AN0096
应用笔记

AT32F435/437xx GPIO 使用指南

前言

AT32F435/437xx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。AT32F435/437xx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，并且保证每个引脚在某一时刻只会连接到一个外设，从而避免了外设冲突的产生。

参考资料：

- AT32F435_437_Firmware_Library_V2.x.x\project\at_start_f437\examples\gpio
- RM_AT32F435_437 文档的通用和复用功能 I/O（GPIO 和 IOMUX）章节

支持型号列表：

支持型号	AT32F435 系列
	AT32F437 系列

目录

表目录

图目录

1 GPIO 特性

- 最大封装（144pin）具有116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；
- 绝大部分I/O口可容忍5V输入信号；
- 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来开启写保护，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；
- 端口位设置/清除寄存器（GPIOx_SCR）和端口位清除寄存器（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，它们被激活或断开有赖于GPIOx_PULL寄存器的值。

图 1. I/O 端口位的基本结构

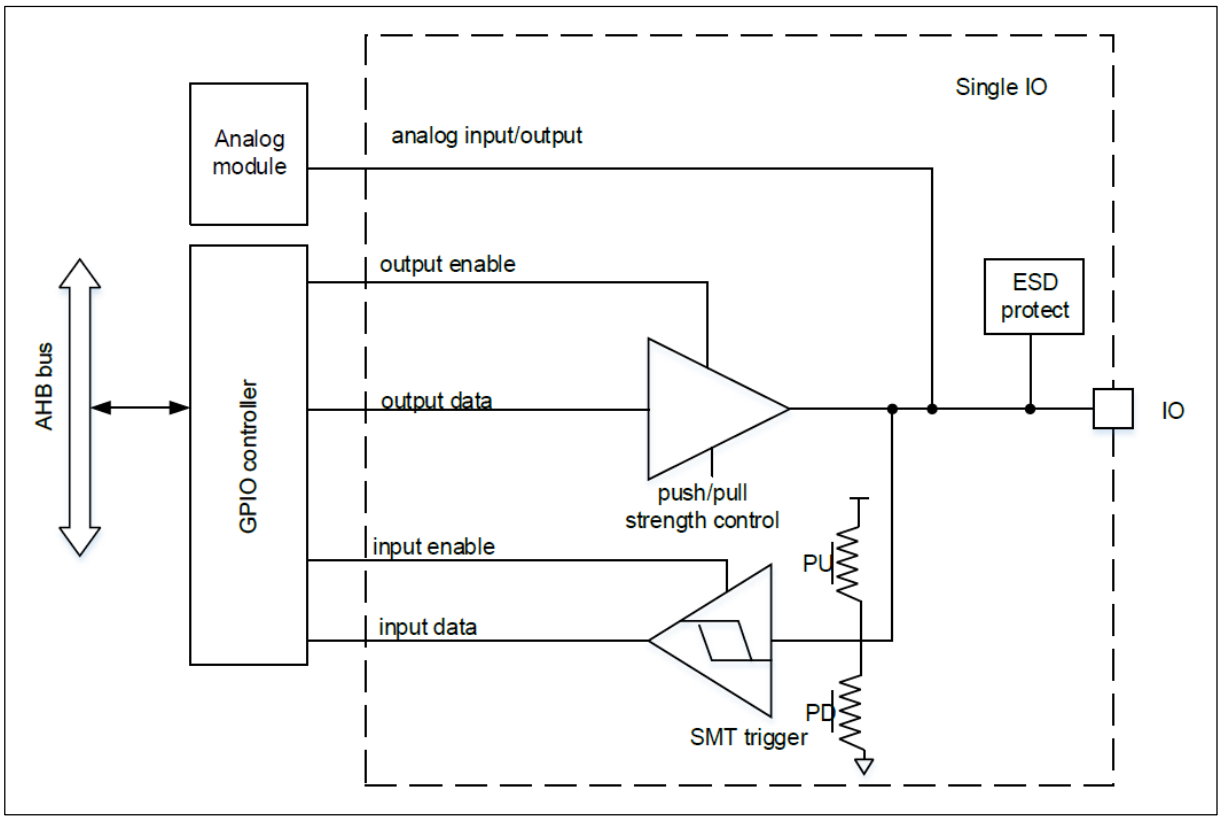


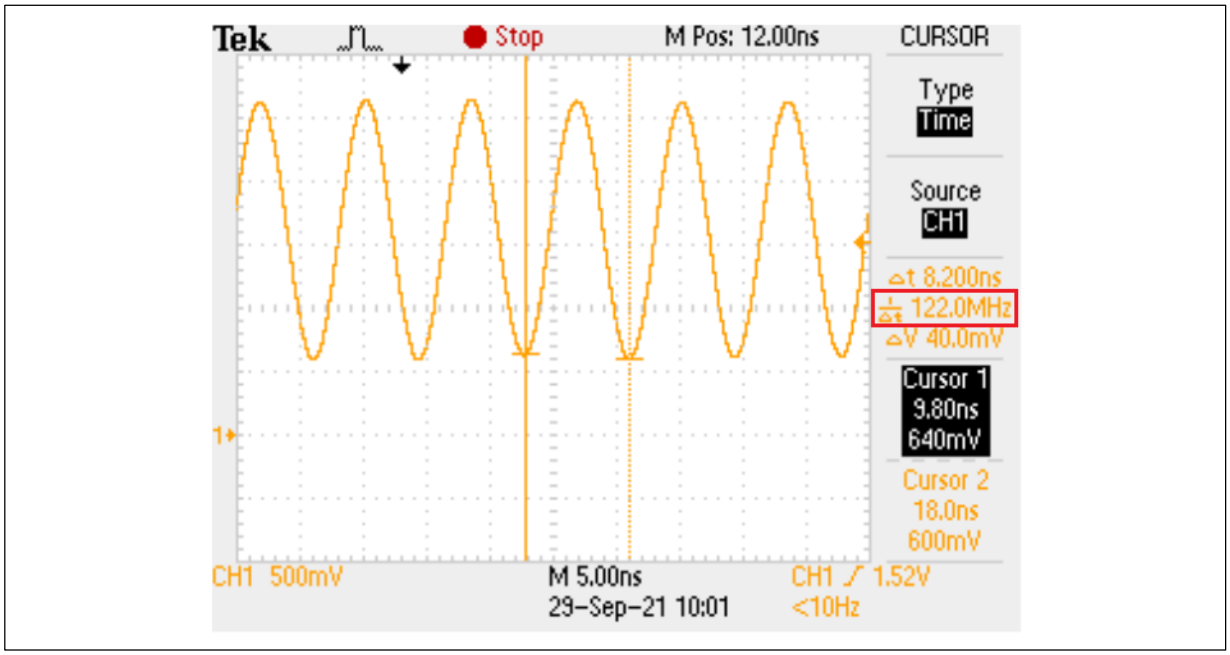
表 1. I/O 端口位配置表

配置模式	IOMC	OM	HDRV	ODRV[1：0]	PULL
通用浮空输入	00	不使用	不使用	不使用	00
通用下拉输入					10
通用上拉输入					01
模拟输入输出	11	不使用	不使用	不使用	不使用
通用推挽无上拉/下拉	01	0	000：适中电流推动/吸入能力 001：较大电流推动/吸入能力 01x：适中电流推动/吸入能力 1xx：极大电流推动/吸入能力		00
通用推挽上拉					01
通用推挽下拉					10
通用开漏无上拉/下拉		1			00
通用开漏上拉					01
通用开漏下拉					10
复用浮空输入	10	不使用	不使用		00
复用上拉输入					01
复用下拉输入					10

2.1 GPIO toggle

