH

5.24.1. DEMO Purpose

GD32207I-EVAL-V1.0 development board integrated ETH（Random number generator）, The Ethernet peripheral of GD32F20x contains the 10/100Mbps Ethernet MAC(media access controller), designed to provide optimized performance through the use of DMA hardware acceleration, support two standard communication interface with the physical layer (PHY): MII(media independent interface) and RMII(reduced media independent interface) to transmit and receive data. This Demo is used to demonstrate the function and usage of the ETH module in the GD32207I-EVAL-V1.0 development board.

GD32F20X ETH main features:

MAC

- Support 10/100 Mbit/s data transfer rates.
- Support CSMA/CD Protocol for half-duplex Back-pressure operation.
- Support IEEE 802.3x flow control for full-duplex operation, Automatic transmission of pause frame on deassertion of flow control input.
- Option for automatic pad/CRC generation in transmit operation.
- Option for automatic pad/CRC stripping in receive operation.
- Option for frame length to support Standard frames with sizes up to 16 KB.
- Option for interframe gap (40-96 bit times in steps of 8).

- Support different receiving filter mode.
- Support IEEE 802.1Q VLAN tag detection for reception frames.
- Support mandatory network statistics with RMON/MIB counters (RFC2819/RFC2665).
- Support Detection of LAN wakeup frames and AMD Magic Packet frames.
- Support Receive feature for checksum off-load for received IPv4 and TCP packets encapsulated by the Ethernet frame.
- Support Enhanced receive feature for checking IPv4 header checksum and TCP, UDP, or ICMP checksum encapsulated in IPv4 or IPv6 datagrams.
- Support Ethernet frame time stamping as described in IEEE 1588-2002. 64 bit time stamps are given in each frame's transmit or receive status.
- Two independent FIFO of byte 2K for transmitting and receiving.
- Support statistics by generating pulses for frames dropped or corrupted (due to overflow) in the Receive FIFO.
- Automatic generation of PAUSE frame control or back pressure signal to the MAC core based on Receive FIFO-fill (threshold configurable) level.
- Discard frames on late collision, excessive collisions, excessive deferral and underrun conditions.
- Calculate and insert IPv4 header checksum and TCP, UDP, or ICMP checksum in frames transmitted in Store-and-Forward mode.

DMA

- Support ring or chain descriptor chaining.
- Each descriptor can transfer up to 8 KB of data.
- Round-robin or fixed-priority arbitration between reception and transmission controller priority.

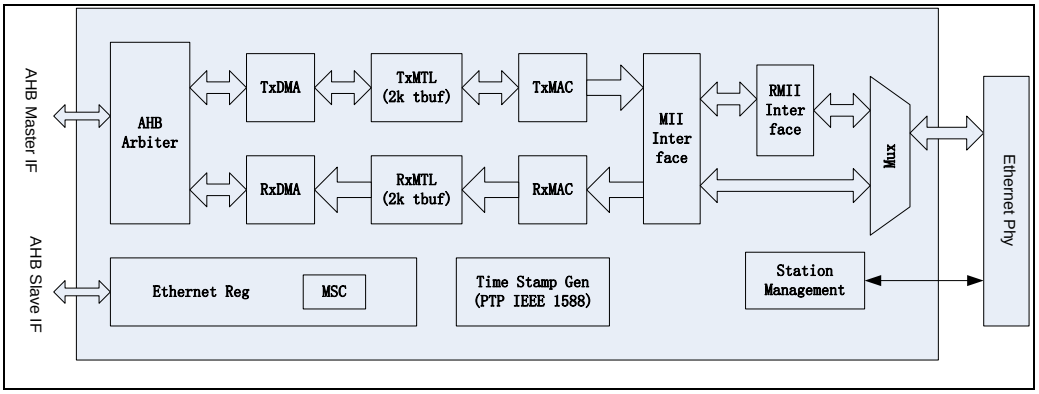
PTP

- Support IEEE1588 time synchronization function.
- Support two correction methods: Coarse or fine.
- Pulse per second output.

5.24.2. DEMO Principle

The Ethernet module is composed of a MAC (media access controller) module, MII/RMII module and a DMA module by descriptor control.

Figure 5-59 ETH module block diagram



The MAC module is connected to the external PHY by MII or RMII through one selection bit (refer to AFIO_PCFR1 register). The SMI interface (MDIO and MDC), is used to configure and manage external PHY.

Transmitting data module includes:

- Tx DMA controller, used to read descriptors and data from memory and writes status to memory.
- Tx FIFO, used to cache for MAC transmission data.
- The MAC transmission control register group, used to control frame transmit.

Receiving data module includes:

- Rx DMA controller, used to read descriptors from memory and writes data and status to memory.
- MAC receive control register group, used to control frame receive and marked the receiving state.
- The receiving filter, can use a variety of filtering mode, filter out specific Ethernet frame
- Rx FIFO, delay a received frame to achieve, thus filter can filter out specific frames, and then receives the frame into the memory.

GD32F2 ETH can communication that must have external PHY chip, , PHY chip is connected to the internal MAC through the MII / RMII interface, and supports SMI (MDIO & MDC) interface that allows MAC configuration of the external Ethernet PHY chip.

Next, the PHY chip interface SMI and MAC communication interface MII / RMII are introduced. And describes the PHY chip development board used.

MII/RMII selection

The application has to set the MII/RMII mode through configuration of the AFIO_PCFR1 register 23 bits MII_RMII_SEL while the Ethernet controller is under reset or before enabling the clocks. The MII mode is set by default.

Station management interface: SMI

Station management interface (SMI) through two wire: clock line(MDC) and data line(MDIO) for communication with the external PHY, it can access to the any PHY register. The interface supports accessing up to 32 PHYs, but only one register in one PHY can be addressed at the same time.

Two wires: MDC and MDIO Specific functions as follows:

- MDC: a clock of maximum frequency is 2.5 MHz. The pin remains low level in the idle state. The minimum high and low times for MDC must be 160 ns each, and the minimum period for MDC must be 400 ns in data transmission.

- MDIO: Used to transfer data in conjunction with the MDC clock line, receiving / sending data.

SMI write operation

Applications need to write transmission data to the ETH_MAC_PHYDR register and operate the ETH_MAC_PHYAR register as follows: Set the PHY device address and register address will operate, PW is set to 1, so that can enable write mode. After that set PB bit start transmission. In the process of transaction PB is always high until the transfer is complete SMI interface will clear it. The application can determine whether a transaction complete through PB bit. When PB is 1, the application should not change the PHY Address register contents or the PHY Data register. Write operations to the PHY Address register or the PHY Data Register during this period are ignored (the PB bit is high), and the transaction is completed without any error.

SMI read operation

Applications need to operate the ETH_MAC_PHYAR register as follows: Set the PHY device address and register address will operate, PW is set to 0, so that can enable read mode. After that set PB bit start reception. In the process of transaction PB is always high until the transfer is complete SMI interface will clear it. The application can determine whether a transaction complete through PB bit. When PB is 1, the application should not change the PHY Address register contents or the PHY Data register. Write operations to the PHY Address register or the PHY Data Register during this period (the PB bit is high) are ignored, and the transaction is completed without any error.

Note: Because the PHY register address 16-31 register functions define by each manufacturer, access different PHY devices's this part registers should accord to the manufacturer manual to adjust the parameters of software. Details of Catalog that GD32F20x firmware library currently supports the PHY device can refer to firmware library related instructions.

SMI clock selection

The SMI clock is a divided clock whose source is the application clock (AHB clock). In order to guarantee the clock frequency is less than 2.5MHZ, according to the AHB clock frequency set the PHY address register related bit, select the appropriate frequency division factor. The following table lists the frequency factor corresponding AHB clock selection.

Table 5-8 Clock range

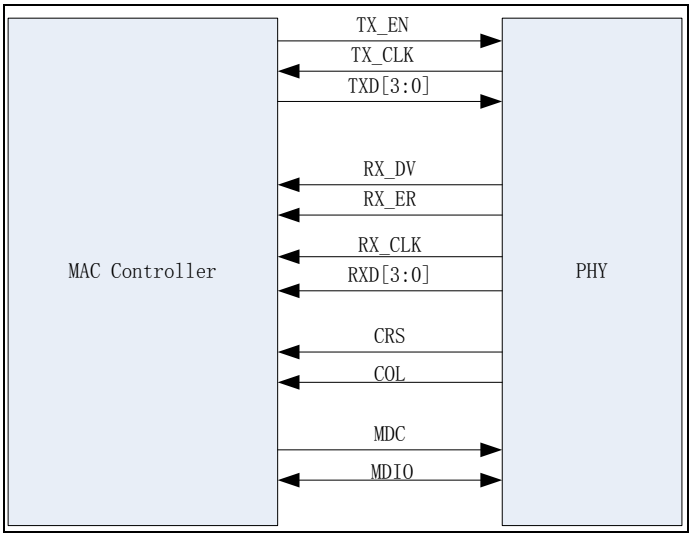
AHB clock	MDC clock	Selection
Reserved	—	0100，0101，0110，0111

20~35MHz	AHB clock/16	0011
35~60MHz	AHB clock/26	0010
90~108 MHz	AHB clock/64	0001
60~90MHz	AHB clock/42	0000

Media-independent interface: MII

The media-independent interface (MII) defines the interconnection between the MAC sublayer and the PHY for data transfer at 10 Mbit/s and 100 Mbit/s.

Figure 5-60 Media independent interface signals



- MII_TX_CLK: clock signal for transmitting data. For the data transmission of 10M /s, the clock is 2.5MHz, for the data transmission of 100M /s, the clock is 25MHz.

- MII_RX_CLK: clock signal for receiving data. For the data transmission of 10M /s, the clock is 2.5MHz, for the data transmission of 100M /s, the clock is 25MHz.

