



Ameba-Z Power Modes

This document introduces power modes of Ameba-Z.

REALTEK

Table of Contents

1. SUMMARY	3
2. POWER MODE.....	3
2.1. DEEP SLEEP.....	4
2.1.1. Power Domain	4
2.1.2. Wakeup Source	4
2.2. DEEP STANDBY	4
2.2.1. Power Domain	4
2.2.2. Wakeup Source	5
2.3. SLEEP.....	5
2.3.1. Power Domain	5
2.3.2. Wakeup Source	5
3. PULL CONTROL.....	5
4. POWER CONSUMPTION.....	6

REALTEK

1. Summary

Ameba supports three low power modes which are deep sleep mode, deep standby mode, and sleep mode. Deep sleep mode turn off more power domain than deep standby mode, and deep standby mode turn off more power domain than sleep mode. Various power modes can only switch back to run mode before change to other mode.

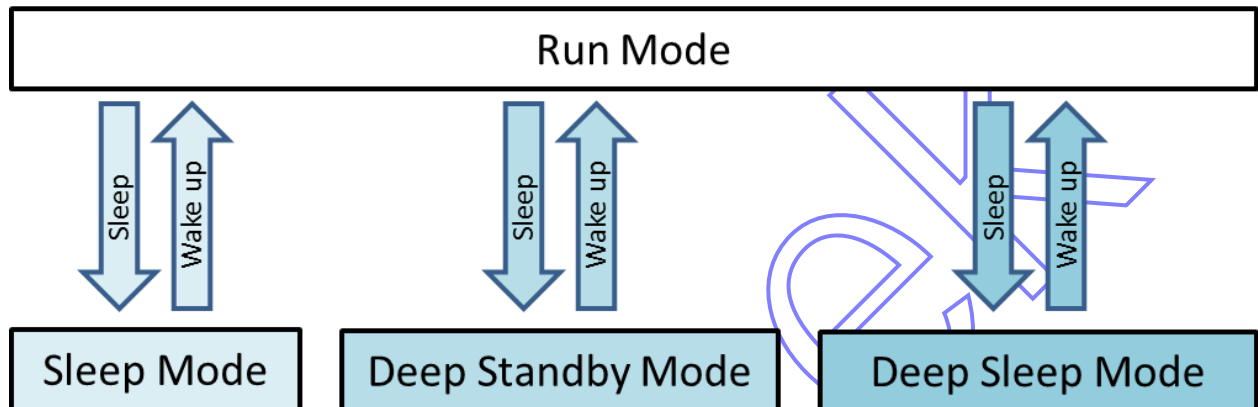


Figure 1 Ameba Power Mode

2. Power Mode

Table 1 Power domain comparison

	<i>CM4 core</i>	<i>System clock</i>	<i>Low Power Clock</i>	<i>SRAM</i>	<i>Register</i>	<i>Regulator</i>	<i>Main digital supply</i>	<i>Peripherals</i>	<i>RTC</i>
<i>Deep Sleep</i>	X	X	○	X	X	X	○	△	X
<i>Deep Standby</i>	X	X	○	X	X	X	○	△	○
<i>Sleep</i>	△	△	○	○	○	○	○	△	○
<i>Active</i>	○	○	○	○	○	○	○	○	○

2.1. Deep Sleep

2.1.1. Power Domain

<i>functions</i>	<i>Power State</i>	<i>comment</i>
<i>cortex-M4 core</i>	OFF	
<i>system clock</i>	OFF	
<i>SRAM</i>	OFF	
<i>Regulator</i>	OFF	
<i>Peripherals</i>	OFF	
<i>Backup register</i>	OFF	
<i>RTC</i>	OFF	
<i>low precision timer(1KHz)</i>	ON	Max. 140min
<i>Dsleep wake pin</i>	ON	4

2.1.2. Wakeup Source

<i>Wakeup source</i>	<i>Can wakeup</i>	<i>comment</i>
<i>low precision timer</i>	YES	
<i>Dsleep Wake pin</i>	YES	GPIOA_5 GPIOA_18 GPIOA_22 GPIOA_23

2.2. Deep Standby

2.2.1. Power Domain

<i>functions</i>	<i>Power State</i>	<i>comment</i>
<i>cortex-M4 core</i>	OFF	
<i>system clock</i>	OFF	
<i>SRAM</i>	OFF	
<i>Regulator</i>	OFF	
<i>Peripherals</i>	OFF	
<i>Backup register</i>	ON	16B
<i>RTC</i>	ON	
<i>System timer(250KHz)</i>	ON	Max. 8s
<i>low precision timer(1KHz)</i>	ON	Max. 140min
<i>wake pin</i>	ON	4

2.2.2. Wakeup Source

<i>Wakeup source</i>	<i>Can wakeup</i>	<i>comment</i>
<i>Wake pin</i>	YES	GPIOA_5 GPIOA_18 GPIOA_22 GPIOA_23
<i>RTC</i>	YES	
<i>System timer(250KHz)</i>	YES	Max. 8s
<i>low precision timer(1KHz)</i>	YES	Max. 140min

2.3. Sleep

2.3.1. Power Domain

Sleep mode turn off power domain including cortex-M3 core, and system clock. System is not required to restart after wakeup.

