AN0110
应用笔记

AT32F403A/407 GPIO Application Note

前言

AT32F403A/407xx的通用功能I/O (GPIO)提供了一系列与外部环境通讯的接口，可用于MCU与其他嵌入式设备之间通过数字或模拟方式的通讯。AT32F403A/407系列的GPIO还提供了丰富I/O复用功能，能够使得多个外设可以同时工作。

参考资料：

- AT32F403A_407_Firmware_Library_V2.x.x\project\at_start_f403a\examples\gpio
- RM_AT32F403A_407 文档的通用和复用功能 I/O（GPIO 和 IOMUX）章节

注：本应用笔记对应的代码是基于雅特力提供的V2.x.x 板级支持包（BSP）而开发，对于其他版本BSP，需要注意使用上的区别。

支持型号列表：

支持型号	AT32F403A 系列
	AT32F407 系列

目录

表目录

图目录

1 GPIO 特性

- 最大封装（100pin）具有80个多功能双向的I/O口
- 所有I/O口都可以映射到16个外部中断
- 几乎所有I/O口可容忍5V输入信号（4个LEXT / HEXT引脚除外）
- 所有I/O口均为快速I/O，寄存器存取速度最高f_{AHB}
- I/O引脚的外设功能可以通过一个特定的操作锁定，以避免意外的写入I/O寄存器
- 每个GPIO引脚都可以由软件配置成输出(推挽或开漏)、输入(带或不带上拉或下拉)或复用的外设功能端口
- 可选的每个I/O口的电流推动/吸入能力
- GPIO设置/清除寄存器（GPIOx_SCR）和GPIO清除寄存器（GPIOx_CLR）为GPIOx_ODT寄存器提供位访问能力

2 GPIO

GPIO在复位期间和刚复位后，复用功能未开启，大部分I/O端口被配置成浮空输入模式。

当作为输出配置时，写到输出数据寄存器（GPIOx_ODT）上的值会输出到相应的I/O引脚。可以以推挽模式或开漏模式（仅低电平被驱动，高电平表现为高阻）使用输出驱动器。