- MII_TX_EN: Transmission enable signal. It must be asserted synchronously with the first bit of the preamble and must remain asserted while all bits to be transmitted are presented to the MII.

- MII_TXD [3:0]: Transmit data line, each 4 bit data transfer, data are valid in the MII_TX_EN signal is effective. MII_TXD [0] is the least significant bit, MII_TXD[3] is the most significant bit. While MII_TX_EN is deasserted the transmit data must have no effect upon the PHY.

- MII_CRS: Carrier sense signal, only working in half duplex mode. Controlled by the PHY, enable it when either the transmit or receive medium is non idle. The PHY must ensure that the MII_CRS signal remains asserted throughout the duration of a collision condition. This signal is not required to transition synchronously with respect to the TX and RX clocks.

- MII_COL: collision detection signal, only working in half duplex mode. Controlled by the PHY, enable it when detection of a collision on the medium and must remain asserted while the collision condition persists. This signal is not required to transition synchronously with respect to the TX and RX clocks.

- MII_RXD[3:0]: Receive data line, each 4 bit data transfer, data are valid in the MII_RX_DV signal is effective. MII_RXD[0] is the least significant bit, MII_RXD[3] is the most significant bit. While MII_RX_EN is deasserted and MII_RX_ER is asserted, a specific MII_RXD[3:0] value is used to indicate specific information (see Table 10-3).

- MII_RX_DV: Receive data enable signal. Controlled by the PHY, enable it when PHY is presenting on the MII for reception. It must be asserted synchronously with the first bit of the frame and must remain asserted while all bits to be transmitted are presented to the MII. It must be deasserted prior to the first clock cycle that follows the final bit. In order to receive the frame correctly, the effective signal starting no later than the SFD field.

- MII_RX_ER: Receive error signal.It must be asserted for one or more clock periods to indicate MAC detected an error in the receiving process. The specific error reason need to cooperate with the state of the MII_RX_DV and the MII_RXD[3:0] data value.

MII clock sources

To generate both TX_CLK and RX_CLK clock signals. The external PHY must be clocked with an external 25 MHz. The clock does not require the same with MAC clock. Can use the external 25MHz crystal or GD32F20x microcontroller MCO pin provides the clock. When the clock source from MCO pins need to configure the appropriate PLL, ensure the MCO pin output clock for 25MHZ.

Reduced media-independent interface: RMII

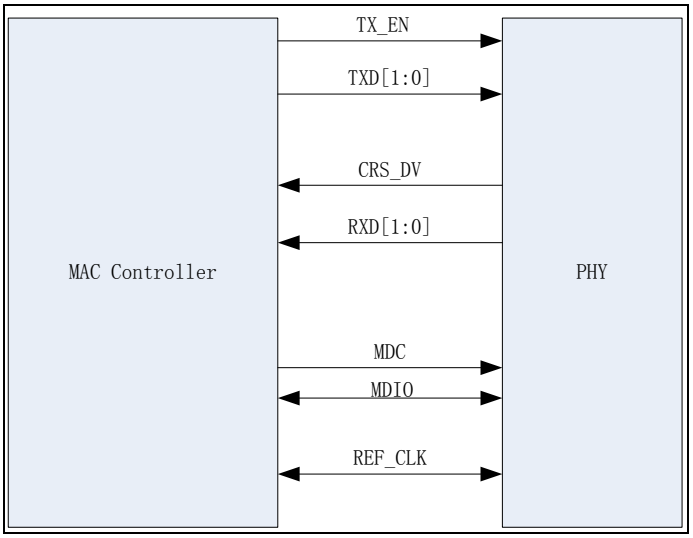
The reduced media-independent interface (RMII) specification reduces the pin count when ethernet communication. According to the IEEE 802.3 standard, an MII contains 16 pins for data and control. The RMII specification is dedicated to reduce the pin count to 7 pins
The RMII block has the following characteristics:

- The clock signal needs to be increased to 50MHz.

- MAC and external PHY need to use the same clock source

- Using the 2-bit wide data transceiver

Figure 5-61 Reduced media-independent interface signals



MII/RMII bit transmission order

Each bit from the MII is transmitted on the RMII a dibit at a time with the order of dibit transmission. as follows: The first transmit / receive low 2 bits, then transmit / receive high 2 bits.

RMII clock sources

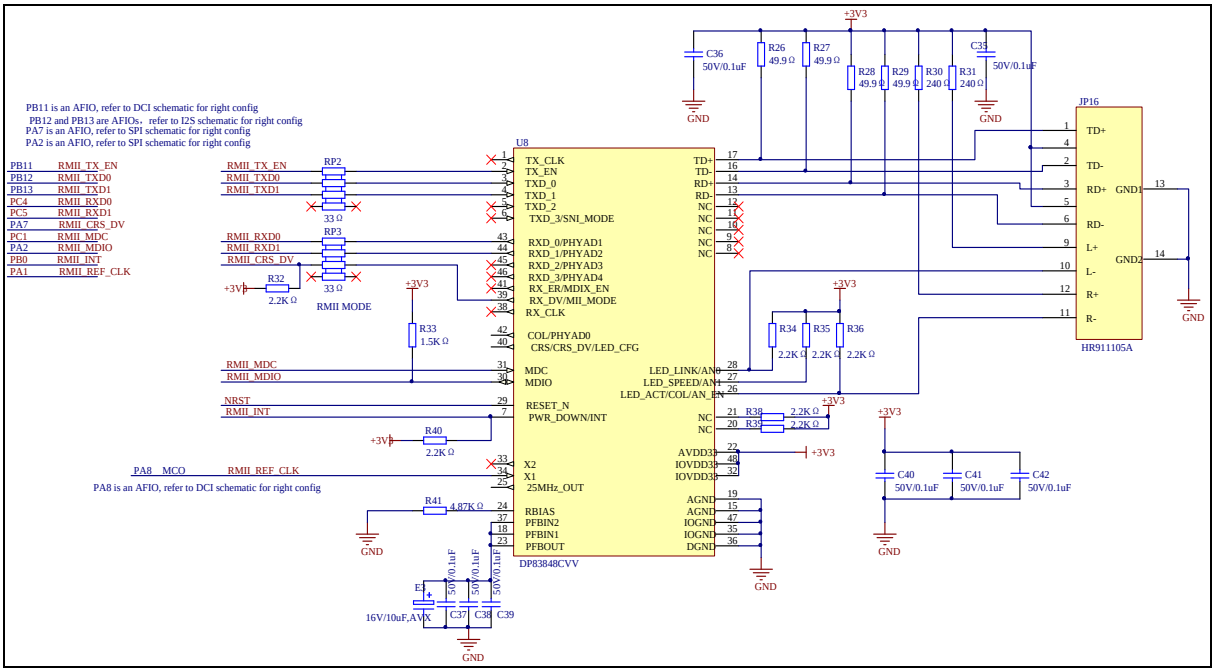
To ensure the synchronization of the clock source by the same clock source to the MAC and Ethernet PHY REF_CLK pins. Can use the external 50MHz crystal or GD32F20x microcontroller MCO pin provides the clock. When the clock source from MCO pins need to configure the appropriate PLL, ensure the MCO pin output clock for 50MHZ.

The Demo selected PHY chip DP83848 and used RMII interface to connect external PHY. Detailed information on the DP83848 chip, please refer to the relevant datasheet

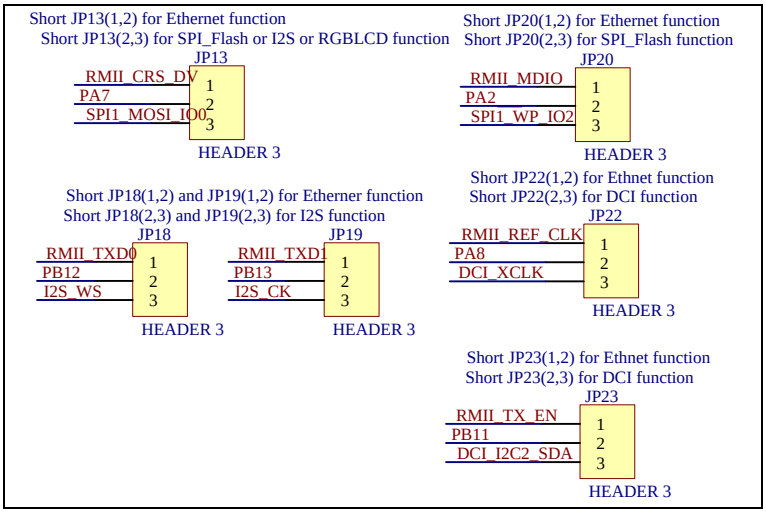
The Demo used the LWIP as TCP / IP protocol stack. LWIP is a small open source TCP / IP protocol stack, with or without operating system support can run and reduce the occupation of RAM on the basis of maintaining the main functions of the TCP protocol, it can run just used a dozen KB RAM and 40K or so ROM. For more information about LWIP can refer to the website：http://savannah.nongnu.org/projects/lwip/。

Demo of ETH in GD32207I-EVAL-V1.0 development board needs to use external PHY chip and RJ45 connector with integrated magnetics，The hardware circuit is shown in the following diagram.