AT32F435/437提供的I/O口均为快速I/O，寄存器存取速度最高为f_{AHB}，所以可以看到在主频为240MHz时，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）

所有振荡器和USB_OTG用到的引脚都是标准3.3V容忍引脚。

- PC14/PC15 (LEXT_IN / OUT)
- PH0/PH1 (HEXT_IN/ OUT)
- PA11/PA12 (OTGFS1_D-/ D+)
- PB14/PB15 (OTGFS2_D-/ D+)

表 2. TC 引脚示例

引脚名称	引脚类型	IO结构	引脚多工功能	附加功能
PH0/HEXT_IN(PH0)	I/O	TC	I2C1_SDA	HEXT_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。

- PA0 – PA7，PB0 – PB1，PC0 – PC5，PF3 – PF10
- 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结构	引脚多工功能	附加功能
PA0	I/O	FTa	TMR1_ETR / USART2_CTS /I2C2_SCL / COMP_OUT	ADC_IN0COMP_INP2 / COMP_INM6 / WKUP1

2.2.3 带 20mA 吸入能力 5V 容忍引脚（FTf）

部分I2C可提供带20mA吸入能力的5V容忍引脚。

表 4. FT 引脚示例

引脚名称	引脚类型	IO结构	引脚多工功能	附加功能
PB9	I/O	FTf	IR_OUT / TMR2_CH2 / TMR4_CH4 / TMR11_CH1 / I2C1_SDA / SPI2_CS/I2S2_WS / SPI4_MOSI / I2S4_SD / I2C2_SDA / UART5_TX / CAN1_TX / QSPI1_CS / SDIO1_D5 / DVP_D7	-

2.2.4 5V 容忍引脚（FT）

其余的GPIO都为5V容忍引脚。

表 5. FT 引脚示例

引脚名称	引脚类型	IO结构	引脚多工功能	附加功能
PA8	I/O	FT	TMR1_CH1 / USART1_CK / UART2_TX / I2C2_SCL / CLKOUT / EVENTOUT	-

3 IOMUX

3.1 I/O 复用功能输入/输出

- 大多数外设共享同一个GPIO引脚（比如PA0，可作为TMR2_CH1 / TMR2_EXT / TMR5_CH1 / TMR8_EXT / I2C2_SCL / USART2_CTS）
- 而对某个具体的GPIO引脚，在任意时刻只有一个外设能够与之相连
- 某些外设功能还可以重映射到其他引脚，从而使得能同时使用的外设数量更多

选择每个端口线的有效复用功能之一是由两个寄存器来决定的，分别是GPIOx_MUXL和GPIOx_MUXH复用功能寄存器。可根据应用的需求用这两寄存器连接复用功能模块到其他引脚。