输入数据寄存器（GPIOx_IDT）在每个AHB时钟周期捕捉I/O引脚上的数据。

所有GPIO引脚有一个内部弱上拉和弱下拉。

图 1. GPIO 基本结构

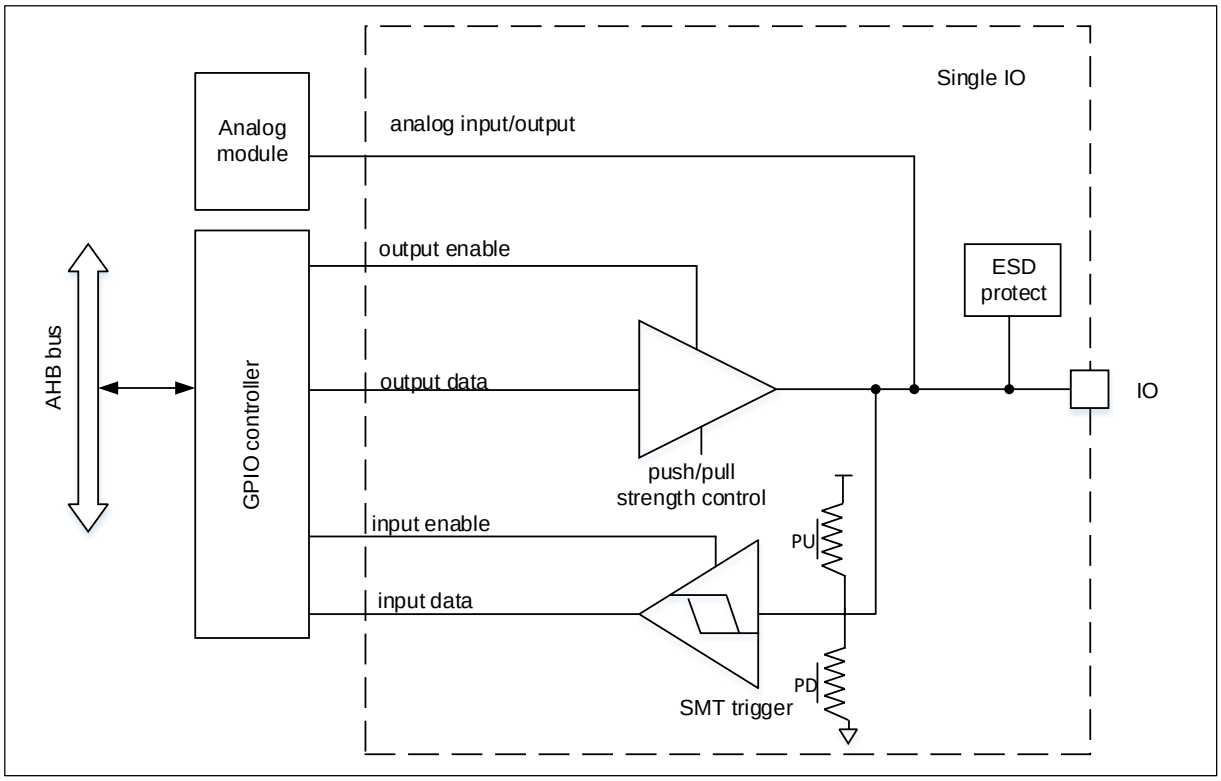


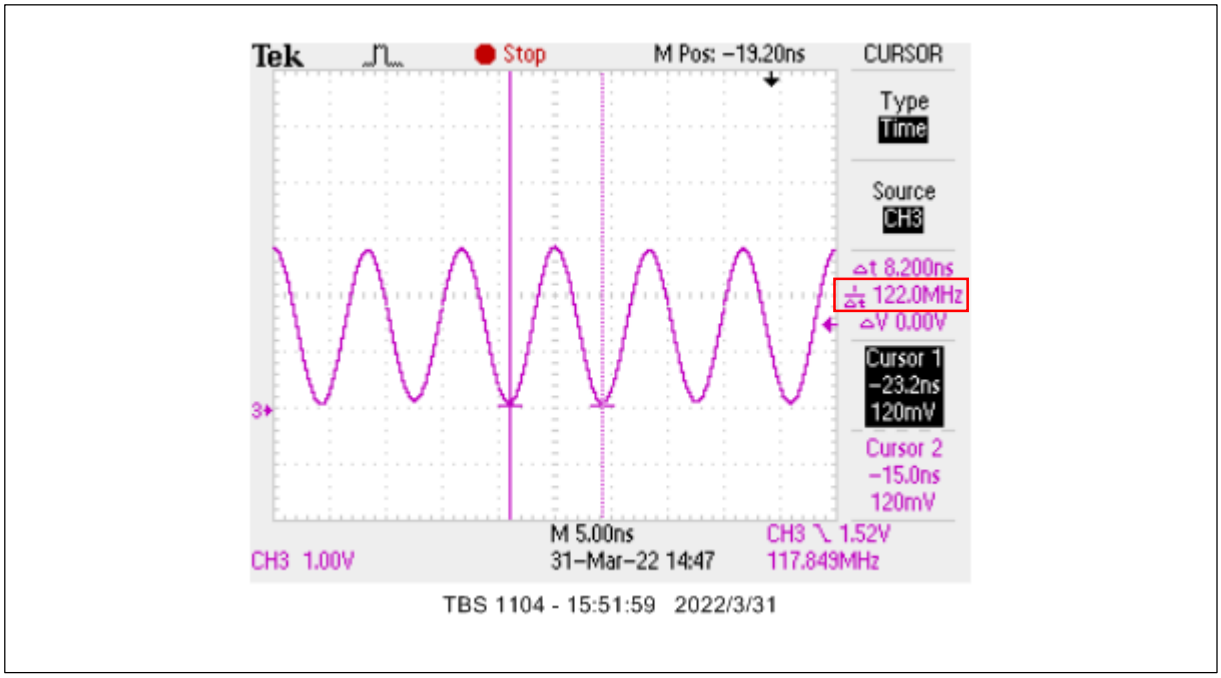
表 1. GPIO 配置表

配置模式	IOFC	HDRV	IOMC[1]	IOMC[0]	ODT 寄存器
浮空输入	01	000			不使用
下拉输入	10				0
上拉输入					1
模拟输入输出	00	000			不使用
推挽（Push-Pull）	00	000/100：输入模式 001：输出模式，较大电流推动/吸入能力 010：输出模式，适中电流推动/吸入能力 011：输出模式，适中电流推动/吸入能力 1xx：输出模式，极大电流推动/吸入能力			0 或 1
开漏（Open-Drain）	01				0 或 1

2.1 GPIO toggle

AT32F403A/407提供的I/O口均为快速I/O，寄存器存取速度最高为f_{AHB}，所以可以看到GPIO翻转频率能够轻松达到120MHz：

图 2. I/O 翻转速度



2.2 IO 引脚的 5V or 3.3V 容忍

2.2.1 标准 3.3V 容忍引脚（TC）

所有振荡器用到的引脚都是标准3.3V容忍引脚。

- PA0/PA11/PA12(WKUP/USBFS1_D-/USBFS1_D+)
- PC13 – PC15 (TAMPER-RTC/LEXT_IN/ OUT)
- PD0/PD1 (HEXT_IN/ OUT)

