AN0069
应用笔记

AT32F421 GPIO 使用指南

前言

AT32F421xx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。AT32F421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：

- AT32F421_Firmware_Library_V2.x.x\project\at_start_f421\examples\gpio
- RM_AT32F421 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	AT32F421 系列

目录

表目录

图目录

1 GPIO 特性

- 最大封装（48pin）具有39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，它们被激活或断开有赖于GPIOx_PULL寄存器的值。

图 1. GPIO 基本结构

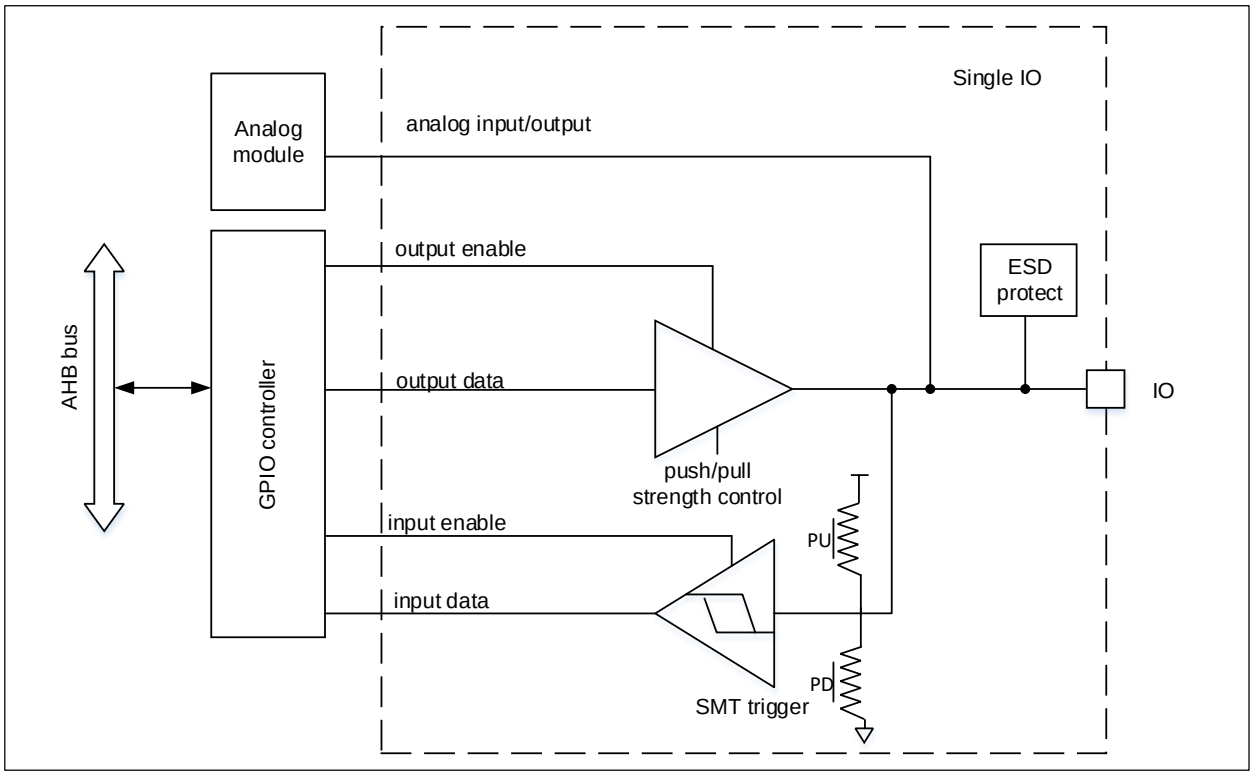


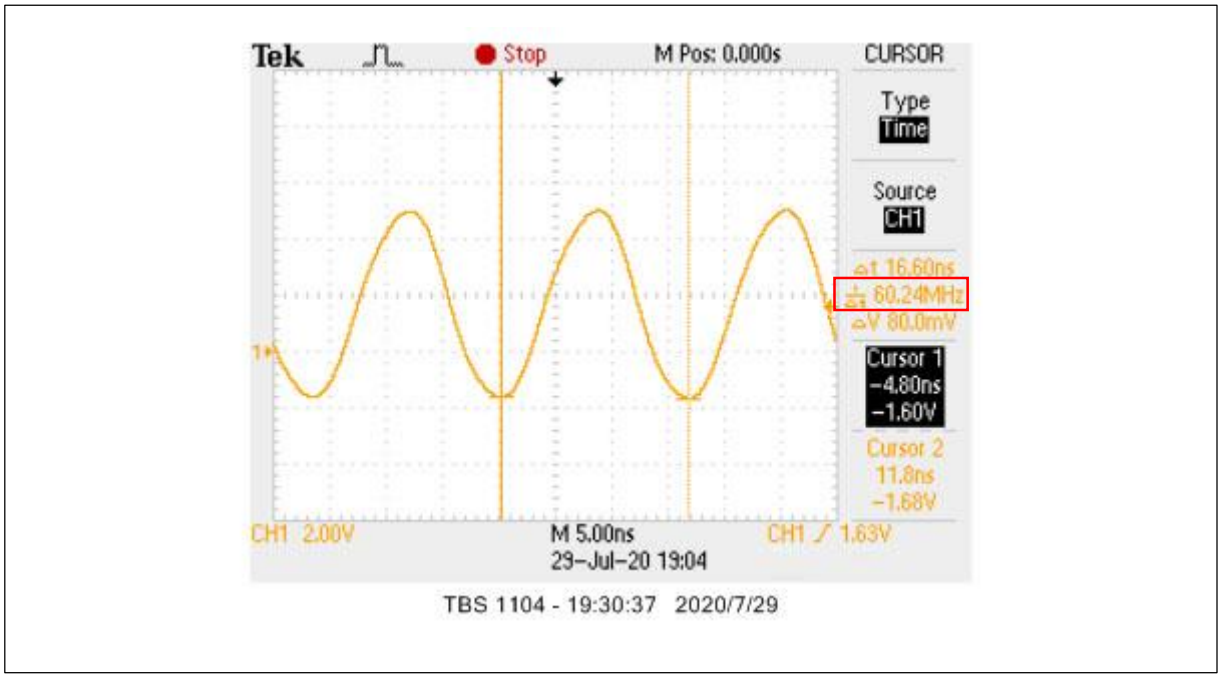
表 1. GPIO 配置表

CFGR[1：0]	OMODE	ODRVR[1：0]		PULL[1：0]		I/O 配置	
01	0	ODRV[1：0]		0	0	通用输出	PP
	0			0	1	通用输出	PP
	0			1	0	通用输出	PP
	0			1	1	保留	
	1			0	0	通用输出	OD
	1			0	1	通用输出	OD
	1			1	0	通用输出	OD
	1			1	1	保留(通用输出 OD)	
10	0			0	0	复用输出	PP
	0			0	1	复用输出	PP
	0			1	0	复用输出	PP
	0			1	1	保留	
	1			0	0	复用输出	OD
	1			0	1	复用输出	OD
	1			1	0	复用输出	OD
	1			1	1	保留	
00	x	x	x	0	0	输入	浮空
	x	x	x	0	1	输入	PU
	x	x	x	1	0	输入	PD
	x	x	x	1	1	保留(浮空输入)	
11	x	x	x	0	0	输入/输出	模拟
	x	x	x	0	1	保留	
	x	x	x	1	0		
	x	x	x	1	1		

2.1 GPIO toggle

AT32F421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60MHz：

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。

- PC14/PC15 (HEXT_IN/ OUT)
- PF0/PF1 (LEXT_IN/ OUT)

表 2. TC 引脚示例

引脚名称	引脚类型	IO电平	复用功能	附加功能
PF0/OSC_IN(PF0)	I/O	TC	I2C1_SDA	LEXT_IN

2.2.2 带模拟功能 5 V 容忍引脚（FTa）