表 6. 通过 GPIOA_AFR 寄存器配置端口 A 的复用功能

引脚名	MUX0	MUX1	MUX2	MUX3	MUX4	MUX5	MUX6	MUX7
PA0		TMR2_CH1 TMR2_EXT	TMR5_CH1	TMR8_EXT	I2C2_SCL			USART2_CTS
PA1		TMR2_CH2	TMR5_CH2		I2C2_SDA	SPI4_MOSI I2S4_SDEXT		USART2_RTS_DE
PA2		TMR2_CH3	TMR5_CH3	TMR9_CH1				USART2_TX
PA3		TMR2_CH4	TMR5_CH4	TMR9_CH2		I2S2_MCK		USART2_RX
PA4						SPI1_CS I2S1_WS	SPI3_CS I2S3_WS	USART2_CK
PA5		TMR2_CH1 TMR2_EXT		TMR8_CH1C		SPI1_SCK I2S1_CK		
PA6		TMR1_BRK	TMR3_CH1	TMR8_BRK		SPI1_MISO	I2S2_MCK	USART3_CTS
PA7		TMR1_CH1C	TMR3_CH2	TMR8_CH1C		SPI1_MOSI I2S1_SDEXT		
PA8	CLKOUT1	TMR1_CH1			I2C3_SCL			USART1_CK
PA9		TMR1_CH2			I2C3_SMBA	SPI2_SCK I2S2_CK		USART1_TX
PA10		TMR1_CH3				SPI2_MOSI I2S2_SDEXT	I2S4_MCK	USART1_RX
PA11		TMR1_CH4			I2C2_SCL	SPI2_CS I2S2_WS	SPI4_MISO	USART1_CTS
PA12		TMR1_EXT			I2C2_SDA	SPI2_MISO		USART1_RTS_DE
PA13	JTMS SWDIO	IR_OUT					SPI3_MISO	
PA14	JTCK SWCLK						SPI3_MOSI I2S3_SDEXT	
PA15	JTDI	TMR2_CH1 TMR2_EXT				SPI1_CS I2S1_WS	SPI3_CS I2S3_WS	USART1_TX

引脚名	MUX8	MUX9	MUX10	MUX11	MUX12	MUX13	MUX14	MUX15
PA0	UART4_TX			EMAC_MII_CRS				EVENTOUT
PA1	UART4_RX	QSPI1_IO3		EMAC_MII_RX_CLK EMAC_RMII_REF_CLK				EVENTOUT
PA2			SDIO2_CK	EMAC_MDIO			XMC_D4	EVENTOUT
PA3		QSPI2_IO3	SDIO2_CMD	EMAC_MII_COL			XMC_D5	EVENTOUT

PA4	USART6_TX		SDIO2_D4	SDIO2_D0	OTG2_SOF	DVP_HSYNC	XMC_D6	EVENTOUT
PA5	USART6_RX	QSPI2_IO2	SDIO2_D5	SDIO2_D1			XMC_D7	EVENTOUT
PA6		TMR13_CH1	QSPI1_IO0	SDIO2_D2	SDIO1_CMD	DVP_PIXCLK	SDIO2_D6	EVENTOUT
PA7		TMR14_CH1	QSPI1_IO1	EMAC_MII_RX_DV EMAC_RMII_CRS_DV	XMC_SDNWE	SDIO2_D3	SDIO2_D7	EVENTOUT
PA8	USART2_TX		OTG1_SOF		SDIO1_D1		XMC_A4	EVENTOUT
PA9	I2C1_SCL		OTG1_VBUS		SDIO1_D2	DVP_D0		EVENTOUT
PA10	I2C1_SDA		OTG1_ID			DVP_D1		EVENTOUT
PA11	USART6_TX	CAN1_RX	OTG1_D-			DVP_D2		EVENTOUT
PA12	USART6_RX	CAN1_TX	OTG1_D+			DVP_D3		EVENTOUT
PA13			OTG1_OE					EVENTOUT
PA14	USART2_TX							EVENTOUT
PA15	USART2_RX	QSPI2_IO1	QSPI1_IO2		XMC_NE2 XMC_NCE3			EVENTOUT

表 7. 通过 GPIOB_AFR 寄存器配置端口 B 的复用功能

引脚名	MUX0	MUX1	MUX2	MUX3	MUX4	MUX5	MUX6	MUX7
PB0		TMR1_CH2C	TMR3_CH3	TMR8_CH2C		I2S1_MCK	USART2_RX	SPI3_MOSI I2S3_SDEXT
PB1		TMR1_CH3C	TMR3_CH4	TMR8_CH3C			SPI2_SCK I2S2_CK	
PB2		TMR2_CH4	TMR20_CH1		I2C3_SMBA			SPI3_MOSI I2S3_SDEXT
PB3	JTDO SWO	TMR2_CH2			I2C2_SDA	SPI1_SCK I2S1_CK	SPI3_SCK I2S3_CK	USART1_RX
PB4	JNTRST		TMR3_CH1		I2C3_SDA	SPI1_MISO	SPI3_MISO	I2S3_SDEXT
PB5			TMR3_CH2		I2C1_SMBA	SPI1_MOSI I2S1_SDEXT	SPI3_MOSI I2S3_SDEXT	USART1_CK
PB6			TMR4_CH1		I2C1_SCL	I2S1_MCK	SPI4_CS I2S4_WS	USART1_TX
PB7			TMR4_CH2	TMR8_BRK	I2C1_SDA		SPI4_SCK I2S4_CK	USART1_RX
PB8		TMR2_CH1 TMR2_EXT	TMR4_CH3	TMR10_CH1	I2C1_SCL		SPI4_MISO	
PB9	IR_OUT	TMR2_CH2	TMR4_CH4	TMR11_CH1	I2C1_SDA	SPI2_CS I2S2_WS	SPI4_MOSI I2S4_SDEXT	I2C2_SDA
PB10		TMR2_CH3			I2C2_SCL	SPI2_SCK I2S2_CK	I2S3_MCK	USART3_TX
PB11		TMR2_CH4	TMR5_CH4		I2C2_SDA			USART3_RX
PB12		TMR1_BRK	TMR5_CH1		I2C2_SMBA	SPI2_CS I2S2_WS	SPI4_CS I2S4_WS	SPI3_SCK I2S3_CK
PB13		TMR1_CH1C			I2C3_SMBA	SPI2_SCK I2S2_CK	SPI4_SCK I2S4_CK	I2C3_SCL
PB14		TMR1_CH2C		TMR8_CH2C	I2C3_SDA	SPI2_MISO	I2S2_SDEXT	USART3_RTS_DE
PB15	ERTC_REFIN	TMR1_CH3C		TMR8_CH3C	I2C3_SCL	SPI2_MOSI I2S2_SDEXT		