表 2. TC 引脚示例

引脚名称	引脚类型	IO电平	默认功能	重映设
LEXT_IN / PC14	I/O	TC	LEXT_IN	–

2.2.2 带模拟功能 5 V 容忍引脚（FTa）

ADC占用端口为带模拟功能5 V容忍引脚。

- PA1 – PA7
- PB0 – PB1
- PC0 – PC5
- FTa引脚设置为输入浮空、输入上拉、或输入下拉时，具有5V电平容忍特性；设置为模拟模式

时，不具5V电平容忍特性，此时输入电平必须小于VDD + 0.3V

表 3. FTa 引脚示例

引脚名称	引脚类型	IO电平	默认功能	重映设
PA1	I/O	FTa	ADC123_IN1 / USART2_RTS/ TMR2_CH2/ TMR5_CH2	UART4_RX

2.2.3 5V 容忍引脚（FT）

其余的GPIO都为5V容忍引脚。

表 4. FT 引脚示例

引脚名称	引脚类型	IO电平	默认功能	重映设
PA8	I/O	FT	CLKOUT / USART1_CK / I2C3_SCL / USBFS_SOF / SPIM_CS / TMR1_CH1	–

3 IOMUX

管脚作为复用输入功能时，与通用输入功能一样，端口配置成输入模式（浮空、上拉、下拉）。

要实现复用输出功能，必须配置GPIO配置低寄存器（GPIOx_CFGLR）或GPIO配置高寄存器（GPIOx_CFGHR）将该端口设定为复用功能输出模式（推挽或开漏）。此时管脚和GPIO控制器断开，由IOMUX控制器进行控制。

图 3. IOMUX 基本结构

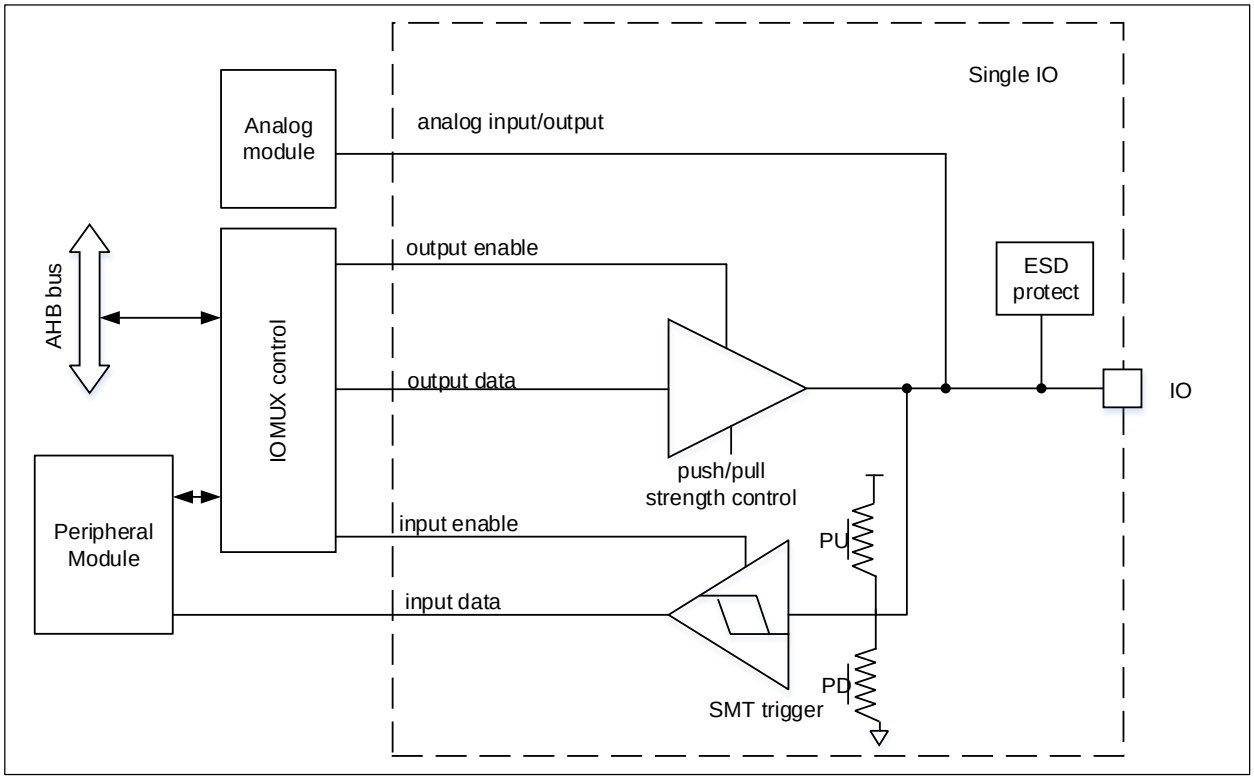


表 5. IOMUX 配置表

配置模式	IOFC	HDRV	IOMC[1]	IOMC[0]	ODT 寄存器
浮空输入	01	000			不使用
下拉输入	10				0
上拉输入					1
推挽（Push-Pull）	00	000/100：输入模式 001：输出模式，较大电流推动/吸入能力 010：输出模式，适中电流推动/吸入能力 011：输出模式，适中电流推动/吸入能力 1xx：输出模式，极大电流推动/吸入能力			不使用
开漏（Open-Drain）	01				不使用