比较器输入引脚以及ADC占用端口为带模拟功能5 V容忍引脚。

- PA0 – PA7
- PB0 – PB2，PB12 – PB15
- 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	引脚名称	IO结构	复用功能	附加功能
PA0	I/O	FTa	TMR1_EXT / USART2_CTS /I2C2_SCL / CMP_OUT	ADC_IN0 / CMP_NINV2 / CMP_INV6 / WKUP1

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	引脚名称	IO结构	复用功能	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TMR1_EXT / USART2_CTS /I2C2_SCL / CMP_OUT）
- 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 5. 通过 GPIOA_MUX*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		USART2_CTS			I2C2_SCL	TMR1_ETR		COMP_OUT
PA1	EVENTOUT	USART2_RTS			I2C2_SDA	TMR15_CH1N		
PA2	TMR15_CH1	USART2_TX						
PA3	TMR15_CH2	USART2_RX				I2S2_MCLK		
PA4	SPI1_NSS/I2S1_WS	USART2_CK			TMR14_CH1			
PA5	SPI1_SCK/I2S1_CK							
PA6	SPI1_MISO/I2S1_MCLK	TMR3_CH1	TMR1_BKIN	I2S2_MCLK		TMR16_CH1	EVENTOUT	COMP_OUT
PA7	SPI1_MOSI/I2S1_SD	TMR3_CH2	TMR1_CH1N		TMR14_CH1	TMR17_CH1	EVENTOUT	
PA8	CLKOUT	USART1_CK	TMR1_CH1	EVENTOUT	USART2_TX			I2C2_SCL
PA9	TMR15_BKIN	USART1_TX	TMR1_CH2		I2C1_SCL	CLKOUT		I2C2_SMBA
PA10	TMR17_BKIN	USART1_RX	TMR1_CH3		I2C1_SDA			
PA11	EVENTOUT	USART1_CTS	TMR1_CH4		I2C1_SMBA	I2C2_SCL		COMP_OUT
PA12	EVENTOUT	USART1_RTS	TMR1_ETR			I2C2_SDA		
PA13	SWDIO	IR_OUT					SPI2_MISO/I2S2_MCLK	
PA14	SWCLK	USART2_TX					SPI2_MOSI/I2S2_SD	
PA15	SPI1_NSS/I2S1_WS	USART2_RX		EVENTOUT			SPI2_NSS/I2S2_WS	