Figure 5-62 Schematic diagram of Ethernet

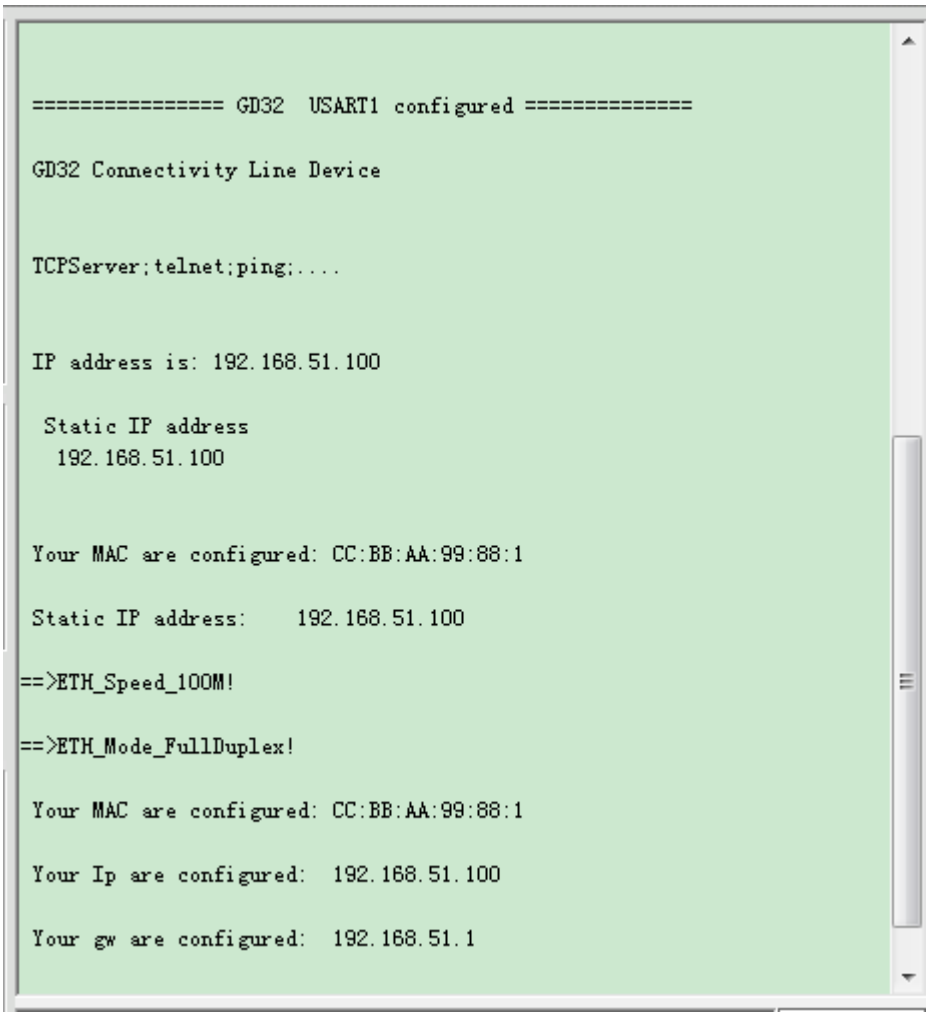


Jumper: Need to JP13/JP18/JP19/JP20/JP22/JP23 configured to the correct location.

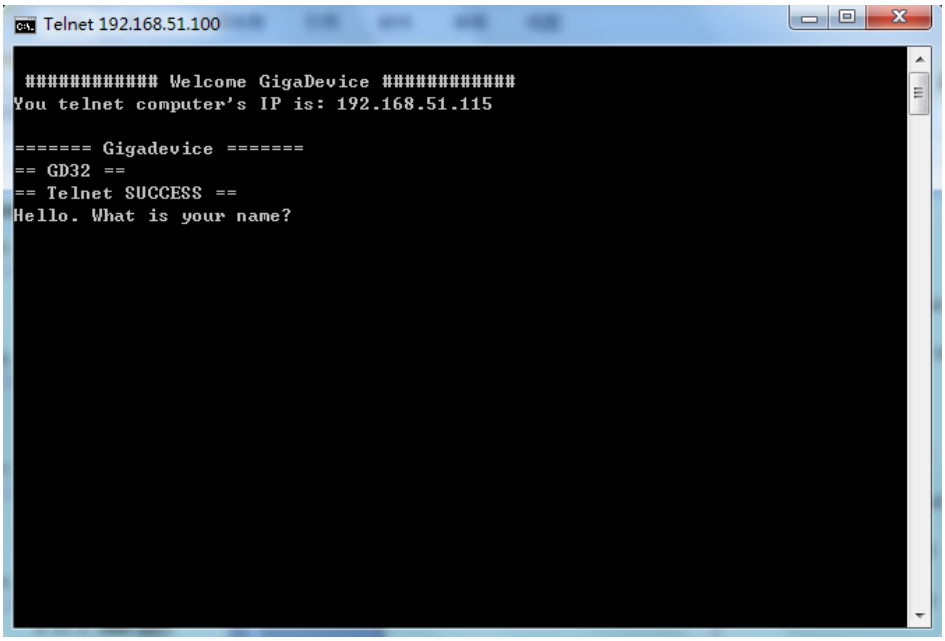


5.24.3. DEMO Implementation Result

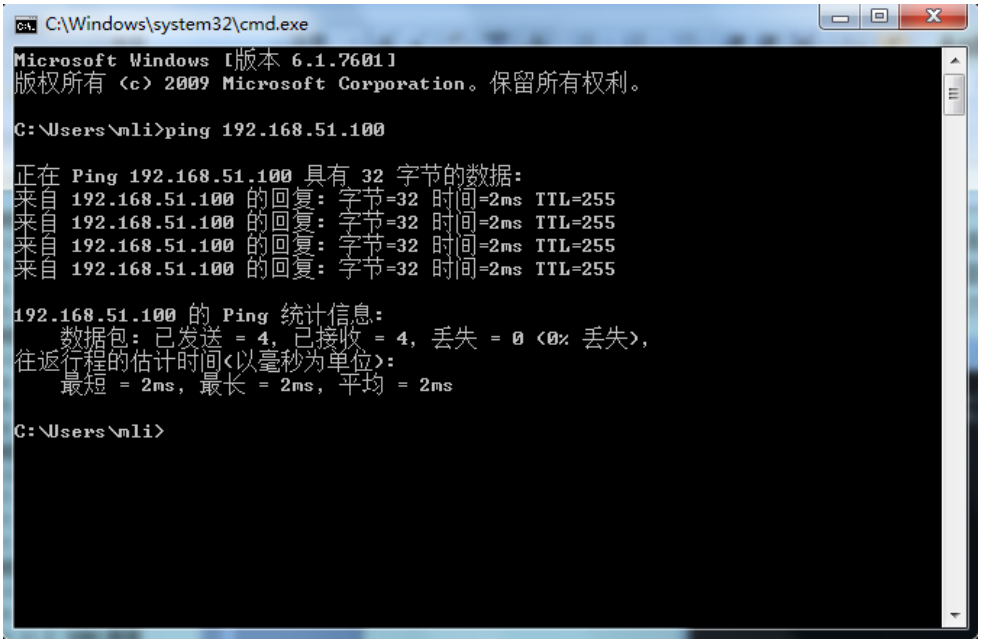
Connecting PC to the development board with a network cable，Download the program to the development board, serial port output information, as shown in the following figure.



User must ensure that the IP address in board and the PC on the same network segment. The user can experiment with CMD command-line tool. According to the chart shows, the development board IP address 192.168.51.100, enter telnet 192.168.51.100 in the CMD command line returns the following results：



Enter “PING 192.168.51.100” can be implemented on the development board PING operation



User can also use network assistant software build TCP connection with the development board by port 8000, when TCP connection is established the board will return information shown below

5.25. EXMC_SDRAM

5.25.1. DEMO Purpose

There is a 256Mb SDRAM (MT48LC16M16A2P-6AIT) on the GD32207I-EVAL-V1.0 board. The SDRAM can be accessed via EXMC module of GD32F207IKT6. This Demo is used to show how to use EXMC SDRAM controller to access SDRAM.

GD32F20X EXMC SDRAM main features:

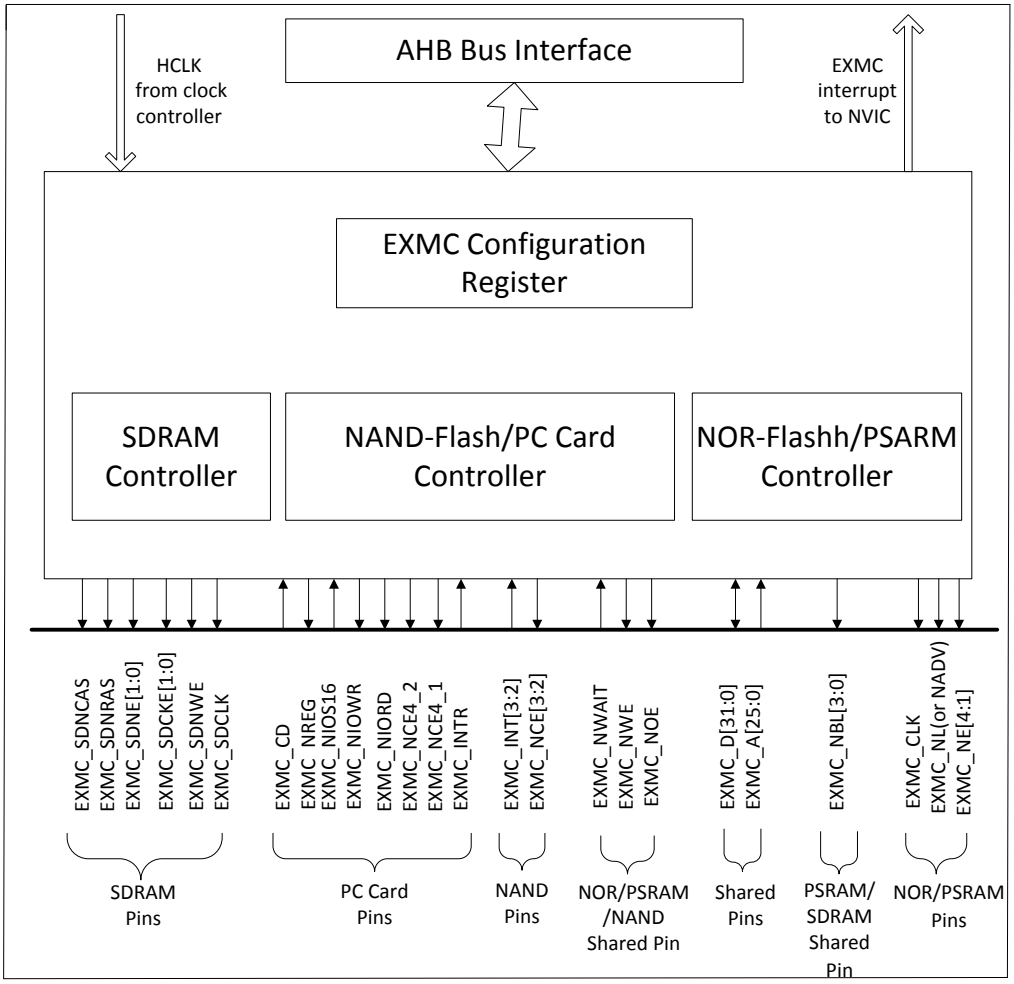
- Two SDRAM banks with independent configuration
- 8-,16- or 32-bit wide data bus
- Up to 13-bits Row Address, 11-bits Column Address, 2-bits internal banks address
- Memory size: 4x16Mx32bit(256 MB), 4x16Mx16bit (128 MB), 4x16Mx8bit (64 MB)
- AHB Word, half-word, byte access
- Independent Chip Select control for each memory bank
- Independent configuration for each memory bank