3.1 复用功能输入配置

当I/O端口配置为复用功能输入时：

- 管脚状态可通过对输入数据寄存器的读访问得到
- 可配置管脚为浮空输入、上拉输入或下拉输入
- 施密特触发器有效
- 不能对该管脚进行输出。

3.2 复用功能输出或双向复用功能配置

当I/O端口配置为复用功能输出或双向复用功能时：

- 管脚输出由外设决定
- 施密特触发器有效
- 上拉和下拉电阻均关闭
- 如果管脚被误配成多个复用功能输出，管脚将按映射优先级输出，详见下面小节。
- 开漏模式时，读输入数据寄存器时可得到I/O口状态
- 推挽模式时，读输入数据寄存器时可得到I/O口状态

3.3 外设复用功能管脚配置

当外设需要使用IOMUX 复用功能时：

- 如果外设管脚需要作为复用输出则对应的管脚配置成复用推挽/开漏输出
- 如果外设管脚需要作为复用输入则对应的管脚配置成浮空输入/上拉输入/下拉输入
- ADC外设需要将模拟通道对应的管脚配置为模拟输入/输出模式
- I2C外设需要对应管脚作为双向复用功能时，需把对应的管脚配置复用开漏模式
- 如需配置外设复用映射还需调用crm_periph_clock_enable(CRM_IOMUX_PERIPH_CLOCK, TRUE);函数用以开启IOMUX时钟。

外设复用映射表如下：

表 6. 外设复用映射表

外设	复用定义	复用引脚
SPI1	SPI1_MUX_01/ SPI1_GMUX_0001	spi1_cs/i2s1_ws(pa15),spi1_sck/i2s1_ck(pb3), spi1_miso(pb4),spi1_mosi/i2s1_sd(pb5),i2s1_mck(pb0)
	SPI1_MUX_10/ SPI1_GMUX_0010	spi1_cs/i2s1_ws(pa4),spi1_sck/i2s1_ck(pa5), spi1_miso(pa6),spi1_mosi/i2s1_sd(pa7),i2s1_mck(pb6)
	SPI1_MUX_11/ SPI1_GMUX_0011	spi1_cs/i2s1_ws(pa15),spi1_sck/i2s1_ck(pb3), spi1_miso(pb4),spi1_mosi/i2s1_sd(pb5),i2s1_mck(pb6)
SPI2	SPI2_GMUX_0001	i2s2_mck(pa3)
	SPI2_GMUX_0010	i2s2_mck(pa6)
	SPI3_GMUX_0010	spi3_cs/i2s3_ws(pa15),spi3_sck/i2s3_ck(pb3), spi3_miso(pb4),spi3_mosi/i2s3_sd(pb5),i2s3_mck(pb10)

