
How to wake up AT32F421 from standby mode?

Questions:

How to wake up AT32F421 from Standby mode?

Answer:

AT32F421 series can be woken up from Standby mode through the rising edge on the WKUPx pin, the rising edge on the ERTC alarm event, external reset on NRST pin or IWDG reset.

1. For ERTC alarm event wakeup, please refer to the example code located under the *Project\AT_START_F421\Examples\PWR\STANDBY* directory.

```
int main(void)
{
    __IO uint32_t index = 0;
    /* config the system clock */
    system_clock_config();
    /* init at start board */
    at32_board_init();
    /* config priority group */
    nvic_priority_group_config(NVIC_PRIORITY_GROUP_4);
    /* turn on the led light */
    at32_led_off(LED2);
    at32_led_off(LED3);
    at32_led_off(LED4);
    /* enable pwc and bpr clock */
    crm_periph_clock_enable(CRM_PWC_PERIPH_CLOCK, TRUE);
    if(pwc_flag_get(PWC_STANDBY_FLAG) != RESET)
    {
        /* wakeup from standby */
        pwc_flag_clear(PWC_STANDBY_FLAG);
        at32_led_on(LED2);
    }
    if(pwc_flag_get(PWC_WAKEUP_FLAG) != RESET)
    {
        /* wakeup event occurs */
        pwc_flag_clear(PWC_WAKEUP_FLAG);
        at32_led_on(LED3);
    }
    delay_sec(1);
    /* config ertc */
    ertc_config();
}
```

```

at32_led_on(LED4);
ertc_alarm_config();
delay_sec(1);
ertc_alarm_value_set(3);
/* enter standby mode */
pwc_standby_mode_enter();
while(1)
{
}
}

```

- For IWDG wakeup event, please refer to demo located under the *Project\AT_START_F421\Examples\IWDG\IWDG_Standby* directory.

```

int main(void)
{
system_clock_config();
at32_board_init();
/* enable the pwc clock */
crm_periph_clock_enable(CRM_PWC_PERIPH_CLOCK, TRUE);
if(crm_flag_get(CRM_WDT_RESET_FLAG) != RESET)
{
/* reset from wdt */
crm_flag_clear(CRM_WDT_RESET_FLAG);
at32_led_on(LED4);
}
else
{
/* reset from other mode */
at32_led_off(LED4);
}
delay_ms(100);
/* disable register write protection */
wdt_register_write_enable(TRUE);
/* set the wdt divider value */
wdt_divider_set(WDT_CLK_DIV_4);
/* set reload value
timeout = reload_value * (divider / lick_freq )    (s)
lick_freq    = 40000 Hz
divider      = 4
reload_value = 3000
timeout = 3000 * (4 / 40000 ) = 0.3s = 300ms
*/
wdt_reload_value_set(3000 - 1);
/* enable wdt */
wdt_enable();
/* enter standby mode */

```

```
pwc_standby_mode_enter();
while(1)
{
}
}
```

3. AT32F421 series can be woken up through the following WKUP PINs:

WKUP1	PA0
WKUP2	PC13
WKUP6	PB5
WKUP7	PB15

For more information, refer to the demo under the [project/at_start_f421/examples/pwc/standby_wakeup_pin](#)