引脚名	MUX8	MUX9	MUX10	MUX11	MUX12	MUX13	MUX14	MUX15

PB0	USART3_CK	QSPI2_IO0	QSPI1_IO0	EMAC_MII_RXD2	SDIO1_D1			EVENTOUT
PB1	USART3_RTS_DE	QSPI1_SCK	QSPI2_SCK	EMAC_MII_RXD3	SDIO1_D2			EVENTOUT
PB2		QSPI1_SCK			SDIO1_CK			EVENTOUT
PB3	UART7_RX		QSPI1_IO3			DVP_D4		EVENTOUT
PB4	UART7_TX				SDIO1_D0	DVP_D5		EVENTOUT
PB5	UART5_RX	CAN2_RX		EMAC_PPS_OUT	XMC_SDCKE1	DVP_D10	SDIO1_D3	EVENTOUT
PB6	UART5_TX	CAN2_TX	QSPI1_CS		XMC_SDCS1	DVP_D5	SDIO1_D0	EVENTOUT
PB7		QSPI2_IO1			XMC_NADV	DVP_VSYNC	SDIO1_D0	EVENTOUT
PB8	UART5_RX	CAN1_RX	QSPI2_CS	EMAC_MII_TXD3	SDIO1_D4	DVP_D6		EVENTOUT
PB9	UART5_TX	CAN1_TX	QSPI1_CS		SDIO1_D5	DVP_D7		EVENTOUT
PB10		QSPI1_CS	QSPI1_IO1	EMAC_MII_RX_ER	SDIO1_D7		XMC_NOE	EVENTOUT
PB11			QSPI1_IO0	EMAC_MII_TX_EN EMAC_RMII_TX_EN				EVENTOUT
PB12	USART3_CK	CAN2_RX		EMAC_MII_TXD0 EMAC_RMII_TXD0	OTG2_ID		XMC_D13	EVENTOUT
PB13	USART3_CTS	CAN2_TX		EMAC_MII_TXD1 EMAC_RMII_TXD1	OTG2_VBUS			EVENTOUT
PB14		TMR12_CH1			OTG2_D-	SDIO1_D6	XMC_D0	EVENTOUT
PB15		TMR12_CH2			OTG2_D+	SDIO1_CK		EVENTOUT

表 8. 通过 GPIOF_AFR 寄存器配置端口 C 的复用功能

引脚名	MUX0	MUX1	MUX2	MUX3	MUX4	MUX5	MUX6	MUX7
PC0					I2C3_SCL			
PC1					I2C3_SDA	SPI3_MOSI I2S3_SDEXT		SPI2_MOSI I2S2_SDEXT
PC2			TMR20_CH2			SPI2_MISO	I2S2_SDEXT	
PC3						SPI2_MOSI I2S2_SDEXT		
PC4				TMR9_CH1		I2S1_MCK		USART3_TX
PC5				TMR9_CH2	I2C1_SMBA			USART3_RX
PC6			TMR3_CH1	TMR8_CH1	I2C1_SCL	I2S2_MCK		
PC7			TMR3_CH2	TMR8_CH2	I2C1_SDA	SPI2_SCK I2S2_CK	I2S3_MCK	
PC8			TMR3_CH3	TMR8_CH3		I2S4_MCK	TMR20_CH3	UART8_TX
PC9	CLKOUT2		TMR3_CH4	TMR8_CH4	I2C3_SDA			UART8_RX
PC10			TMR5_CH2				SPI3_SCK I2S3_CK	USART3_TX
PC11			TMR5_CH3			I2S3_SDEXT	SPI3_MISO	USART3_RX
PC12				TMR11_CH1	I2C2_SDA		SPI3_MOSI I2S3_SDEXT	USART3_CK
PC13								

PC14								
PC15								

引脚名	MUX8	MUX9	MUX10	MUX11	MUX12	MUX13	MUX14	MUX15
PC0	UART7_TX		SDIO2_D0		XMC_SDNWE			EVENTOUT
PC1	UART7_RX		SDIO2_D1	EMAC_MDC				EVENTOUT
PC2	UART8_TX		SDIO2_D2	EMAC_MII_TXD2	XMC_SDCS0		XMC_NWE	EVENTOUT
PC3	UART8_RX	QSPI2_IO1	SDIO2_D3	EMAC_MII_TX_CLK	XMC_SDCKE0		XMC_A0	EVENTOUT
PC4			QSPI1_IO2	EMAC_MII_RXD0 EMAC_RMII_RXD0	XMC_SDCS0	SDIO2_CK	XMC_NE4	EVENTOUT
PC5			QSPI1_IO3	EMAC_MII_RXD1 EMAC_RMII_RXD1	XMC_SDCKE0	SDIO2_CMD	XMC_NOE	EVENTOUT
PC6	USART6_TX		XMC_A0		SDIO1_D6	DVP_D0	XMC_D1	EVENTOUT
PC7	USART6_RX		XMC_A1		SDIO1_D7	DVP_D1		EVENTOUT
PC8	USART6_CK	QSPI1_IO2	XMC_A2		SDIO1_D0	DVP_D2		EVENTOUT
PC9		QSPI1_IO0	XMC_A3	OTG2_OE	SDIO1_D1	DVP_D3		EVENTOUT
PC10	UART4_TX	QSPI1_IO1			SDIO1_D2	DVP_D8		EVENTOUT
PC11	UART4_RX	QSPI1_CS			SDIO1_D3	DVP_D4	XMC_D2	EVENTOUT
PC12	UART5_TX				SDIO1_CK	DVP_D9	XMC_D3	EVENTOUT
PC13								EVENTOUT
PC14								EVENTOUT
PC15								EVENTOUT