- Write enable and byte lane select outputs
- Automatic row and bank boundary management
- Multibank ping-pong access
- SDRAM clock can be fHCLK/2 or fHCLK /3
- Programmable timing parameters
- Automatic Refresh operation with programmable Refresh rate
- SDRAM power-up initialization by software
- CAS latency of 1,2,3
- Write Data FIFO with 16 x35-bit depth
- Write Address FIFO with 16x31-bit depth
- Cacheable Read Data FIFO with 6 x32-bit depth
- Cacheable Read address FIFO with 6 x14-bit depth
- Ajustable read data sample clock
- Self-refresh mode
- Power-down mode

5.25.2. DEMO Principle

As shown in Figure 5-63, EXMC module includes six parts: AHB bus interface, EXMC configuration registers, NOR Flash memory controller, NAND Flash and PC Card controller, SDRAM controller, external device interface. Reference clock of EXMC module is the AHB clock (HCLK).

Figure 5-63 The EXMC block diagram



SDRAM can be accessed through the SDRAM controller of EXMC. Table 5-9 lists the SDRAM interface.

Table 5-9 SDRAM interface signal

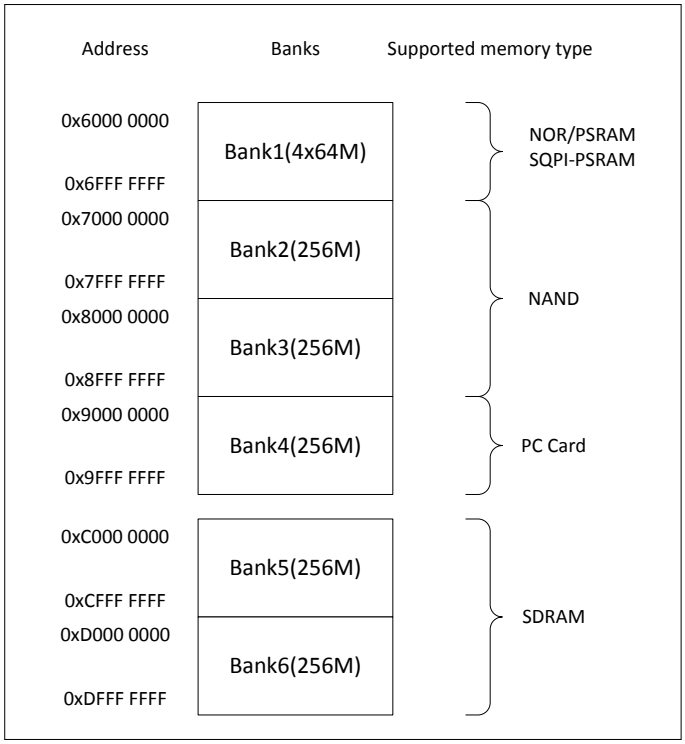
Signal	Direction	Description
SDCLK	O	SDRAM memory clock
SDCKE[0]	O	Clock enable for SDRAM memory 0
SDCKE[1]	O	Clock enable for SDRAM memory 1
SDNE[0]	O	Chip select for SDRAM memory 0, active low
SDNE [1]	O	Chip select for SDRAM memory 1, active low
SDNRAS	O	Row address strobe, active low
SDNCAS	O	Column address strobe, active low
SDNWE	O	Write enable, active low
A[12:0]	O	Address
BA[1:0]	O	Bank address
D[31:0]	I/O	Read/Write Data
NBL[3:0]	O	Write data mask

EXMC provides the conversion interface between AHB bus and external device protocol.

32-bit of AHB read or write accesses can be split into several consecutive 8-bit or 16-bit read or write operations.

EXMC external memory can be divided into many banks and they can support different types of memory. Each bank is 256 Mbytes. The address space of each bank and the supported memory type are shown in Figure 5-64.

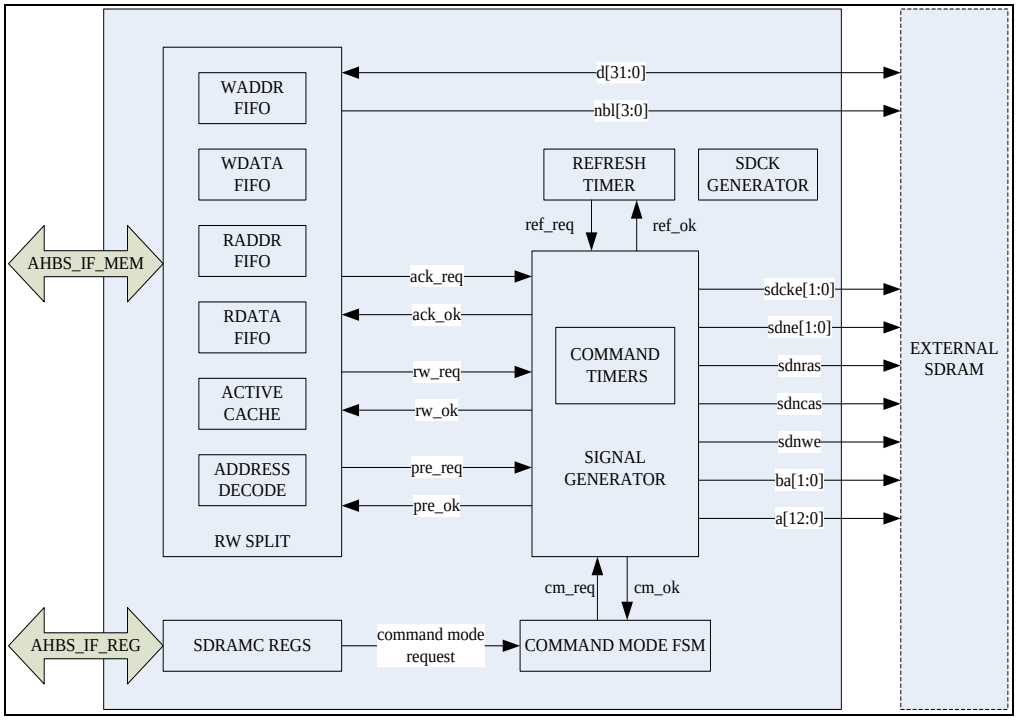
Figure 5-64 EXMC memory banks



You can access SDRAM through the Bank5 or Bank6, and EXMC module provides dedicated registers to generate the appropriate read and write timing according to user needs and the characteristics of external memory.

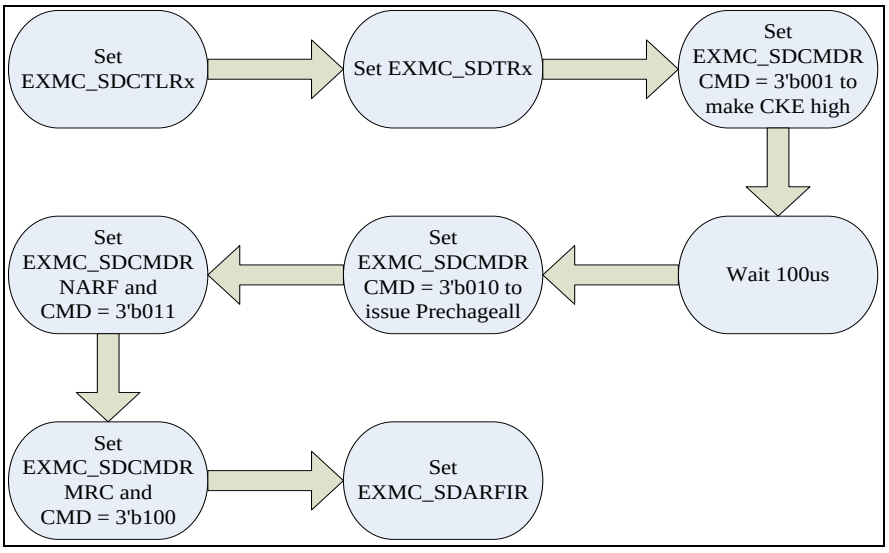
As shown in Figure 5-65, the SDRAM controller of EXMC module mainly includes the following sub modules: RW split module, refresh timer, command mode FSM, SDCLK generator, signal generator. The RW split module accepts AHB commands, and transfers them to single read/write access on the SDRAM memory according to the ratio of the data width between the AHB bus and the SDRAM memory interface. Inside the RW split module, there are two write FIFOs, which buffers the data and address of the AHB write commands. The refresh timer calculates the auto-refresh interval and generates auto-refresh request. The command mode FSM generates corresponding requests to the signal generator to make the command true and record the state of the external memory devices. The SDCLK generator generates clock for the SDRAM. The signal generator handles requests from command mode FSM, refresh timer and the RW split module.