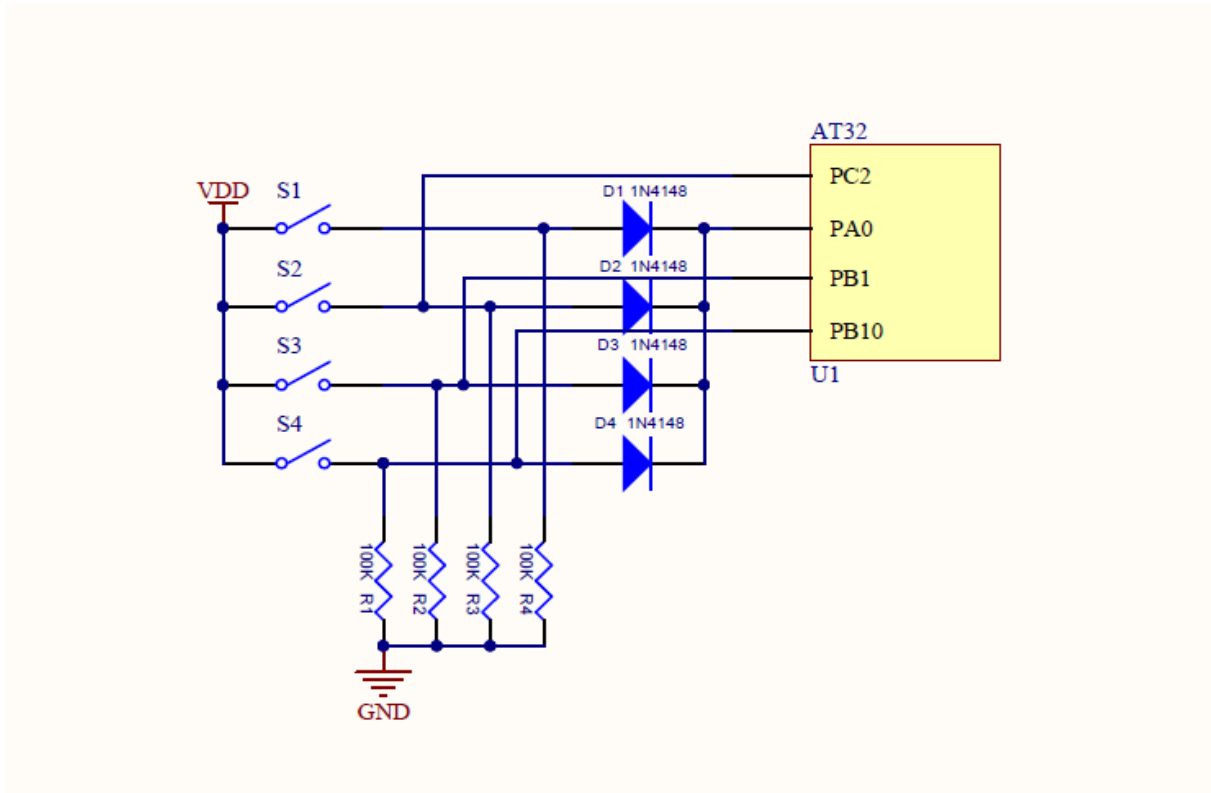
```
int main(void)
{
  __IO uint32_t index = 0;
  /* config the system clock */
  system_clock_config();
  /* init at start board */
  at32_board_init();
  /* config priority group */
  nvic_priority_group_config(NVIC_PRIORITY_GROUP_4);
  /* turn on the led light */
  at32_led_off(LED2);
  at32_led_off(LED3);
  at32_led_off(LED4);
  /* enable pwc clock */
  crm_periph_clock_enable(CRM_PWC_PERIPH_CLOCK, TRUE);
  if(pwc_flag_get(PWC_STANDBY_FLAG) != RESET)
  {
    /* wakeup from standby */
    pwc_flag_clear(PWC_STANDBY_FLAG);
    at32_led_on(LED2);
  }
  if(pwc_flag_get(PWC_WAKEUP_FLAG) != RESET)
  {
    /* wakeup event occurs */
    pwc_flag_clear(PWC_WAKEUP_FLAG);
    at32_led_on(LED3);
  }
  at32_led_on(LED4);
  for(index = 0; index < 0xFFFFFFFF; index++);
  /* enable wakeup pin1(pa0), pin2(pc13), pin6(pb5), pin7(pb15) */
  pwc_wakeup_pin_enable(PWC_WAKEUP_PIN_1 | PWC_WAKEUP_PIN_2 | PWC_WAKEUP_PIN_6 |
PWC_WAKEUP_PIN_7, TRUE);
  /* enter standby mode */
  pwc_standby_mode_enter();
}
```

```

while(1)
{
}

```

4. Additionally, AT32F421 can be woken through other pins, such as PC2, PB1 and PB10, with the circuit shown below:



Type: MCU

Applicable products: AT32F421

Main function: PWC

Minor function: WDT, GPIO

Document revision history

Date	Revision	Changes
2022.2.24	2.0.0	Initial release

IMPORTANT NOTICE – PLEASE READ CAREFULLY

Purchasers are solely responsible for the selection and use of ARTERY's products and services, and ARTERY assumes no liability whatsoever relating to the choice, selection or use of the ARTERY products and services described herein

No license, express or implied, to any intellectual property rights is granted under this document. If any part of this document deals with any third party products or services, it shall not be deemed a license granted by ARTERY for the use of such third party products or services, or any intellectual property contained therein, or considered as a warranty regarding the use in any manner of such third party products or services or any intellectual property contained therein.

Unless otherwise specified in ARTERY's terms and conditions of sale, ARTERY provides no warranties, express or implied, regarding the use and/or sale of ARTERY products, including but not limited to any implied warranties of merchantability, fitness for a particular purpose (and their equivalents under the laws of any jurisdiction), or infringement on any patent, copyright or other intellectual property right.

Purchasers hereby agree that ARTERY's products are not designed or authorized for use in: (A) any application with special requirements of safety such as life support and active implantable device, or system with functional safety requirements; (B) any aircraft application; (C) any aerospace application or environment; (D) any weapon application, and/or (E) or other uses where the failure of the device or product could result in personal injury, death, property damage. Purchasers' unauthorized use of them in the aforementioned applications, even if with a written notice, is solely at purchasers' risk, and Purchasers are solely responsible for meeting all legal and regulatory requirements in such use.

Resale of ARTERY products with provisions different from the statements and/or technical characteristics stated in this document shall immediately void any warranty grant by ARTERY for ARTERY's products or services described herein and shall not create or expand any liability of ARTERY in any manner whatsoever.

© 2022 Artery Technology -All rights reserved