表 9. 通过 GPIOF_AFR 寄存器配置端口 D 的复用功能

引脚名	MUX0	MUX1	MUX2	MUX3	MUX4	MUX5	MUX6	MUX7
PD0						SPI4_MISO	SPI3_MOSI I2S3_SDEXT	SPI2_CS I2S2_WS
PD1							SPI2_SCK I2S2_CK	SPI2_CS I2S2_WS
PD2			TMR3_EXT					USART3_RTS_DE
PD3						SPI2_SCK I2S2_CK	SPI2_MISO	USART2_CTS
PD4							SPI2_MOSI I2S2_SDEXT	USART2_RTS_DE
PD5								USART2_TX
PD6						SPI3_MOSI I2S3_SDEXT		USART2_RX
PD7								USART2_CK
PD8								USART3_TX
PD9								USART3_RX

PD10								USART3_CK
PD11					I2C2_SMBA			USART3_CTS
PD12			TMR4_CH1		I2C2_SCL			USART3_RTS_DE
PD13			TMR4_CH2		I2C2_SDA			
PD14			TMR4_CH3		I2C3_SCL			
PD15			TMR4_CH4		I2C3_SDA			

引脚名	MUX8	MUX9	MUX10	MUX11	MUX12	MUX13	MUX14	MUX15
PD0		CAN1_RX	XMC_A5		XMC_D2			EVENTOUT
PD1		CAN1_TX	XMC_A6		XMC_D3			EVENTOUT
PD2	UART5_RX		XMC_A7		SDIO1_CMD	DVP_D11	XMC_NWE	EVENTOUT
PD3		QSPI1_SCK	XMC_A8		XMC_CLK	DVP_D5		EVENTOUT
PD4			XMC_A9		XMC_NOE			EVENTOUT
PD5			XMC_A10		XMC_NWE			EVENTOUT
PD6			XMC_A11		XMC_NWAIT	DVP_D10		EVENTOUT
PD7			XMC_A12		XMC_NE1 XMC_NCE2			EVENTOUT
PD8				EMAC_MII_RX_DV EMAC_RMII_CRS_DV	XMC_D13			EVENTOUT
PD9				EMAC_MII_RXD0 EMAC_RMII_RXD0	XMC_D14			EVENTOUT
PD10				EMAC_MII_RXD1 EMAC_RMII_RXD1	XMC_D15			EVENTOUT
PD11		QSPI1_IO0	XMC_A14 XMC_SDBA0	EMAC_MII_RXD2	XMC_A16 XMC CLE			EVENTOUT
PD12		QSPI1_IO1	XMC_A15 XMC_SDBA1	EMAC_MII_RXD3	XMC_A17 XMC ALE			EVENTOUT
PD13	UART8_TX	QSPI1_IO3	XMC_SD_CLK		XMC_A18			EVENTOUT
PD14	UART8_RX				XMC_D0			EVENTOUT
PD15					XMC_D1			EVENTOUT

表 10. 通过 GPIOF_AFR 寄存器配置端口 E 的复用功能

引脚名	MUX0	MUX1	MUX2	MUX3	MUX4	MUX5	MUX6	MUX7
PE0			TMR4_EXT				TMR20_EXT	
PE1		TMR1_CH2C					TMR20_CH4	
PE2			TMR3_EXT			SPI4_SCK I2S4_CK	TMR20_CH1	
PE3			TMR3_CH1				TMR20_CH2	
PE4	CLKOUT1		TMR3_CH2			SPI4_CS I2S4_WS	TMR20_CH1C	
PE5			TMR3_CH3	TMR9_CH1		SPI4_MISO	TMR20_CH2C	

PE6			TMR3_CH4	TMR9_CH2		SPI4_MOSI I2S4_SDEXT	TMR20_CH3C	
PE7		TMR1_EXT						
PE8		TMR1_CH1C						UART4_TX
PE9		TMR1_CH1						UART4_RX
PE10		TMR1_CH2C						
PE11		TMR1_CH2				SPI4_CS I2S4_WS		
PE12		TMR1_CH3C			SPI1_CS I2S1_WS	SPI4_SCK I2S4_CK		
PE13		TMR1_CH3			SPI1_SCK I2S1_CK	SPI4_MISO		
PE14		TMR1_CH4			SPI1_MISO	SPI4_MOSI I2S4_SDEXT		
PE15		TMR1_BRK			SPI1_MOSI I2S1_SDEXT			

引脚名	MUX8	MUX9	MUX10	MUX11	MUX12	MUX13	MUX14	MUX15
PE0	UART8_RX				XMC_LB/XMC_SDDQML	DVP_D2		EVENTOUT
PE1	UART8_TX				XMC_UB/XMC_SDDQMH	DVP_D3		EVENTOUT
PE2		QSPI1_IO2	XMC_SDNCAS	EMAC_MII_TXD3	XMC_A23			EVENTOUT
PE3					XMC_A19	DVP_D9		EVENTOUT
PE4					XMC_A20	DVP_D4		EVENTOUT
PE5					XMC_A21	DVP_D6		EVENTOUT
PE6			XMC_SDNRAS		XMC_A22	DVP_D7		EVENTOUT
PE7	UART7_RX		QSPI2_IO0		XMC_D4			EVENTOUT
PE8	UART7_TX		QSPI2_IO1		XMC_D5			EVENTOUT
PE9			QSPI2_IO2		XMC_D6			EVENTOUT
PE10	UART5_TX		QSPI2_IO3		XMC_D7			EVENTOUT
PE11	UART5_RX				XMC_D8			EVENTOUT
PE12					XMC_D9			EVENTOUT
PE13					XMC_D10			EVENTOUT
PE14					XMC_D11			EVENTOUT
PE15					XMC_D12			EVENTOUT