Figure 5-65 SDRAMC block diagram



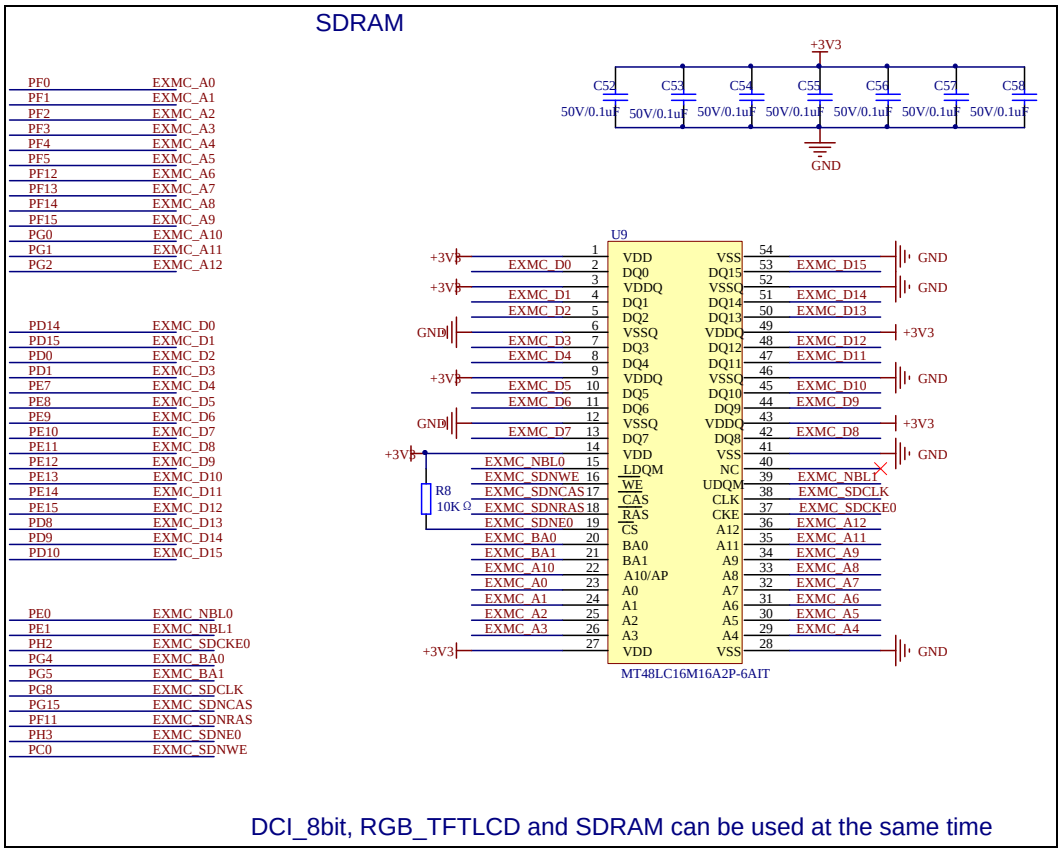
The initialization sequence of SDRAM is shown in Figure 5-66.

Figure 5-66 initialization sequence of SDRAM controller



The schematic diagram of 256Mb SDRAM (MT48LC16M16A2P-6AIT) which is on the GD32207I-EVAL-V1.0 board is shown in Figure 5-67.

Figure 5-67 Schematic diagram of SDRAM



Data bus width of SDRAM is 16bit. SDRAM is internally configured as a quad-bank DRAM with a synchronous interface. Each of the banks is organized as 8192 rows(13 bits width of a row address) by 512 columns(9 bits width of column address) by 16 bits. And it provides for programmable read or write burst lengths (BL) of 1, 2, 4, or 8 locations, or the full page.

Prior to normal operation, the SDRAM must be initialized. The initialization sequence is managed by software and executed by the Command Mode FSM.

According to user needs and the characteristics of the SDRAM, configuring the EXMC_SDCTLRx and EXMC_SDTRx registers. After the configuration is complete, sending a CKE high command to SDRAM to enable the clock. Then, the SDRAM requires a 100us delay. during this 100us period, COMMAND INHIBIT or NOP commands must be applied. After the 100us delay, software should send a precharge all command to make all banks into idle state. Once in the idle state, at least two AUTO REFRESH cycles must be performed, which will make the SDRAM into the ready state of the load mode register. Because the mode register will power up in an unknown state, it must be loaded prior to applying any operational command. Finally, programming the SDRAM refresh rate into the EXMC_SDARFIR register.

After the completion of the initialization, the SDRAM enters the ready state to receive any command.

The definition of the mode register can refer to the SDRAM user manual. It defines the

specific mode of operation, including burst length (BL), burst type, CAS latency (CL), operating mode, and write burst mode. The mode register is programmed via load mode register command and retains the stored information until it is programmed again or the device loses power.

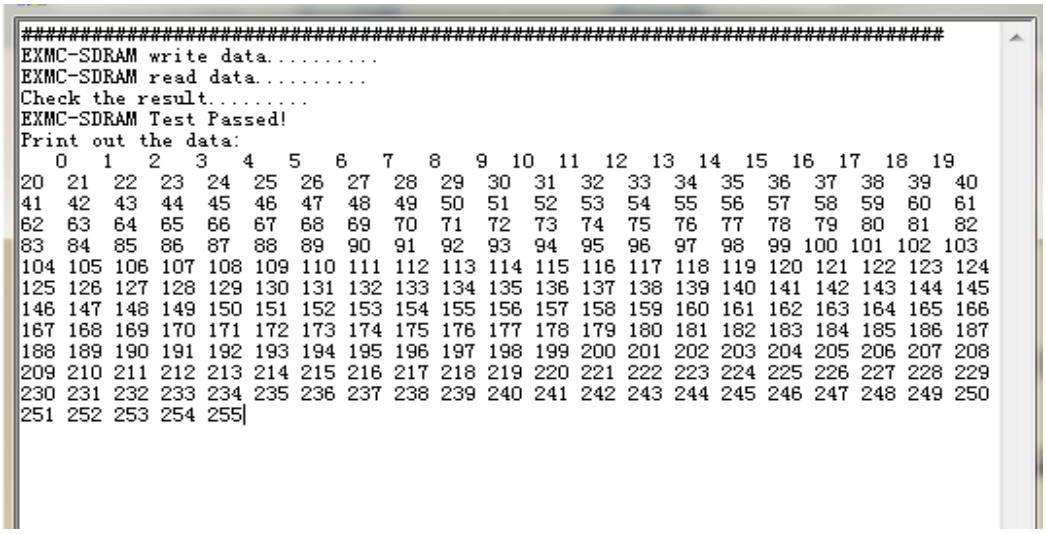
5.25.3. DEMO Implementation Result

The SDRAM on GD32207I-EVAL-V1.0 board is connected to the Bank5 of EXMC. This Demo mainly achieved the read and write operations of SDRAM through EXMC module.

First, the SDRAM is initialized by the software according to the initialization sequence. Then, write 256 bytes data starting from a designated address of SDRAM and read out the data for verification. If all the read data and write data are equal, LED1 and LED3 on the GD32207I-EVAL-V1.0 board will be turned on and the serial port will print out the successful access information and the 256 bytes data read from SDRAM. Otherwise, four LEDs will be turned on, and the serial port will print out the failure information.

Put jumper "JP5" to "USART1" and the serial line connect to COM1. The operating result can be view via the serial port.

After you download the program to the development board, if the program is running correctly, the following information will be shown through the serial port.



5.26. DCI Camera Capture

5.26.1. DEMO Purpose

This Demo uses OV7670 camera capture images and display on the TFT-LCD screen of GD32207I-EVAL-V1.0 board. Digital camera interface is a synchronous parallel interface, you can get the video and images from digital camera, support for 8, 10, 12 and 14 bit data flow and DMA operation.