外设	复用定义	复用引脚
SPI3	SPI3_GMUX_0011	spi3_cs/i2s3_ws(pa4),spi3_sck/i2s3_ck(pc10), spi3_miso(pc11),spi3_mosi/i2s3_sd(pc12),i2s3_mck(pb10)
	SPI3_MUX/ SPI3_GMUX_0001	spi3_cs/i2s3_ws(pa4),spi3_sck/i2s3_ck(pc10), spi3_miso(pc11),spi3_mosi/i2s3_sd(pc12),i2s3_mck(pc7)
SPI4	SPI4_MUX/ SPI4_GMUX_0001	spi4_cs/i2s4_ws(pe12),spi4_sck/i2s4_ck(pe11), spi4_miso(pe13),spi4_mosi/i2s4_sd(pe14),i2s4_mck(pc8)
	SPI4_GMUX_0010	spi4_cs/i2s4_ws(pb6),spi4_sck/i2s4_ck(pb7), spi4_miso(pb8),spi4_mosi/i2s4_sd(pb8),i2s4_mck(pc8)
	SPI4_GMUX_0011	spi4_cs/i2s4_ws(pb6),spi4_sck/i2s4_ck(pb7), spi4_miso(pb8),spi4_mosi/i2s4_sd(pb8),i2s4_mck(pa10)
I2C1	I2C1_MUX/ I2C1_GMUX_0001	i2c1_scl(pb8),i2c1_sda(pb9)
I2C3	I2C3_MUX/ I2C3_GMUX_0001	i2c3_scl(pa8),i2c3_sda(pb4)
USART1	USART1_MUX/ USART1_GMUX_0001	usart1_tx(pb6),usart1_rx(pb7)
USART2	USART2_MUX/ USART2_GMUX_0001	usart2_tx(pd5),usart2_rx(pd6),usart2_ck(pd7), usart2_cts(pd3),usart2_rts(pd4)
USART3	USART3_MUX_01/ USART3_GMUX_0001	usart3_tx(pc10),usart3_rx(pc11),usart3_ck(pc12), usart3_cts(pb13),usart3_rts(pb14)
	USART3_MUX_11/ USART3_GMUX_0011	usart3_tx(pd8),usart3_rx(pd9),usart3_ck(pd10), usart3_cts(pd11),usart3_rts(pd12)
UART4	UART4_GMUX_0010	uart4_tx(pa0),uart4_rx(pa1)
UART5	UART5_GMUX_0001	uart5_tx(pb9),uart5_rx(pb8)
USART6	USART6_GMUX	usart6_tx(pa4),usart6_rx(pa5)
UART7	UART7_GMUX	uart7_tx(pb4),uart7_rx(pb3)
UART8	UART8_GMUX	uart8_tx(pc2),uart8_rx(pc3)
TMR1	TMR1_MUX_01/ TMR1_GMUX_0001	tmr1_ext(pa12),tmr1_ch1(pa8),tmr1_ch2(pa9), tmr1_ch3(pa10),tmr1_ch4(pa11),tmr1_brkin(pa6), tmr1_ch1c(pa7),tmr1_ch2c(pb0),tmr1_ch3c(pb1)
	TMR1_MUX_11/ TMR1_GMUX_0011	tmr1_ext(pe7),tmr1_ch1(pe9),tmr1_ch2(pe11), tmr1_ch3(pe13),tmr1_ch4(pe14),tmr1_brkin(pe15), tmr1_ch1c(pe8),tmr1_ch2c(pe10),tmr1_ch3c(pe12)
TMR2	TMR2_MUX_01/ TMR2_GMUX_01	tmr2_ch1_ext(pa15),tmr2_ch2(pb3),tmr2_ch3(pa2),tmr2_ch4(pa3)
	TMR2_MUX_10/ TMR2_GMUX_10	tmr2_ch1_ext(pa0),tmr2_ch2(pa1), tmr2_ch3(pb10),tmr2_ch4(pb11)
	TMR2_MUX_11/ TMR2_GMUX_11	tmr2_ch1_ext(pa15),tmr2_ch2(pb3), tmr2_ch3(pb10),tmr2_ch4(pb11)
	TMR2ITR1_MUX/ TMR2ITR1_GMUX_10	ethernet ptp as input to tmr2_itr1
	TMR2ITR1_GMUX_11	usbdev sof as input to tmr2_itr1
	TMR3_MUX_10/ TMR3_GMUX_0010	tmr3_ch1(pb4),tmr3_ch2(pb5),tmr3_ch3(pb0),tmr3_ch4(pb1)

外设	复用定义	复用引脚
TMR3	TMR3_MUX_11/ TMR3_GMUX_0011	tmr3_ch1(pc6),tmr3_ch2(pc7),tmr3_ch3(pc8),tmr3_ch4(pc9)
TMR4	TMR4_MUX/ TMR4_GMUX_0001	tmr4_ch1(pd12),tmr4_ch2(pd13),tmr4_ch3(pd14),tmr4_ch4(pd15)
TMR5	TMR5CH4_MUX/ TMR5CH4_GMUX	lick connected to tmr5_ch4 input capture for calibration
TMR9	TMR9_MUX/ TMR9_GMUX	tmr9_ch1(pe5),tmr9_ch2(pe6)
CAN	CAN_MUX_00/ CAN1_GMUX_0000	can_rx(pa11),can_tx(pa12)
	CAN_MUX_10/ CAN1_GMUX_0010	can_rx(pb8),can_tx(pb9)
	CAN_MUX_11/ CAN1_GMUX_0011	can_rx(pd0),can_tx(pd1)
CAN2	CAN2_MUX/ CAN2_GMUX_0001	can2_rx(pb5),can2_tx(pb6)
PD01	PD01_MUX/ PD01_GMUX	pd0/pd1 mapping on osc_in/osc_out
ADC1	ADC1_ETP_MUX/ ADC1_ETP_GMUX	adc1 external trigger preempted conversion muxing
	ADC1_ETO_MUX/ ADC1_ETO_GMUX	adc1 external trigger ordinary conversion muxing
ADC2	ADC2_ETP_MUX/ ADC2_ETP_GMUX	adc2 external trigger preempted conversion muxing
	ADC2_ETO_MUX/ ADC2_ETO_GMUX	adc2 external trigger ordinary conversion muxing
EMAC	EMAC_MUX/ EMAC_GMUX_01	rx_dv/crs_dv(pd8), rxd0(pd9), rxd1(pd10), rxd2(pd11), rxd3(pd12)
	MII_RMII_SEL_MUX/ MII_RMII_SEL_GMUX	mii or rmii selection
	PTP_PPS_MUX/ PTP_PPS_GMUX	ethernet ptp pps mux function remapping
SWJTAG	SWJTAG_MUX_001/ SWJTAG_GMUX_001	full swj enabled (jtag-dp + sw-dp) but without jtrst
	SWJTAG_MUX_010/ SWJTAG_GMUX_010	jtag-dp disabled and sw-dp enabled
	SWJTAG_MUX_100/ SWJTAG_GMUX_100	full swj disabled (jtag-dp + sw-dp)
XMC	XMC_NADV_MUX/ XMC_NADV_GMUX	xmc_nadv not used
	XMC_GMUX_001	xmc_nwe(pd2),xmc_d0(pb14),xmc_d1(pc6),xmc_d2(pc11), xmc_d3(pc12),xmc_d4(pa2),xmc_d5(pa3),xmc_d6(pa4), xmc_d7(pa5),xmc_d13(pb12),xmc_noe(pc5)
		xmc_nwe(pc2),xmc_d0(pb14),xmc_d1(pc6),xmc_d2(pc11),