表 11. 通过 GPIOF_AFR 寄存器配置端口 F 的复用功能

引脚名	MUX0	MUX1	MUX2	MUX3	MUX4	MUX5	MUX6	MUX7
PF0					I2C2_SDA			
PF1					I2C2_SCL			

PF2			TMR20_CH3		I2C2_SMBA			
PF3			TMR20_CH4					
PF4			TMR20_CH1C					
PF5			TMR20_CH2C					
PF6			TMR20_CH4	TMR10_CH1				
PF7			TMR20_BRK	TMR11_CH1				
PF8								
PF9			TMR20_BRK					
PF10		TMR1_EXT	TMR5_CH4					
PF11			TMR20_EXT	TMR8_EXT				
PF12			TMR20_CH1	TMR8_BRK				
PF13			TMR20_CH2		I2C3_SMBA			
PF14			TMR20_CH3		I2C3_SCL			
PF15			TMR20_CH4		I2C3_SDA			

引脚名	MUX8	MUX9	MUX10	MUX11	MUX12	MUX13	MUX14	MUX15
PF0					XMC_A0			EVENTOUT
PF1					XMC_A1			EVENTOUT
PF2					XMC_A2			EVENTOUT
PF3					XMC_A3			EVENTOUT
PF4					XMC_A4			EVENTOUT
PF5					XMC_A5			EVENTOUT
PF6	UART7_RX	QSPI1_IO3			XMC_NIORD			EVENTOUT
PF7	UART7_TX	QSPI1_IO2			XMC_NREG			EVENTOUT
PF8		TMR13_CH1	QSPI1_IO0		XMC_NIOWR			EVENTOUT
PF9		TMR14_CH1	QSPI1_MOSI_IO1		XMC_CD			EVENTOUT
PF10		QSPI1_SCK			XMC_INTR	DVP_D11		EVENTOUT
PF11					XMC_SDNRAS	DVP_D12		EVENTOUT
PF12					XMC_A6			EVENTOUT
PF13					XMC_A7			EVENTOUT
PF14					XMC_A8			EVENTOUT
PF15					XMC_A9			EVENTOUT

表 12. 通过 GPIOF_AFR 寄存器配置端口 G 的复用功能

引脚名	MUX0	MUX1	MUX2	MUX3	MUX4	MUX5	MUX6	MUX7
PG0			TMR20_CH1C			SPI1_MISO		
PG1			TMR20_CH2C			SPI1_MOSI I2S1_SDEXT		
PG2			TMR20_CH3C					
PG3			TMR20_BRK					
PG4								
PG5			TMR20_EXT					
PG6								
PG7								
PG8						QSPI2_CS		
PG9								
PG10						QSPI2_IO2		
PG11						QSPI2_IO3	SPI4_SCK I2S4_CK	
PG12						QSPI2_IO1	SPI4_MISO	
PG13						QSPI2_SCK	SPI4_MOSI I2S4_SDEXT	
PG14						QSPI2_IO0	SPI4_CS I2S4_WS	
PG15								

引脚名	MUX8	MUX9	MUX10	MUX11	MUX12	MUX13	MUX14	MUX15
PG0		CAN1_RX			XMC_A10			EVENTOUT
PG1		CAN1_TX			XMC_A11			EVENTOUT
PG2					XMC_A12			EVENTOUT
PG3					XMC_A13			EVENTOUT
PG4					XMC_A14 XMC_SDBA0			EVENTOUT
PG5					XMC_A15 XMC_SDBA1			EVENTOUT
PG6			QSPI1_CS		XMC_INT2	DVP_D12		EVENTOUT
PG7	USART6_CK				XMC_INT3	DVP_D13		EVENTOUT
PG8	USART6_RTS_DE			EMAC_PPS_OUT	XMC_SDCLK			EVENTOUT
PG9	USART6_RX	QSPI1_IO 2			XMC_NE2 XMC_NCE3	DVP_VSY NC		EVENTOUT
PG10					XMC_NE3 XMC_NCE4_1	DVP_D2		EVENTOUT
PG11		CAN2_RX		EMAC_MII_TX_EN EMAC_RMII_TX_E N	XMC_NCE4_2	DVP_D3		EVENTOUT
PG12	USART6_RTS_DE	CAN2_TX			XMC_NE4			EVENTOUT

PG13	USART6_CTS			EMAC_MII_TXD0 EMAC_RMII_TXD0	XMC_A24			EVENTOUT
PG14	USART6_TX	QSPI1_IO3		EMAC_MII_TXD1 EMAC_RMII_TXD1	XMC_A25			EVENTOUT
PG15	USART6_CTS				XMC_SDNCAS	DVP_D13		EVENTOUT

表 13. 通过 GPIOF_AFR 寄存器配置端口 H 的复用功能

引脚名	MUX0	MUX1	MUX2	MUX3	MUX4	MUX5	MUX6	MUX7
PH0					I2C1_SDA			
PH1					I2C1_SCL			
PH2			TMR5_CH1		I2C2_SCL			
PH3			TMR5_CH2		I2C2_SDA			