GD32F20X DCI main features:

- Digital video/picture capture
- 8/10/12/14 data width supported
- High transfer efficiency with DMA interface
- Video/picture crop supported
- Various pixel formats supported including JPEG/YCrCb/RGB
- Hard/embedded synchronous signals supported

5.26.2. DEMO Principle

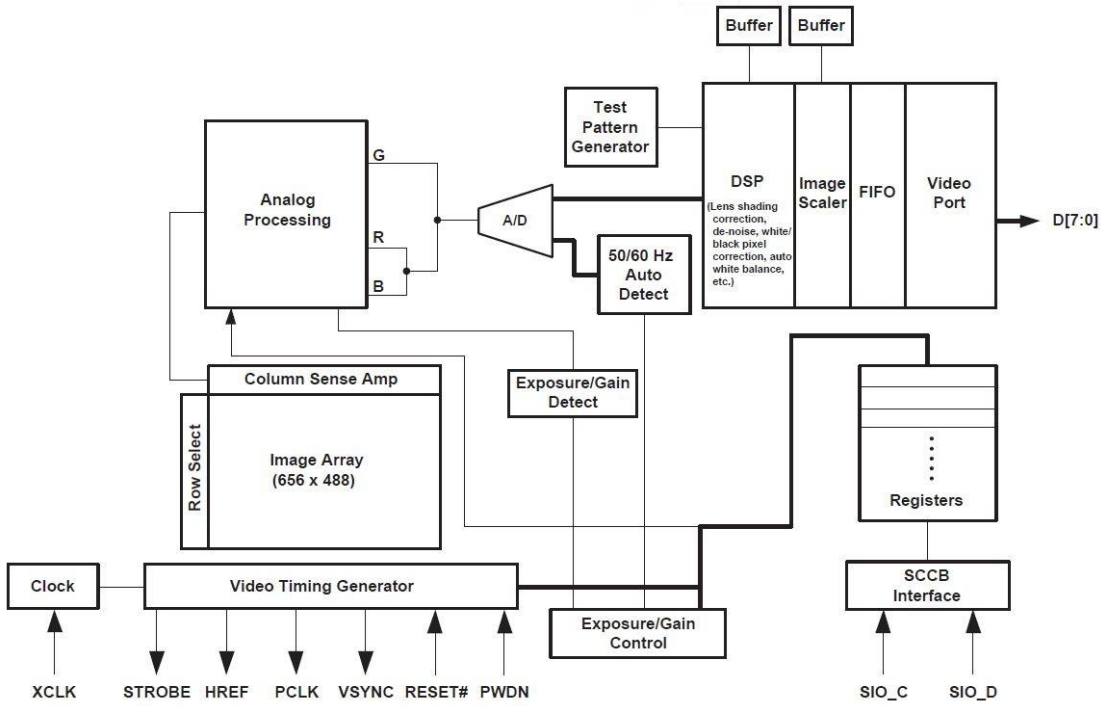
DCI principle:

Digital camera interface can receive the external 8, 10, 12, 14 bit CMOS camera capture images , support different data formats: JPEG, RGB and YCrCb. DCI receives the image data through DMA, the image data can choose VSYNC and HSYNC hardware synchronization, or embedded code synchronization.

DCI interface through the DCI_HSYNC, DCI_PIXCLK, DCI_VSYNC to obtain the image data timing input, through the DCI_D[0:13] to get the image data, through the DCI_XCLK to provide external camera clock input.

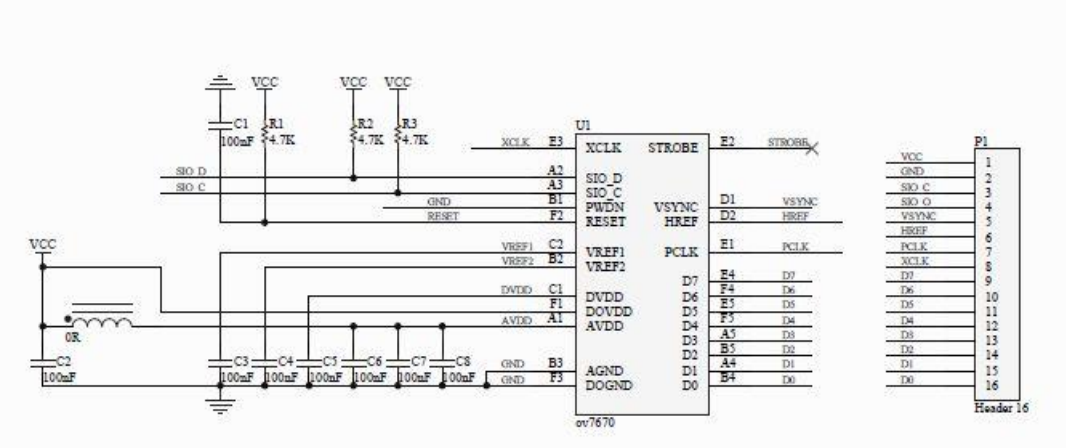
OV7670 principle:

OV company defines a serial camera control bus SCCB, compatible with the I2C protocol, the internal registers of the image sensor through the SCCB interface access. Via XCLK signal line input clock. The output image data of the camera is controlled by VSYNC, HREF and PCLK pins, and the image data is output through D[7:0]. Here is the functional framework of the OV7670 camera:



Hardware design:

In this Demo, the signal line is connected to the OV7670 module using the DCI interface. The following diagram is a schematic diagram of the OV7670 module:



Connected linesare shown as follows:

PB10——I2C2_SCL——OV7670_SIO_C

PB11——I2C2_SDA——OV7670 _SIO_D

PG9——DCI_VSYNC——OV7670 _VSYNC

PH8——DCI_HSYNC——OV7670 _HREF

PA6——DCI_PIXCLK——OV7670 _PCLK

PA8——DCI_XCLK——OV7670 _XCLK

PC6——DCI_D0——OV7670 _D0

PC7——DCI_D1——OV7670 _D1

PC8——DCI_D2——OV7670 _D2

PC9——DCI_D3——OV7670 _D3

PC11——DCI_D4——OV7670 _D4

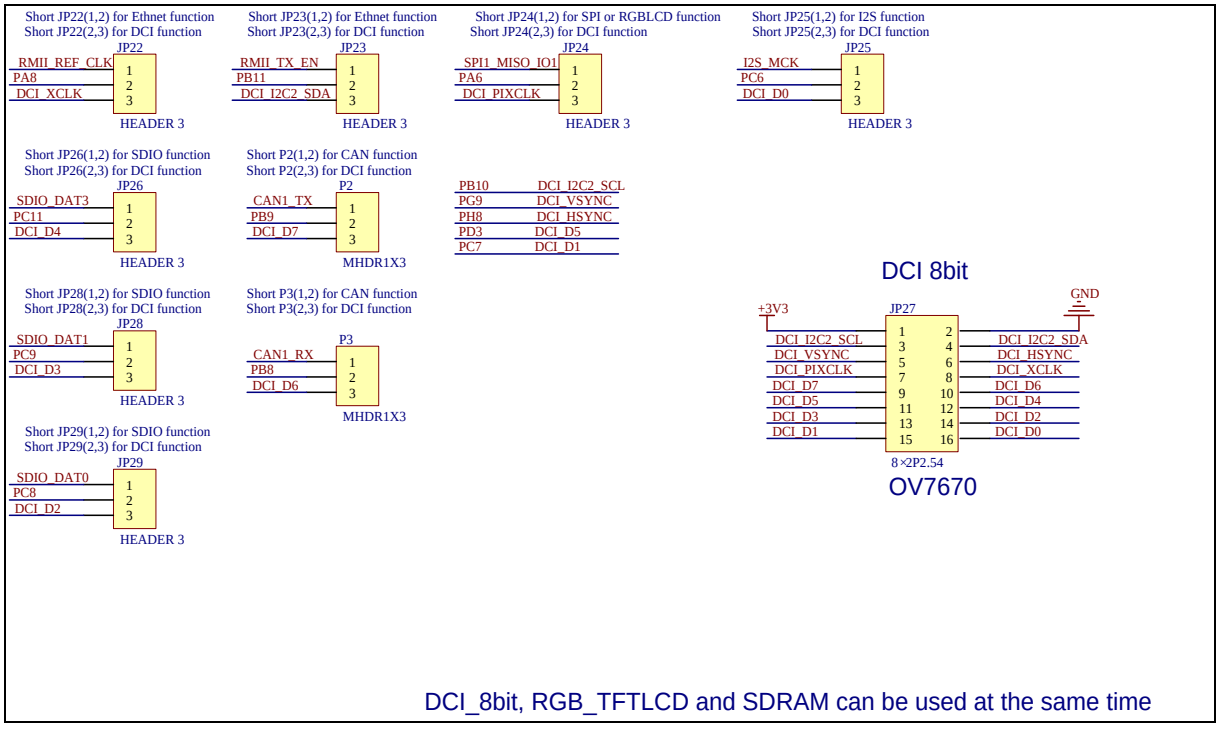
PD3——DCI_D5——OV7670 _D5

PB8——DCI_D6——OV7670 _D6

PB9——DCI_D7——OV7670 _D7

The following figure is GD32207I-EVAL-V1.0 development board Hardware connection of OV7670 module：

Figure 5-68 Schematic diagram of DCI



Software design：

Firstly, Camera Demo initialize the camera, then read the camera ID, and finally displayed on the TFT-LCD screen, at the same time ,the demo can take photos and return camera capture state use different key. There are four parts: camera initialization, image storage , image display and Take photos

Camera initialization

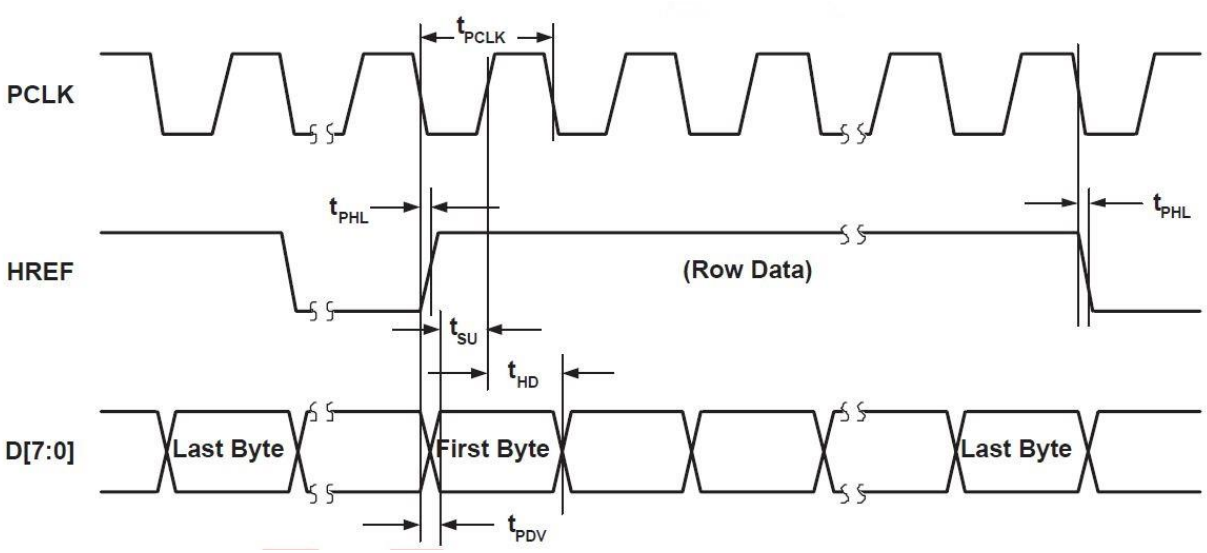
OV7670 camera through the SCCB interface to initialize the camera sensor internal registers, SCCB protocol compatible with I2C. Use I2C2 of GD32207I-EVAL-V1.0 board initialize the

camera sensor internal registers. In the initialization process you can change the parameters to set the camera's performance. The specific parameters are prefer to the OV7670 datasheet. In addition, the camera's clock input, provided by the PA8 pin of GD32207I-EVAL-V1.0 development board.