外设	复用定义	复用引脚
	XMC_GMUX_010	xmc_d3(pc12),xmc_d4(pa2),xmc_d5(pa3),xmc_d6(pa4), xmc_d7(pa5),xmc_d13(pb12),xmc_noe(pc5)
SDIO2	SDIO2_MUX01/ SDIO2_GMUX_0001	sdio2_ck(pc4),sdio2_cmd(pc5),sdio2_d0(pa4), sdio2_d1(pa5),sdio2_d2(pa6),sdio2_d3(pa7)
	SDIO2_MUX10/ SDIO2_GMUX_0010	sdio2_ck(pa2),sdio2_cmd(pa3),sdio2_d0(pc0), sdio2_d1(pc1),sdio2_d2(pc2),sdio2_d3(pc3), sdio2_d4(pa4),sdio2_d5(pa5),sdio2_d6(pa6),sdio2_d7(pa7)
	SDIO2_MUX11/ SDIO2_GMUX_0011	sdio2_ck(pa2),sdio2_cmd(pa3),sdio2_d0(pa4), sdio2_d1(pa5),sdio2_d2(pa6),sdio2_d3(pa7)
EXT_ SPIM	EXT_SPIM_EN_MUX/ EXT_SPIM_GMUX_1000	enable external spi-flash interface
	EXT_SPIM_GMUX_1001	spim_sck(pb1),spim_cs(pa8),spim_io0(pb10), spim_io1(pb11),spim_io2(pb7),spim_sio3(pb6)

3.4 IOMUX 映射优先级

单个管脚可能有多个外设复用映射，当多个外设复用映射到同一个管脚时，外设遵循以下优先级规则：

- 硬件抢占功能优先
- JTAG调试端口优先
- 非timer外设复用映射优先于timer外设。
- 多个非timer外设之间复用映射无优先级关系，复用功能叠加到同一个管脚。

3.4.1 硬件抢占功能

某些管脚不管GPIO配置为任何模式，都会被特定的硬件功能占用。

表 7. FT 引脚示例

引脚名称	引脚类型	IO电平
PA0	PWC_CTRLSTS[8] = 1	抢占使能位有效之后，PA0管脚直接作为PWC 的WKUP功能使用
PA4	DAC_CTRL[2] =1	抢占使能位有效之后，PA4作为DAC1模拟通道使用
PA5	DAC_CTRL[18] =1	抢占使能位有效之后，PA5作为DAC2模拟通道使用
PA11	CRM_APB1EN[23]=1	抢占使能位有效之后，PA11作为USB_DM 通道使用
PA12	CRM_APB1EN[23]=1	抢占使能位有效之后，PA12作为USB_DP 通道使用
PC13	CRM_APB1EN[27]=1& (BPR_CTRL[0] = 1 | BPR_RTCCAL[8] = 1 | BPR_RTCCAL[7] = 1)	抢占使能位有效之后，PC13作为RTC 通道使用
PC14	CMR_BPDC[0]=1	抢占使能位有效之后，PC14作为LEXT 通道使用
PC15	CMR_BPDC[0]=1	抢占使能位有效之后，PC15作为LEXT 通道使用