引脚名	MUX8	MUX9	MUX10	MUX11	MUX12	MUX13	MUX14	MUX15
PH0								EVENTOUT
PH1								EVENTOUT
PH2	UART4_RX		QSPI1_IO0					EVENTOUT
PH3	UART4_TX		QSPI1_IO1					EVENTOUT

3.2 特殊 I/O

3.2.1 调试复用引脚

- 在复位时，和复位后不像其他GPIO一样处于浮空输入状态，而是处于AF模式
- PA13：JTMS/SWDIO，AF上拉
- PA14：JTCK/SWCLK，AF下拉
- PA15：JTDI，AF上拉
- PB3：JTDO/SWO，AF浮空
- PB4：JNTRST，AF上拉

3.2.2 振荡器复用引脚

- 振荡器关闭的状态下（复位后的默认状态），相关引脚可用作GPIO
- 振荡器使能状态下，相应引脚的GPIO配置无效
- 振荡器处于bypass模式（使用外部时钟源）时，HEXT_IN/LEXT_IN为振荡器时钟输入引脚，HEXT_OUT/LEXT_OUT可做GPIO使用

3.2.3 电池供电域引脚

- 电池供电域引脚包括PC13、PC14以及PC15。电池供电域由VDD或VBAT引脚供电，当VDD主电源被切断时，电池供电域自动切换至VBAT引脚供电，以保障ERTC正常工作。
- 当电池供电域由VDD供电时，PC13可以作为通用I/O口、TAMPER引脚、ERTC校准时钟、ERTC闹钟或秒输出，PC14和PC15可以用于GPIO或LEXT引脚。（PC13至PC15作为I/O口的速度必须限制在2MHz以下，最大负载为30pF，而且这些I/O口绝对不能当作电流源）。
- 当电池供电域由VBAT供电时，PC13可以作为TAMPER引脚、ERTC闹钟或秒输出，PC14和PC15只能用于LEXT引脚。

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：

- GPIO寄存器复位
- 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

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;      /* 引脚选择 */
gpio_init_struct.gpio_pull = GPIO_PULL_NONE;  /* 可选无、上拉或下拉 */
gpio_init_struct.gpio_mode = GPIO_MODE_OUTPUT; /* 选择输出模式 */
gpio_init_struct.gpio_out_type  = GPIO_OUTPUT_PUSH_PULL; /* 推挽或开漏 */
gpio_init_struct.gpio_drive_strength = GPIO_DRIVE_STRENGTH_STRONGER; /* 驱动力 */
gpio_init(GPIOX, &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_pull = GPIO_PULL_NONE;
gpio_init_struct.gpio_mode = GPIO_MODE_INPUT; /* 选择输入模式 */
gpio_init_struct.gpio_out_type  = GPIO_OUTPUT_PUSH_PULL;
gpio_init_struct.gpio_drive_strength = GPIO_DRIVE_STRENGTH_STRONGER;
gpio_init(GPIOX, &gpio_init_struct);
```

4.3 模拟模式

当需要使用ADC或COMP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_pull = GPIO_PULL_NONE;
gpio_init_struct.gpio_mode = GPIO_MODE_ANALOG; /* 选择模拟模式 */
gpio_init_struct.gpio_out_type  = GPIO_OUTPUT_PUSH_PULL;
gpio_init_struct.gpio_drive_strength = GPIO_DRIVE_STRENGTH_STRONGER;
gpio_init(GPIOX, &gpio_init_struct);
```

4.4 复用模式

1. 不论使用何种外设模式，都必须将 I/O 配置为复用功能，之后系统才能正确使用 I/O（输入或输出）。
2. I/O 引脚通过复用器连接到相应的外设，该复用器一次只允许一个外设的复用功能 (IOMUX)连接到 I/O 引脚。这样便可确保共用同一个 I/O 引脚的外设之间不会发生冲突。每个 I/O引脚都有一个复用器，该复用器具有16路复用功能输入/输出（MUX0到MUX15），可通过 gpio_pin_mux_config () 函数对这些引脚进行配置：
   - 复位后，所有 I/O 都会连接到系统的复用功能 0 (MUX_0)
   - 通过配置 MUX0 到 MUX15 可以映射外设的复用功能
3. 除了这种灵活的 I/O 复用架构之外，各外设还具有映射到不同 I/O 引脚的复用功能，这可以针对不同器件封装优化外设 I/O 功能的数量；例如，可将 USART2_TX 引脚映射到PA2 或 PA14 引脚上。
4. 配置过程：
   - 使用 gpio_pin_mux_config() 函数将引脚连接到所需的外设复用功能，例如配置PA0作为TMR2_EXT输入

     ```
     gpio_pin_mux_config(GPIOA, GPIO_PINS_SOURCE0, GPIO_MUX_1);
     ```

   - 使用 gpio_init() 函数配置 I/O 引脚：
     - 通过以下方式配置复用功能模式下的所需引脚

       ```
       gpio_init_struct.gpio_mode = GPIO_MODE_MUX;
       ```

     - 通过以下成员选择类型、上拉/下拉和驱动能力

       gpio_pull、gpio_out_type 和 gpio_drive_strength 成员

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_GPIOA_PERIPH_CLOCK, TRUE);

/* 将 PA9 连接到 USART1_Tx */
gpio_pin_mux_config(GPIOA, GPIO_PINS_SOURCE9, GPIO_MUX_7);
/* 将 PA10 连接到 USART1_Rx */
```