2.3.2. Wakeup Source

<i>Wakeup source</i>	<i>Can wakeup</i>	<i>comment</i>
<i>GPIO interrupt</i>	YES	High/Low active
<i>general purpose timer</i>	YES	
<i>wlan</i>	YES	
<i>ADC</i>	YES	
<i>UART</i>	YES	
<i>I2C</i>	YES	
<i>SDIO/GSPI</i>	YES	
<i>USB</i>	YES	
<i>Wake pin</i>	YES	GPIOA_5 GPIOA_18 GPIOA_22 GPIOA_23
<i>RTC</i>	YES	
<i>System timer(250KHz)</i>	YES	Max. 8s
<i>low precision timer(1KHz)</i>	YES	Max. 140min

3. Pull Control

It needs doing I/O pull control when enter deep sleep, deep standby, and sleep mode. Otherwise it result

power leakage. For example, UART voltage level is high. If we pull down uart pin or not pull, then power leakage happens. So **we need make sure each pin has proper pull control**.

In SDK you should set GPIO function pull control and sleep pull control in `pmap_func` (Ref: UM0120) based on you PCB board, and SDK will set pull control based on your setting between suspend and resume .

```
const PMAP_TypeDef pmap_func[] =
{
  // Pin Name      Func Select      Func PU/PD      Slp PU/PD      DrvStrenth
  {_PA_14, PINMUX_FUNCTION_SW, GPIO_PuPd_NOPULL, GPIO_PuPd_UP, PAD_DRV_STRENGTH_0}, //SWD_CLK
  {_PA_15, PINMUX_FUNCTION_SW, GPIO_PuPd_NOPULL, GPIO_PuPd_UP, PAD_DRV_STRENGTH_0}, //SWD_DATA
  {_PA_13, PINMUX_FUNCTION_PWM, GPIO_PuPd_NOPULL, GPIO_PuPd_UP, PAD_DRV_STRENGTH_0}, //PWM4
  {_PA_0, PINMUX_FUNCTION_PWM, GPIO_PuPd_NOPULL, GPIO_PuPd_DOWN, PAD_DRV_STRENGTH_0}, //PWM2
  {_PA_16, PINMUX_FUNCTION_PWM, GPIO_PuPd_NOPULL, GPIO_PuPd_UP, PAD_DRV_STRENGTH_0}, //PWM1
  {_PA_17, PINMUX_FUNCTION_PWM, GPIO_PuPd_NOPULL, GPIO_PuPd_UP, PAD_DRV_STRENGTH_0}, //PWM2
  {_PA_25, PINMUX_FUNCTION_UART, GPIO_PuPd_NOPULL, GPIO_PuPd_UP, PAD_DRV_STRENGTH_0}, //UART1_RXD
  {_PA_26, PINMUX_FUNCTION_UART, GPIO_PuPd_NOPULL, GPIO_PuPd_UP, PAD_DRV_STRENGTH_0}, //UART1_TXD
  {_PA_28, PINMUX_FUNCTION_I2C, GPIO_PuPd_UP, GPIO_PuPd_UP, PAD_DRV_STRENGTH_0}, //I2C1_SCL
  {_PA_27, PINMUX_FUNCTION_I2C, GPIO_PuPd_UP, GPIO_PuPd_UP, PAD_DRV_STRENGTH_0}, //I2C1_SDA
  {_PA_12, PINMUX_FUNCTION_PWM, GPIO_PuPd_NOPULL, GPIO_PuPd_UP, PAD_DRV_STRENGTH_0}, //PWM3
  {_PA_4, PINMUX_FUNCTION_UART, GPIO_PuPd_NOPULL, GPIO_PuPd_UP, PAD_DRV_STRENGTH_0}, //UART0_TXD
  {_PA_1, PINMUX_FUNCTION_UART, GPIO_PuPd_NOPULL, GPIO_PuPd_UP, PAD_DRV_STRENGTH_0}, //UART0_RXD
  {_PA_3, PINMUX_FUNCTION_UART, GPIO_PuPd_NOPULL, GPIO_PuPd_UP, PAD_DRV_STRENGTH_0}, //UART0_RTS
  {_PA_2, PINMUX_FUNCTION_UART, GPIO_PuPd_NOPULL, GPIO_PuPd_UP, PAD_DRV_STRENGTH_0}, //UART0_CTS