3.4.2 调试端口优先

在进行芯片调试时，为了防止其他外设对调试端口的干扰，导致不能被调试，配置好后的调试端口管脚，不管对应管脚的GPIO寄存器被配置为任何模式，都会一直保持为调试端口。

通过设置复用重映射和调试I/O配置寄存器（IOMUX_REMAP）的SWJTAG_MUX [2：0]位或复用重映射和调试I/O配置寄存器7（IOMUX_REMAP7）的SWJTAG_GMUX [2：0]位，可以改变上述重映射配置

3.4.3 其他外设输出优先级关系

除了硬件抢占功能和端口调试功能以外，其他外设输出优先级如下：

- 非timer外设输出优先于timer外设，即其他外设和timer同时映射到某一管脚，timer不能输出。
- 多个非timer外设输出如果映射到同一管脚，则这些外设输出会叠加输出到该管脚

3.5 外部中断/唤醒线

每个管脚都支持作为外部中断的输入，对应的管脚须配置为输入模式。

4 GPIO 固件驱动程序 API

Artery 提供的固件驱动程序包含了一系列固件函数来管理 GPIO 的下列功能：

- 初始化配置
- 读取输入端口或某个输入引脚
- 读取输出端口或某个输出引脚
- 设置或清除某个引脚的输出
- 锁定引脚
- 引脚的复用功能配置

注：所有project都是基于keil 5而建立，若用户需要在其他编译环境上使用，请参考AT32xxx_Firmware_Library_V2.x.x\project\at_start_xxx\templates中各种编译环境（例如IAR6/7,keil 4/5）进行简单修改即可。

4.1 输出模式

GPIO提供了两种不同类型的输出模式分别是，推挽输出以及开漏输出，下面是输出模式的配置示例：

```
gpio_init_struct.gpio_pins = GPIO_PINS_x;
gpio_init_struct.gpio_mode = GPIO_MODE_OUTPUT; /* 选择输出 */
gpio_init_struct.gpio_pull = GPIO_PULL_NONE;    /* 无、上拉或下拉 */
gpio_init_struct.gpio_out_type = GPIO_OUTPUT_PUSH_PULL; /* 选择推挽或开漏 */
gpio_init_struct.gpio_drive_strength = GPIO_DRIVE_STRENGTH_STRONGER;
gpio_init(GPIOy, &gpio_init_struct);
```

4.2 输入模式

GPIO提供了三种不同类型的输入模式分别是，浮空输入、上拉输入以及下拉输入，下面是输入模式的配置示例：

```
gpio_init_struct.gpio_pins = GPIO_PINS_x;
gpio_init_struct.gpio_mode = GPIO_MODE_INPUT; /* 选择输入*/
gpio_init_struct.gpio_pull = GPIO_PULL_NONE;    /* 无、上拉或下拉 */
gpio_init_struct.gpio_drive_strength = GPIO_DRIVE_STRENGTH_STRONGER;
gpio_init(GPIOy, &gpio_init_struct);
```

4.3 模拟模式

当需要使用ADC或CMP通道作为输入时，需要将相应的引脚配置为模拟模式，下面是模拟模式的配置示例：

```
gpio_init_struct.gpio_pins = GPIO_PINS_x;
```

```
gpio_init_struct.gpio_mode = GPIO_MODE_ANALOG; /* 选择模拟输入*/
gpio_init_struct.gpio_pull = GPIO_PULL_NONE;    /* 无、上拉或下拉 */
gpio_init_struct.gpio_drive_strength = GPIO_DRIVE_STRENGTH_STRONGER;
gpio_init(GPIOy, &gpio_init_struct);
```

4.4 复用模式

1. 每个管脚都通过软件配置GPIO配置低寄存器（GPIOx_CFGLR）或GPIO配置高寄存器（GPIOx_CFGHR）设定成复用功能输入输出端口。
2. 大多数管脚支持多个外设的输出功能映射，可通过IOMUX REMAP寄存器来选择不同的外设输入输出功能。
3. 每个管脚会绑定在一组外设复用下，注意有可能一个引脚会有两种以上外设使用，此时会根据前述3.4章节的IOMUX映射优先级来分配外设控制IO，所以请尽量避开多个外设映射到同一个引脚的状况。
4. 由于IOMUX实际上是在给外设分配不同“组别”的IO引脚，所以无法做到单独针对某一个IO进行配置。