```
gpio_pin_mux_config(GPIOA, GPIO_PINS_SOURCE10, GPIO_MUX_7);
/* 将 USART2_Tx 和 USART2_Rx 配置为复用功能 */
gpio_init_struct.gpio_pins = GPIO_PINS_9 | GPIO_PINS_10;
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);
```

4.4.2 TMR I/O 复用模式配置

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

/* 将 PA6 连接到 TMR1_BRK */
gpio_pin_mux_config(GPIOA, GPIO_PINS_SOURCE6, GPIO_MUX_1);
/* 将 PA8 连接到 TMR1_CH1 */
gpio_pin_mux_config(GPIOA, GPIO_PINS_SOURCE8, GPIO_MUX_1);
/* 将 PA12 连接到 TMR1_EXT */
gpio_pin_mux_config(GPIOA, GPIO_PINS_SOURCE12, GPIO_MUX_1);

/* 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);

/* 将 PB6 连接到 I2C1_SCL */
gpio_pin_mux_config(GPIOB, GPIO_PINS_SOURCE6, GPIO_MUX_4);
/* 将 PB7 连接到 I2C1_SDA */
```

```
gpio_pin_mux_config(GPIOB, GPIO_PINS_SOURCE7, GPIO_MUX_4);
/* 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_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_OPEN_DRAIN;
gpio_init_struct.gpio_drive_strength = GPIO_DRIVE_STRENGTH_STRONGER;
gpio_init(GPIOB, &gpio_init_struct);
```

5 案例 LED 翻转

5.1 功能简介

通过系统时钟延时来对 LED 进行翻转。

5.2 资源准备

1) 硬件环境：

对应产品型号的 AT-START BOARD

2) 软件环境：

project\at_start_f437\examples\gpio\led_toggle

5.3 软件设计

1) 配置流程

- 配置系统时钟；
- 初始化延时函数和 LED；
- 翻转 LED。

2) 代码介绍

main 函数代码描述

```
int main(void)
{
  system_clock_config();                          /* 系统时钟配置，默认 288 MHz */

  at32_board_init();                              /* 初始化延时函数和 LED */

  while(1)
  {
    at32_led_toggle(LED2);                        /* 翻转 LED2 */

    delay_ms(200);                                /* 延时 200 ms */

    at32_led_toggle(LED3);                        /* 翻转 LED3 */
    delay_ms(200);
    at32_led_toggle(LED4);                        /* 翻转 LED4 */
    delay_ms(200);
  }
}
```

LED 翻转代码描述

```
void at32_led_toggle(led_type led)
{
  if(led > (LED_NUM - 1))                         /* 判断 LED 是否有效 */
    return;
  if(led_gpio_pin[led])
    led_gpio_port[led]->odt ^= led_gpio_pin[led];  /* 翻转 LED 对应的 GPIO */
}
```

5.4 实验效果

- 上电运行会看到LED2、LED3和LED4以间隔200ms时间交替的进行翻转。

6 案例 SWJTAG 接口复用

6.1 功能简介

对 SWJTAG 接口的 I/O 进行复用。

6.2 资源准备

1) 硬件环境：

对应产品型号的 AT-START BOARD

2) 软件环境：

project\at_start_f437\examples\gpio\swjtag_mux

6.3 软件设计

1) 配置流程

- 配置系统时钟；
- 初始化延时函数；
- 配置 SWJTAG 接口的复用和 USART2 初始化。

2) 代码介绍

main 函数代码描述

```
int main(void)
{

  system_clock_config();                                  /* 系统时钟配置，默认 288 MHz */

  at32_board_init();                                      /* 初始化延时函数 */

  swj_dp_config();                                        /* 配置 SWJTAG 接口的复用和 USART2 初始化 */

  while(1)
  {
    gpio_pins_toggle(GPIOA, GPIO_PINS_13);                /* 翻转 GPIO（PA13） */
    delay_ms(500);
    while(usart_flag_get(USART2, USART_TDBE_FLAG) == RESET);
    usart_data_transmit(USART2, data++);                  /* USART2_TX（PA14）发送数据 */
  }
}
```

SWJ 配置代码描述

```
void swj_dp_config(void)
{
  gpio_init_type gpio_init_struct;

  crm_periph_clock_enable(CRM_GPIOA_PERIPH_CLOCK, TRUE);
  crm_periph_clock_enable(CRM_USART2_PERIPH_CLOCK, TRUE);

  /* 配置 PA13 (jtms/swdio)作为通用 I/O 推挽输出 */
```

```
  gpio_init_struct.gpio_pins = GPIO_PINS_13;
  gpio_init_struct.gpio_mode = GPIO_MODE_OUTPUT;
  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);

  /* 配置 PA14 (jtck/swclk)作为 USART2_TX 复用 I/O 推挽输出 */
  gpio_init_struct.gpio_pins = GPIO_PINS_14;
  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);
  gpio_pin_mux_config(GPIOA, GPIO_PINS_SOURCE14, GPIO_MUX_8);

  /* USART2 参数初始化 */
  usart_init(USART2, 115200, USART_DATA_8BITS, USART_STOP_1_BIT);
  usart_transmitter_enable(USART2, TRUE);
  usart_enable(USART2, TRUE);
}
```

6.4 实验效果

- 将PA13接示波器，PA14接入串口打印工具；
- 程序运行过程中PA13每隔500ms会翻转一次，表示jtms/swdio引脚已被用为GPIO使用；
- PA14接入串口打印工具后，每隔500ms会看到USART2_TX打印主循环执行次数。

7 版本历史

表 14. 文档版本历史

日期	版本	变更
2020.9.30	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。