表 6. 通过 GPIOB_MUX*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	EVENTOUT	TMR3_CH3	TMR1_CH2N	USART2_RX			I2S1_MCLK	
PB1	TMR14_CH1	TMR3_CH4	TMR1_CH3N				SPI2_SCK/I2S2_CK	
PB2			TMR3_ETR					
PB3	SPI1_SCK/I2S1_CK	EVENTOUT					SPI2_SCK/I2S2_CK	
PB4	SPI1_MISO/I2S1_MCLK	TMR3_CH1	EVENTOUT			TMR17_BKIN	SPI2_MISO/I2S2_MCLK	I2C2_SDA
PB5	SPI1_MOSI/I2S1_SD	TMR3_CH2	TMR16_BKIN	I2C1_SMBA			SPI2_MOSI/I2S2_SD	
PB6	USART1_TX	I2C1_SCL	TMR16_CH1N				I2S1_MCLK	
PB7	USART1_RX	I2C1_SDA	TMR17_CH1N					
PB8		I2C1_SCL	TMR16_CH1					
PB9	IR_OUT	I2C1_SDA	TMR17_CH1	EVENTOUT		I2S1_MCLK		SPI2_NSS/I2S2_WS
PB10		I2C2_SCL						SPI2_SCK/I2S2_CK
PB11	EVENTOUT	I2C2_SDA						
PB12	SPI2_NSS/I2S2_WS	EVENTOUT	TMR1_BKIN			TMR15_BKIN		I2C2_SMBA
PB13	SPI2_SCK/I2S2_CK	-	TMR1_CH1N			I2C2_SCL		
PB14	SPI2_MISO/I2S2_MCLK	TMR15_CH1	TMR1_CH2N			I2C2_SDA		
PB15	SPI2_MOSI/I2S2_SD	TMR15_CH2	TMR1_CH3N	TMR15_CH1N				

表 7. 通过 GPIOF_MUX*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		I2C1_SDA						
PF1		I2C1_SCL						
PF6	I2C2_SCL							
PF7	I2C2_SDA							

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，而是处于复用模式
- PA13：SWDIO，复用上拉

- PA14：SWCLK，复用下拉

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，LEXT_IN/HEXT_IN为振荡器时钟输入引脚，LEXT_OUT/HEXT_OUT可做GPIO使用

3.2.3 电池供电域下的引脚

- 电池供电域下的引脚包括PC13、PC14以及PC15，电池供电域只能由VDD供电。
- 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。

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. 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 (MUX)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 (MUX0)
   - 通过配置 MUX1 到 MUX7 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 (MUX)，例如配置PA0作为TMR1_EXT输入

     ```
     gpio_pin_mux_config(GPIOA, GPIO_PINS_SOURCE0, GPIO_MUX_4);
     ```

   - 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/下拉和驱动力

       gpio_out_type、gpio_pull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_1);
/* 将 PA10 连接到 USART1_Rx */
gpio_pin_mux_config(GPIOA, GPIO_PINS_SOURCE10, GPIO_MUX_1);
```

```
/* 将 USART1_Tx 和 USART1_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_2);
/* 将 PA8 连接到 TMR1_CH1 */
gpio_pin_mux_config(GPIOA, GPIO_PINS_SOURCE8, GPIO_MUX_2);
/* 将 PA12 连接到 TMR1_EXT */
gpio_pin_mux_config(GPIOA, GPIO_PINS_SOURCE12, GPIO_MUX_2);

/* 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_1);
/* 将 PB7 连接到 I2C1_SDA */
gpio_pin_mux_config(GPIOB, GPIO_PINS_SOURCE7, GPIO_MUX_1);
```

```
/* 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 版本历史

表 8. 文档版本历史

日期	版本	变更
2021.11.29	2.0.0	最初版本
2023.03.23	2.0.1	3.2.3 章节内容修改 修改电池供电域部分描述

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。