  {_PA_6, PINMUX_FUNCTION_SPIF, GPIO_PuPd_NOPULL, GPIO_PuPd_UP, PAD_DRV_STRENGTH_0}, //SPIC_CS
  {_PA_7, PINMUX_FUNCTION_SPIF, GPIO_PuPd_NOPULL, GPIO_PuPd_UP, PAD_DRV_STRENGTH_0}, //SPIC_DATA1
  {_PA_8, PINMUX_FUNCTION_SPIF, GPIO_PuPd_NOPULL, GPIO_PuPd_UP, PAD_DRV_STRENGTH_0}, //SPIC_DATA2
  {_PA_9, PINMUX_FUNCTION_SPIF, GPIO_PuPd_NOPULL, GPIO_PuPd_UP, PAD_DRV_STRENGTH_0}, //SPIC_DATA0
  {_PA_10, PINMUX_FUNCTION_SPIF, GPIO_PuPd_NOPULL, GPIO_PuPd_UP, PAD_DRV_STRENGTH_0}, //SPIC_CLK
  {_PA_11, PINMUX_FUNCTION_SPIF, GPIO_PuPd_NOPULL, GPIO_PuPd_UP, PAD_DRV_STRENGTH_0}, //SPIC_DATA3
  {_PA_5, PINMUX_FUNCTION_PWM, GPIO_PuPd_UP, GPIO_PuPd_UP, PAD_DRV_STRENGTH_0}, //PWM4
  {_PA_18, PINMUX_FUNCTION_SPIO, GPIO_PuPd_NOPULL, GPIO_PuPd_UP, PAD_DRV_STRENGTH_0}, //SD_D2
  {_PA_19, PINMUX_FUNCTION_SPIO, GPIO_PuPd_NOPULL, GPIO_PuPd_UP, PAD_DRV_STRENGTH_0}, //SD_D3
  {_PA_20, PINMUX_FUNCTION_SPIO, GPIO_PuPd_NOPULL, GPIO_PuPd_UP, PAD_DRV_STRENGTH_0}, //SD_CMD
  {_PA_21, PINMUX_FUNCTION_SPIO, GPIO_PuPd_NOPULL, GPIO_PuPd_UP, PAD_DRV_STRENGTH_0}, //SD_CLK
  {_PA_22, PINMUX_FUNCTION_SPIO, GPIO_PuPd_NOPULL, GPIO_PuPd_UP, PAD_DRV_STRENGTH_0}, //SD_D0
  {_PA_23, PINMUX_FUNCTION_SPIO, GPIO_PuPd_NOPULL, GPIO_PuPd_UP, PAD_DRV_STRENGTH_0}, //SD_D1
  {_PB_0, PINMUX_FUNCTION_SPIM, GPIO_PuPd_NOPULL, GPIO_PuPd_UP, PAD_DRV_STRENGTH_0}, //SPI1_CS
  {_PB_1, PINMUX_FUNCTION_SPIM, GPIO_PuPd_NOPULL, GPIO_PuPd_UP, PAD_DRV_STRENGTH_0}, //SPI1_CLK
  {_PB_2, PINMUX_FUNCTION_SPIM, GPIO_PuPd_NOPULL, GPIO_PuPd_UP, PAD_DRV_STRENGTH_0}, //SPI1_MISO
  {_PB_3, PINMUX_FUNCTION_SPIM, GPIO_PuPd_NOPULL, GPIO_PuPd_UP, PAD_DRV_STRENGTH_0}, //SPI1_MOSI
  {_PB_4, PINMUX_FUNCTION_I2S, GPIO_PuPd_NOPULL, GPIO_PuPd_UP, PAD_DRV_STRENGTH_0}, //I2S_MCK
  {_PB_5, PINMUX_FUNCTION_I2S, GPIO_PuPd_NOPULL, GPIO_PuPd_UP, PAD_DRV_STRENGTH_0}, //I2S_SD_TX
  {_PA_24, PINMUX_FUNCTION_I2S, GPIO_PuPd_NOPULL, GPIO_PuPd_UP, PAD_DRV_STRENGTH_0}, //I2S_SD_RX
  {_PA_31, PINMUX_FUNCTION_I2S, GPIO_PuPd_NOPULL, GPIO_PuPd_UP, PAD_DRV_STRENGTH_0}, //I2S_CLK
  {_PB_6, PINMUX_FUNCTION_I2S, GPIO_PuPd_NOPULL, GPIO_PuPd_UP, PAD_DRV_STRENGTH_0}, //I2S_IWS
  {_PA_30, PINMUX_FUNCTION_UART, GPIO_PuPd_UP, GPIO_PuPd_UP, PAD_DRV_STRENGTH_0}, //UART2_log_TXD
  {_PA_29, PINMUX_FUNCTION_UART, GPIO_PuPd_UP, GPIO_PuPd_UP, PAD_DRV_STRENGTH_0}, //UART2_log_RXD
  {_PNC, PINMUX_FUNCTION_GPIO, GPIO_PuPd_NOPULL, GPIO_PuPd_NOPULL, PAD_DRV_STRENGTH_0}, //table end
};
```

4. Power consumption

	SWR/LDO	QFN48 SWR	QFN32 LDO
Power Mode			
<i>Dsleep</i>		7.5uA	7.5uA
<i>Dstandby</i>		70uA	70uA
<i>Sleep</i>		120uA	120uA