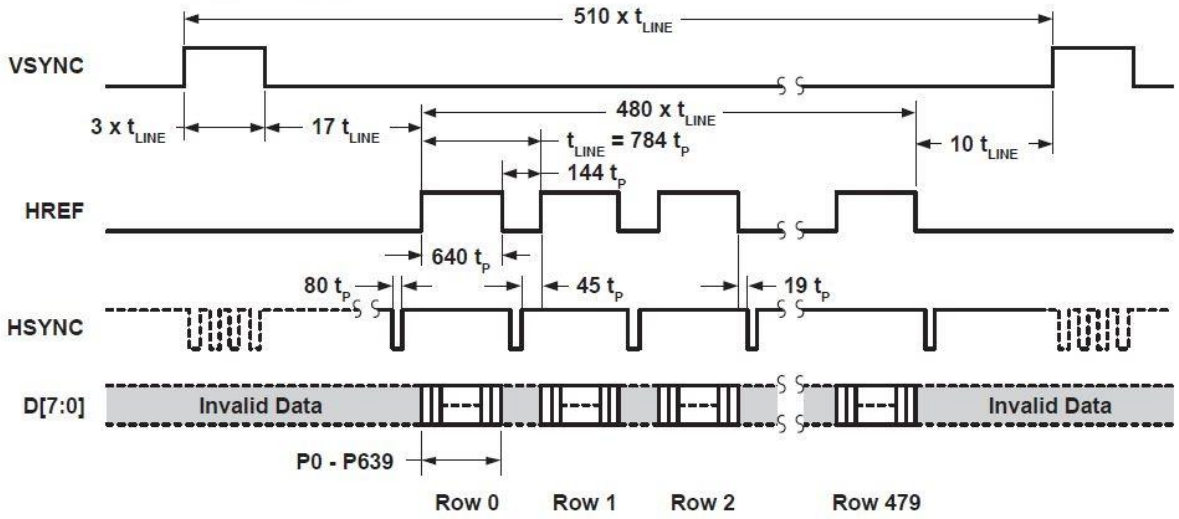
The configuration of the DCI camera interface needs to be configured according to the OV7670's output sequence:

```
/* DCI configuration ***********************************************************/

DCI_InitStructure.DCI_SnapshotMode = DCI_CAPMODE_CONTINUOUS;

DCI_InitStructure.DCI_SyncMode = DCI_SYNCMODE_HARDWARE;

DCI_InitStructure.DCI_CKPolarity = DCI_CKPOL_RISING;

DCI_InitStructure.DCI_VPolarity = DCI_VPOL_HIGH;

DCI_InitStructure.DCI_HPolarity = DCI_HPOL_LOW;

DCI_InitStructure.DCI_FrameRate = DCI_FRATE_ALL_FRAME;

DCI_InitStructure.DCI_InterfaceFormat = DCI_INTERFACEFORMAT_8B;
```

The following figure is the horizontal output timing diagram of OV7670:



The following figure is VGA frame output timing diagram of OV7670:



The output signal of OV7670: data is valid when the VSYNC low level, data is valid when the HREF high level, camera output data in the falling edge of PCLK. Therefore the DCI interface should be configured to: VSYNC pin high level is valid (data is invalid), the HSYNC pin is low level valid (data is invalid), in the rising edge of PIXCLK capture data.

Image storage

Because read speed of the inside FLASH is fast, but write speed is slow, so it is used to store the data that is not frequently modified, the background LOG images are converted into arrays stored in FLASH. And the internal SRAM capacity is limited, For the expansibility, the capture image through the DMA transmission to the SDRAM, and the displayed image data from the SDRAM. After the initialization of the SDRAM, you can write and read data directly to the corresponding address. SDRAM initialization and read or write operations, please refer to the introduction of SDRAM Demo.

Image display

The image shown part uses TLDI module displayed two level image. First layer display the 480*272 sizes GD LOG ,is the same as the LCD screen. The second layer is 240*272 image window , is the same as output image size. LCD display related information please refer to the TLDI display module.

Take photos

You can press the User key on development board to take photos. After press the key, the image will be stored in internal FLASH of the chip, waiting for the end of the image storage. Press the tamper key, read the photo from FLASH and displayed on the LCD screen. You can return to the camera capture state when press the Wakeup key on the development board. The previous photo will be covered after you take photo next time.

5.26.3. DEMO Implementation Result

Before running the Demo, connect jumper JP22/ JP23/JP24/ JP25/ JP26/JP28/JP29/ P2/P3

to DCMI and jumper JP31 to LCD on GD32207I-EVAL-V1.0 board. Download program to the board, the correct installation of LCD display and OV7670 camera to the development board. After power on, you can observe the capture image of camera displayed on the LCD screen, You can press the User key to take photo and press tamper key display photo. You can also return to the camera capture state when press the Wakeup key on the development board.

5.27. I2S Audio Play

5.27.1. DEMO Purpose

GD32207I-EVAL-V1.0 board integrated the I2S (Inter-IC Sound) module, and the module can communicate with external devices using the I2S audio protocol. This Demo mainly studies how to use the I2S interface of the board for audio output.

5.27.2. DEMO Principle

The Demo transmits audio data to the DA chip PCM1170 through the I2S GD32 interface, and sends the control information to the DA chip PCM1170 through the SPI1 interface. The program firstly initializes the I2S GD32 and SPI1 interface, and then the processor reads the audio files placed in their own, to determine whether the audio file format is legal. Program according to some parameters in the audio file sets the parameters of the I2S interface. Program through the SPI1 interface sends some control information to the DA chip PCM1170,

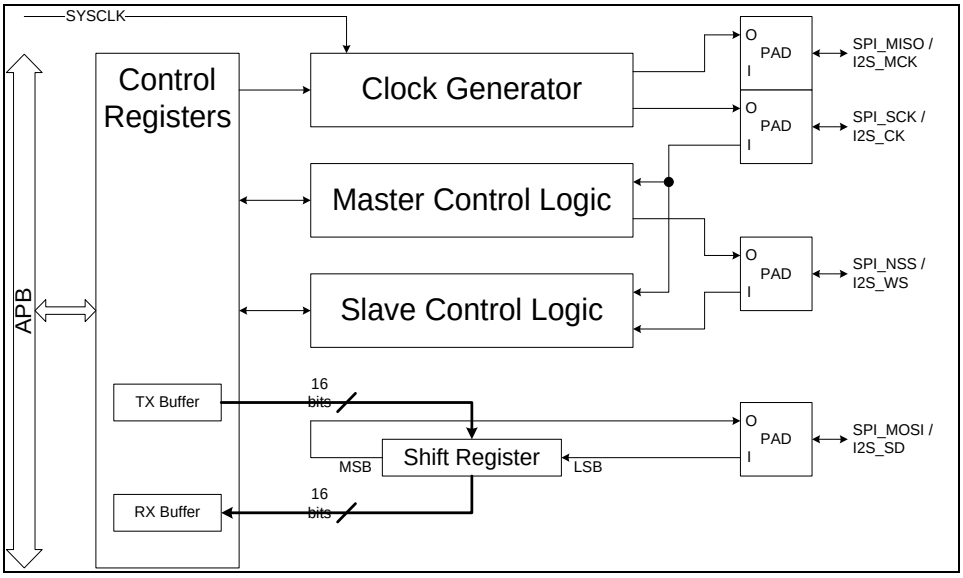
such as increasing the volume, reduce the volume, mute, etc.

I2S introduction:

GD32F20X I2S main features:

- Supported I2S standards:
- I2S Phillips standard
- MSB justified standard
- LSB justified standard
- PCM standard (both short and long frame synchronization mode)
- Supported operation modes:
- Master transmission
- Master reception
- Slave transmission
- Slave reception
- The data length can be 16 bits, 24 bits or 32 bits
- The channel length can be 16 bits or 32 bits
- 16-bit shift register for transmission and reception
- Data direction is always MSB first
- 8-bit programmable linear prescaler to reach accurate audio sample frequencies from 8 kHz to 192 kHz
- Programmable idle state clock polarity
- Master clock can be output to drive an external audio component
- Error flags including the transmission underrun error flag (TXURE) and the reception overrun error flag (RXORE)
- DMA capability for both transmission and reception

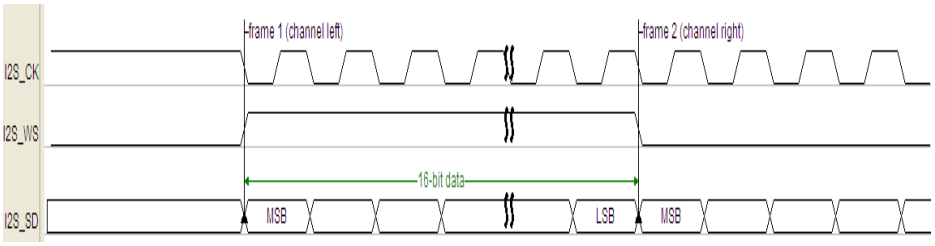
I2S diagram:



The I2S shares the same pins, flags, interrupts, data buffers, and shift register with SPI. When the I2SSEL bit in the SPI_I2SCTLR register is set, the resources are occupied by I2S. Or, they are used by SPI.