根据上述配置过程，下面将介绍几种外设的常用配置示例。

4.4.1 USART I/O 复用模式配置

```
/* 使能 GPIOA 的时钟 */
crm_periph_clock_enable(CRM_GPIOB_PERIPH_CLOCK, TRUE);
crm_periph_clock_enable(CRM_IOMUX_PERIPH_CLOCK, TRUE);

/* 将 USART1的TX和RX重映设到PB6，PB7，复用映射定义参见表6 */
gpio_pin_remap_config(USART1_MUX, TRUE);

/* 将 USART1_Tx 和 USART1_Rx 配置为复用功能 */
gpio_init_struct.gpio_pins = GPIO_PINS_6 | GPIO_PINS_7;
gpio_init_struct.gpio_mode = GPIO_MODE_MUX;
gpio_init_struct.gpio_pull = GPIO_PULL_NONE;
gpio_init_struct.gpio_out_type = GPIO_OUTPUT_PUSH_PULL;
gpio_init_struct.gpio_drive_strength = GPIO_DRIVE_STRENGTH_STRONGER;
gpio_init(GPIOB, &gpio_init_struct);
```

4.4.2 TMR I/O 复用模式配置

```
/* 使能 GPIOA 的时钟 */
```

```
crm_periph_clock_enable(CRM_GPIOA_PERIPH_CLOCK, TRUE);
crm_periph_clock_enable(CRM_IOMUX_PERIPH_CLOCK, TRUE);

/* 将TMR1的BRK、CH1和EXT连接到PA6、PA8和PA12，复用映射定义参见表6 */
gpio_pin_remap_config(TMR1_MUX_01, TRUE);

/* 将以上TMR1引脚(PA6/PA8/PA12)配置为复用功能 */
gpio_init_struct.gpio_pins = GPIO_PINS_6 | GPIO_PINS_8 | GPIO_PINS_12;
gpio_init_struct.gpio_mode = GPIO_MODE_MUX;
gpio_init_struct.gpio_pull = GPIO_PULL_NONE;
gpio_init_struct.gpio_out_type = GPIO_OUTPUT_PUSH_PULL;
gpio_init_struct.gpio_drive_strength = GPIO_DRIVE_STRENGTH_STRONGER;
gpio_init(GPIOA, &gpio_init_struct);
```

4.4.3 I2C I/O 复用模式配置

```
/* 使能 GPIOB 的时钟 */
crm_periph_clock_enable(CRM_GPIOB_PERIPH_CLOCK, TRUE);
crm_periph_clock_enable(CRM_IOMUX_PERIPH_CLOCK, TRUE);

/* 将I2C1的SCL和SDA连接到PB8和PB9，复用映射定义参见表6 */
gpio_pin_remap_config(I2C1_MUX, TRUE);

/* 将 I2C1_SCL 和 I2C1_SDA 配置为复用功能，注意I2C引脚的输出类型应为开漏输出 */
gpio_init_struct.gpio_pins = GPIO_PINS_8 | GPIO_PINS_9;
gpio_init_struct.gpio_mode = GPIO_MODE_MUX;
gpio_init_struct.gpio_pull = GPIO_PULL_NONE;
gpio_init_struct.gpio_out_type = GPIO_OUTPUT_OPEN_DRAIN;
gpio_init_struct.gpio_drive_strength = GPIO_DRIVE_STRENGTH_STRONGER;
gpio_init(GPIOB, &gpio_init_struct);
```

5 版本历史

表 8. 文档版本历史

日期	版本	变更
2022.04.02	2.0.0	最初版本

重要通知 – 请仔细阅读

买方自行负责对本文所述雅特力产品和服务的选择和使用，雅特力概不承担与选择或使用本文所述雅特力产品和服务相关的任何责任。

无论之前是否有过任何形式的表示，本文档不以任何方式对任何知识产权进行任何明示或默示的授权或许可。如果本文档任何部分涉及任何第三方产品或服务，不应被视为雅特力授权使用此类第三方产品或服务，或许可其中的任何知识产权，或者被视为涉及以任何方式使用任何此类第三方产品或服务或其中任何知识产权的保证。

除非在雅特力的销售条款中另有说明，否则，雅特力对雅特力产品的使用和/或销售不做任何明示或默示的保证，包括但不限于有关适销性、适合特定用途（及其依据任何司法管辖区的法律的对应情况），或侵犯任何专利、版权或其他知识产权的默示保证。

雅特力产品并非设计或专门用于下列用途的产品：（A）对安全性有特别要求的应用，例如：生命支持、主动植入设备或对产品功能安全有要求的系统；（B）航空应用；（C）航天应用或航天环境；（D）武器，且/或（E）其他可能导致人身伤害、死亡及财产损害的应用。如果采购商擅自将其用于前述应用，即使采购商向雅特力发出了书面通知，风险及法律责任仍将由采购商单独承担，且采购商应独力负责在前述应用中满足所有法律和法规要求。

经销的雅特力产品如有不同于本文档中提出的声明和/或技术特点的规定，将立即导致雅特力针对本文所述雅特力产品或服务授予的任何保证失效，并且不应以任何形式造成或扩大雅特力的任何责任。