# Chapter 14 Power Management Unit (PMU)

## 14.1 Overview

In order to meet high performance and low power requirements, a power management unit is designed for saving power when RK3128 in low power mode. The RK3128 PMU is dedicated for managing the power of the whole chip.

### 14.1.1 Features

- Support 2 voltage domains including VD_CORE, VD_LOGIC
- Support 4 separate power domains in the whole chip, which can be power up/down by software based on different application scenes
- In low power mode, the pmu could power up/down vd_core by hardware
- Support CORTEX-A7 core source clock gating in low power mode
- Support Logic Bus source clock gating in low power mode
- Support global interrupt disablein low power mode
- Support pd_pmuclock switch to 32KHz or pvtm clock in low power mode
- Support DDR self-refresh in low power mode
- Support DDR controller clock auto gating in low power mode
- Support to send idle requests to all NIU in the SoC (details will be described later)
- A group of configurable counter in PMU for HW control (such as PMIC, Core power up/down and so on)
- Support varies configurable wakeup source for low power mode

## 14.2 Block Diagram

### 14.2.1 power domain partition

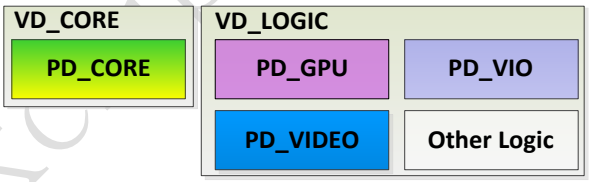


Fig.14-1Power Domain Partition

The above diagram describes the power domain and voltage domain partition, and the following table lists all the power domains.

Table 14-1RK3128 Power Domain and Voltage Domain Summary

| Voltage Domain | Power Domain | Description |
|---|---|---|
| VD_CORE | PD_CORE | A7 logic |
| VD_LOGIC | PD_BUS | include pd_bus, pd_peri and other system control unit (GRF, CRU and so on) |
| | PD_VIO | Video input/output system, include VOP, VIP, IEP, RGA, EBC, MIPI-DSI, HDMI |
| | PD_VIDEO | Video Encode&Decode , include VEPU，VDPU |
| | PD_GPU | GPU |

### 14.2.2 PMU block diagram

The following figure is the PMU block diagram. The PMU includes the 3 following sections:

- APB interface and register, which can accept the system configuration
- Low Power State Control, which generate low power control signals.
- Power Switch Control, which control all power domain switch

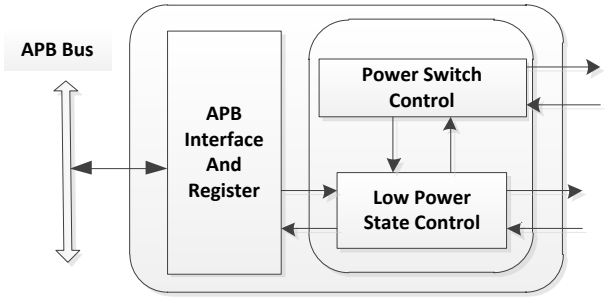


Fig.14-2PMU Bock Diagram

## 14.3 Power Switch Timing Requirement

The following table describe the switch time for power down and power up progress of each power domain. This table gives the time range, and each power domain switch time will be more than the min time and less than the max time.

Table 14-2 Power Switch Timing

| Power domain | type | Power down Switch Timing①(ns) | Power up Switch Timing①(ns) |
|---|---|---|---|
| PD_CORE | min | 170.3 | 132.4 |
| | max | 306.7 | 237.5 |
| | max | 199.0 | 156.1 |
| PD_VIO | min | 280.6 | 217.5 |
| | max | 518.5 | 407.8 |
| PD_VIDEO | min | 315.4 | 244.2 |
| | max | 586.2 | 460.4 |
| PD_GPU | min | 470.2 | 364 |
| | max | 871.4 | 684.1 |

***Notes:the power switch timing is just the chip power electrical parameter, this is not the parameter for the software to determine the power domain status. The software need to check each power domain status register to determine the power status.***

## 14.4 Function Description

### 14.4.1 Normal Mode

First of all, we define two modes of power for chip, normal mode and low power mode.

In normal mode, the PMU can power off/on all power domain (except pd_core)

by setting PMU_PWRDN_CON register. At same, pmu can send idle request for every power domain by setting PMU_IDLE_REQ register.

Don't set pd_core power off or send idle_req_core and idle_req_sys in normal mode. This will cause the system to not work properly.

Basically, there are 2 configurations that software can do in normal mode to save power.

- Configure DDR to self-refresh, DDR IO retention and DDR IO power off
- Power down power domains

The first one will save power consumption of using DDR controller and DDR IO. For avoiding confliction, the software must make sure the execution code of this step is not in DDR.

The second one will save power of the power domain which software is shutting down.

### 14.4.2 Low Power Mode

PMU can work in the Low Power Mode by setting bit[0] of PMU_PWRMODE_CON register. After setting the register, PMU would enter the Low Power mode.In the low power mode, pmu will auto power on/off the specified power domain, send idle req to specified power domain, disable/enable config bus clock and so on. All of above are configurable by setting corresponding registers.

