FAQ0045

常见问题解答

TMR1输出PWM时互补通道引脚如何配置成SPI2功能

Questions： TMR1 通道 CH1,CH2,CH3 输出 PWM 时，CH1C,CH2C,CH3C 对应的引脚（PB13,PB14,PB15）如何配置成 SPI2_CS, SPI2_SCK, SPI2_MISO

Answer：

注意事项如下：

1. 初始化时需要先 TMR1 使能后，再使能 SPI2
2. 在客户使用过程中，开启 TMR1 输出使能位后（OEN=1），不能有再次关闭，再次开启的操作
3. 设置 OEN=1 后，通道互补极性 C1CP 位只能设置为 0，不能和 OEN 位同时为 1
4. 建议使用 CxEN=0 方式去关闭 PWM，CxEN=1 去开启 PWM

示例：

```
#define PWM_OFF     TMR1->cctrl &= (~(u16)(0x0111));
#define PWM_ON      TMR1->cctrl |= (u16)(0x0111);
void tmr1_init(void)
{
  gpio_init_type gpio_init_struct;

  /* gpioa clock enable */
  crm_periph_clock_enable(CRM_GPIOA_PERIPH_CLOCK, TRUE);

  /* tmr1 clock enable */
  crm_periph_clock_enable(CRM_TMR1_PERIPH_CLOCK, TRUE);
  crm_periph_clock_enable(CRM_IOMUX_PERIPH_CLOCK, TRUE);
  gpio_pin_remap_config(TMR1_MUX_01, TRUE);

  gpio_default_para_init(&gpio_init_struct);

  gpio_init_struct.gpio_pins = GPIO_PINS_8 | GPIO_PINS_9 | GPIO_PINS_10;
  gpio_init_struct.gpio_out_type = GPIO_OUTPUT_PUSH_PULL;
  gpio_init_struct.gpio_pull = GPIO_PULL_NONE;
  gpio_init_struct.gpio_mode = GPIO_MODE_MUX;
  gpio_init_struct.gpio_drive_strength = GPIO_DRIVE_STRENGTH_STRONGER;
  gpio_init(GPIOA, &gpio_init_struct);
```

```
  /* compute the prescaler value */
  prescaler_value = (uint16_t)(system_core_clock / 24000000) - 1;

  /* tmr1 time base configuration */
  tmr_base_init(TMR1, 665, prescaler_value);
  tmr_cnt_dir_set(TMR1, TMR_COUNT_TWO_WAY_3);
  tmr_clock_source_div_set(TMR1, TMR_CLOCK_DIV1);

  tmr_output_default_para_init(&tmr_oc_init_structure);
  tmr_oc_init_structure.oc_mode = TMR_OUTPUT_CONTROL_PWM_MODE_A;
  tmr_oc_init_structure.oc_idle_state = FALSE;
  tmr_oc_init_structure.oc_polarity = TMR_OUTPUT_ACTIVE_HIGH;
  tmr_oc_init_structure.oc_output_state = TRUE;
  tmr_output_channel_config(TMR1, TMR_SELECT_CHANNEL_1, &tmr_oc_init_structure);
  tmr_channel_value_set(TMR1, TMR_SELECT_CHANNEL_1, ccr1_val);
  tmr_output_channel_buffer_enable(TMR1, TMR_SELECT_CHANNEL_1, TRUE);

  tmr_output_channel_config(TMR1, TMR_SELECT_CHANNEL_2, &tmr_oc_init_structure);
  tmr_channel_value_set(TMR1, TMR_SELECT_CHANNEL_2, ccr2_val);
  tmr_output_channel_buffer_enable(TMR1, TMR_SELECT_CHANNEL_2, TRUE);

  tmr_output_channel_config(TMR1, TMR_SELECT_CHANNEL_3, &tmr_oc_init_structure);
  tmr_channel_value_set(TMR1, TMR_SELECT_CHANNEL_3, ccr3_val);
  tmr_output_channel_buffer_enable(TMR1, TMR_SELECT_CHANNEL_3, TRUE);

  tmr_period_buffer_enable(TMR1, TRUE);

  tmr_output_enable(TMR1, TRUE);

  /* tmr1 enable counter */
  tmr_counter_enable(TMR1, TRUE);

  PWM_OFF;
  PWM_ON;
}

void spi2_init(void)
{
```

```
gpio_init_type gpio_init_struct;
spi_init_type spi_init_struct;

/* gpiob clock enable */
crm_periph_clock_enable(CRM_GPIOB_PERIPH_CLOCK, TRUE);

crm_periph_clock_enable(CRM_SPI2_PERIPH_CLOCK, TRUE);

gpio_default_para_init(&gpio_init_struct);

/* spi2 sck pin */
gpio_init_struct.gpio_pull           = GPIO_PULL_DOWN;
gpio_init_struct.gpio_mode           = GPIO_MODE_INPUT;
gpio_init_struct.gpio_pins = GPIO_PINS_13;
gpio_init(GPIOB, &gpio_init_struct);

/* slave miso pin */
gpio_init_struct.gpio_pull           = GPIO_PULL_UP;
gpio_init_struct.gpio_mode           = GPIO_MODE_MUX;
gpio_init_struct.gpio_pins = GPIO_PINS_14;
gpio_init(GPIOB, &gpio_init_struct);

/* spi2 mosi pin */
gpio_init_struct.gpio_pull           = GPIO_PULL_UP;
gpio_init_struct.gpio_mode           = GPIO_MODE_INPUT;
gpio_init_struct.gpio_pins = GPIO_PINS_15;
gpio_init(GPIOB, &gpio_init_struct);

spi_default_para_init(&spi_init_struct);
spi_init_struct.transmission_mode = SPI_TRANSMIT_SIMPLEX_RX;
spi_init_struct.master_slave_mode = SPI_MODE_SLAVE;
spi_init_struct.mclk_freq_division = SPI_MCLK_DIV_8;
spi_init_struct.first_bit_transmission = SPI_FIRST_BIT_LSB;
spi_init_struct.frame_bit_num = SPI_FRAME_8BIT;
spi_init_struct.clock_polarity = SPI_CLOCK_POLARITY_LOW;
spi_init_struct.clock_phase = SPI_CLOCK_PHASE_2EDGE;
spi_init_struct.cs_mode_selection = SPI_CS_SOFTWARE_MODE;
spi_init(SPI2, &spi_init_struct);
```

```
    spi_enable(SPI2, TRUE);
}
```

类型： MCU 应用

适用型号： AT32F403，AT32F413，AT32F415，AT32F403A，AT32F407

主功能： TMR1，SPI2

次功能： 无

文档版本历史

日期	版本	变更
2022.2.16	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，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) 汽车应用或汽车环境；(D) 航天应用或航天环境，且/或(E) 武器。因雅特力产品不是为前述应用设计的，而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，风险由购买者单独承担，并且独力负责在此类相关使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。