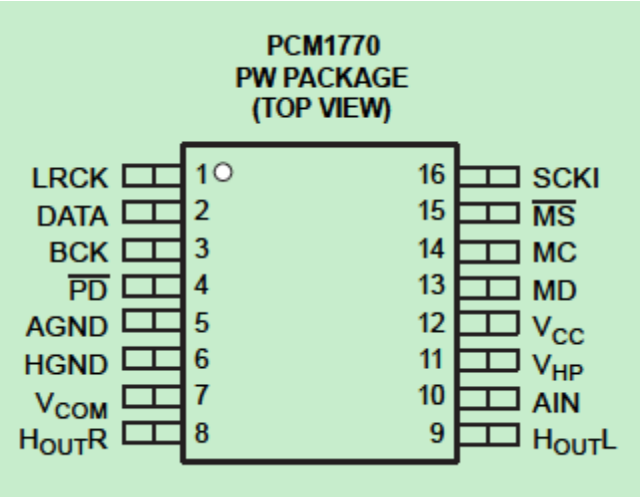
There are four pins on the I2S interface, including I2S_CK, I2S_WS, I2S_SD, and I2S_MCK. I2S_CK is the serial clock signal, which shares the same pin with SPI_SCK. I2S_WS is the data control signal, which shares the same pin with SPI_NSS. I2S_SD is the serial data signal, which shares the same pin with SPI_MOSI. I2S_MCK is the master clock signal, which shares the same pin with SPI_MISO. I2S_MCK is an optional signal for I2S interface. It produces a frequency rate equal to 256 x Fs, where Fs is the audio sampling frequency.

The inter-IC sound function supports four audio standards, including I2S Phillips standard, MSB justified standard, LSB justified standard, and PCM standard. It can operate in four modes, including master transmission mode, master reception mode, slave transmission mode, and slave reception mode. MSB justified standard timing diagram:



PCM1770 introduction:

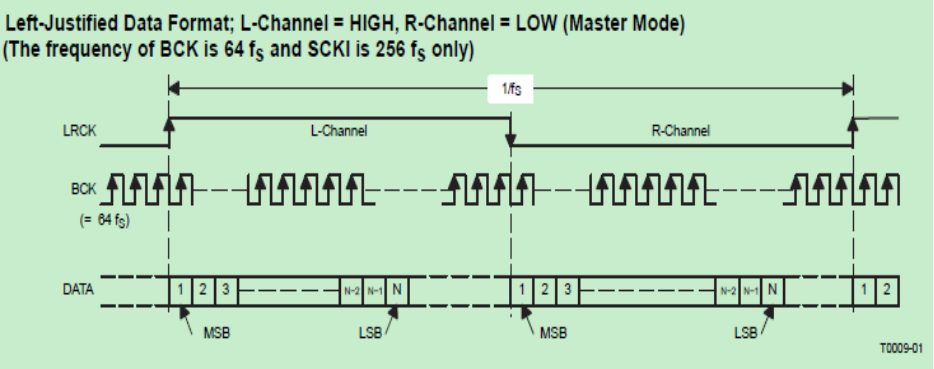
PCM1770 pin diagram as follows:



PCM1770 pins are defined as shown in the figure below:

TERMINAL NAME	NO.	I/O	DESCRIPTION
AGND	5	–	Analog ground. This is a return for V_{CC}.
AIN	10	I	Monaural analog signal mixer input. The signal can be mixed with the outputs of the L- and R-channel DACs.
BCK	3	I/O	Serial bit clock. Clocks the individual bits of the audio data input, DATA. In the slave interface mode, this clock is input from an external device. In the master interface mode, the PCM1770 device generates the BCK output to an external device.
DATA	2	I	Serial audio data input
HGND	6	–	Analog ground. This is a return for V_{HP}.
$H_{OUT}L$	9	O	L-channel analog signal output of the headphone amplifiers
$H_{OUT}R$	8	O	R-channel analog signal output of the headphone amplifiers
LRCK	1	I/O	Left and right clock. Determines which channel is being input on the audio data input, DATA. The frequency of LRCK must be the same as the audio sampling rate. In the slave interface mode, this clock is input from an external device. In the master interface mode, the PCM1770 device generates the LRCK output to an external device.
MC	14	I	Mode control port serial bit clock input. Clocks the individual bits of the control data input, MD.
MD	13	I	Mode control port serial data input. Controls the operation mode on the PCM1770 device.
$\overline{MS}$	15	I	Mode control port select. The control port is active when this terminal is low.
$\overline{PD}$	4	I	Reset input. When low, the PCM1770 device is powered down, and all mode control registers are reset to default settings.
SCKI	16	I	System clock input
V_{CC}	12	–	Power supply for all analog circuits except the headphone amplifier.
V_{COM}	7	–	Decoupling capacitor connection. An external 10-µF capacitor connected from this terminal to analog ground is required for noise filtering. Voltage level of this terminal is 0.5 V_{HP} nominal.
V_{HP}	11	–	Analog power supply for the headphone amplifier circuits. The voltage level must be the same as V_{CC}.

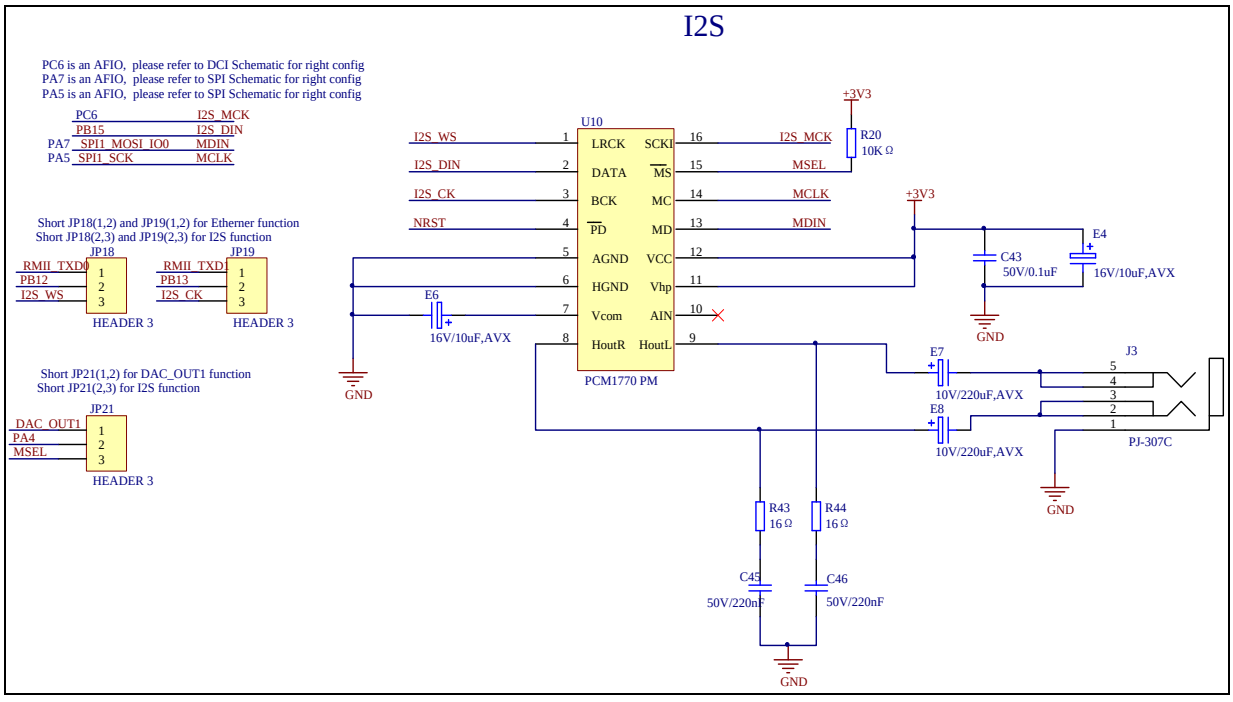
PCM1170 can directly drive headphones, with the software to control the volume size, chip mode and analog voice synthesis and other functions. PCM1170 supports industry standard audio data formats, including standard mode (standard), I2S PHILPS, MSB alignment (left-justified). In the Demo PCM1770 is in the slave equipment mode and the standard of MSB alignment. PCM1170 MSB alignment as follow:



Hardware connection:

Use of the main hardware resources are GD32 I2S interface, SPI1 interface and external DA chip PCM1770 in the Demo. The processor sends the audio data to the PCM1770 through the I2S interface, and PCM1770 decodes the audio data which is sended to J3. The processor accesses the PCM1770 register through the SPI1 interface to achieve configuration and control functions.

Figure 5-69 Schematic diagram of I2S



Note: before the experiment the with jumper cap connect JP18(2, 3), JP19(2, 3), JP21(2, 3),JP25(1,2),JP12(2，3),JP13(2，3).

5.27.3. DEMO Implementation Result

After downloading the program, insert the earphone into the audio port J3, then listen to the audio file. Press the "User" key, the volume decreases, and the state of the LED1 changes. Press the "Tamper" key, the volume increases, and the state of the LED2 changes. Press the "WakeUp" key, mute on or mute off, while the state of the LED3 changes.

6. Revision history

Table 6-1. Revision history

Revision No	Description	Date
0.0	Test version	15-Jul-2015