Table 14-3 Low Power State

| Num | Hardware Flow | Description of Flow | |
|---|---|---|---|
| 0 | NORMAL | nomal status | |
| 1 | TRANS_NO_FIN | wait transfer to finish | bit[22:16] of PMU_PWRMODE_CON |
| 2 | SREF_ENTER | enter to self refresh | bit[8:7] of PMU_PWRMODE_CON |
| 3 | CORE_CLK_DIS | core clock gating | bit[1] of PMU_PWRMODE_CON |
| 4 | BUS_CLK_DIS | cfgbus clock disable | bit[2] of PMU_PWRMODE_CON |
| 5 | CLOCK_LF | switch to 32KHz or pvtm | bit[5] of PMU_PWRMODE_CON |
| 6 | CORE_PWRDN | vd core power down | bit[4] of PMU_PWRMODE_CON |
| 7 | WAIT_WAKEUP | wait wakeup | |
| 8 | WAIT_24M | wait 24MHz stable | bit[6] of PMU_PWRMODE_CON |
| 9 | CLOCK_HF | switch to 24MHz | |
| 10 | BUS_CLK_EN | cfgbus clock enable | |
| 11 | CORE_CLK_EN | core时钟恢复 | |
| 12 | SREF_EXIT | exit self refresh | |
| 13 | CORE_PWRUP | vd core power up | |
| 14 | TRANS_RESTORE | restroe transfer | |

The Low Power mode have three steps:

- Enter Low Power mode, there are some sub-steps in the enter step, every sub-step can be enable/disable by setting the corresponding register.
- Wait wakeup, there is only armint wakeup source by setting PMU_WAKEUP_CFG[0] register
- Exit Low Power mode, the sub-step are executed depend on whether they were executed in enter low power step.

### 14.4.3 Wakeup source

There is only one wakeup source, armint which can trigger PMU from power mode to normal mode.

If software expect PMU be woken up from power mode it should be enabled by

write 1 to PMU_WAKEUP_CFG[0] register before entering into power mode.

## 14.5 Register Description

### 14.5.1 Register Summary

| Name | Offset | Size | Reset Value | Description |
|---|---|---|---|---|
| PMU_WAKEUP_CFG | 0x0000 | W | 0x00000000 | PMU wake-up source configuration register |
| PMU_PWRDN_CON | 0x0004 | W | 0x00000000 | System power gating configuration register |
| PMU_PWRDN_ST | 0x0008 | W | 0x00000000 | System power gating status register |
| PMU_IDLE_REQ | 0x000c | W | 0x00000000 | PMU Noc idle req control |
| PMU_IDLE_ST | 0x0010 | W | 0x00000000 | PMU Noc idle status |
| PMU_PWRMODE_CON | 0x0014 | W | 0x00000000 | PMU configuration register in power mode flow |
| PMU_PWR_STATE | 0x0018 | W | 0x00000000 | PMU Low power mode state |
| PMU_OSC_CNT | 0x001c | W | 0x00005dc0 | 24MHz OSC stabilization counter threshold |
| PMU_CORE_PWRDWN_CNT | 0x0020 | W | 0x00005dc0 | CORE domain power down waiting counter in sleep mode |
| PMU_CORE_PWRUP_CNT | 0x0024 | W | 0x00005dc0 | CORE domain power up waiting counter in sleep mode |
| PMU_SFT_CON | 0x0028 | W | 0x00000000 | PMU Software control in normal mode |
| PMU_DDR_SREF_ST | 0x002c | W | 0x00000000 | PMU DDR self refresh status |
| PMU_INT_CON | 0x0030 | W | 0x00000000 | PMU interrupt configuration register |
| PMU_INT_ST | 0x0034 | W | 0x00000000 | PMU interrupt status register |
| PMU_SYS_REG0 | 0x0038 | W | 0x00000000 | PMU system register0 |
| PMU_SYS_REG1 | 0x003c | W | 0x00000000 | PMU system register1 |
| PMU_SYS_REG2 | 0x0040 | W | 0x00000000 | PMU system register2 |
| PMU_SYS_REG3 | 0x0044 | W | 0x00000000 | PMU system register3 |

### 14.5.2 Detail Register Description

**PMU_WAKEUP_CFG**

Address: Operational Base + offset (0x0000)

PMU wake-up source configuration register

| Bit | Attr | Reset Value | Description |
|---|---|---|---|
| 31:1 | RO | 0x0 | reserved |
| 0 | RW | 0x0 | armint_wakeup_en<br>ARM interrupt wake-up enable<br>1'b0: disable<br>1'b1: enable |

**PMU_PWRDN_CON**

Address: Operational Base + offset (0x0004)

System power gating configuration register

| Bit | Attr | Reset Value | Description |
|---|---|---|---|
| 31:4 | RO | 0x0 | reserved |

| Bit | Attr | Reset Value | Description |
|---|---|---|---|
| 3 | RW | 0x0 | PD_VIO_DWN_EN<br>Power domain VIO power down enable<br>1'b0: power on<br>1'b1: power off |
| 2 | RW | 0x0 | PD_VIDEO_DWN_EN<br>Power domain VIDEO power down enable<br>1'b0: power on<br>1'b1: power off |
| 1 | RW | 0x0 | PD_GPU_DWN_EN<br>Power domain GPU power down enable<br>1'b0: power on<br>1'b1: power off |
| 0 | RW | 0x0 | CORE_PWROFF_EN<br>software conifg power off pd_core<br>1'b1: power off<br>1'b0: not power off |

**PMU_PWRDN_ST**
Address: Operational Base + offset (0x0008)
System power gating status register

| Bit | Attr | Reset Value | Description |
|---|---|---|---|
| 31:4 | RO | 0x0 | reserved |
| 3 | RW | 0x0 | pd_vio_pwr_st<br>Power domain VIO power status<br>1'b0: power on<br>1'b1: power off |
| 2 | RW | 0x0 | pd_video_pwr_st<br>Power domain VIDEO power status<br>1'b0: power on<br>1'b1: power off |
| 1 | RW | 0x0 | pd_gpu_pwr_st<br>Power domain GPU power status<br>1'b0: power on<br>1'b1: power off |
| 0 | RW | 0x0 | pd_core_pwr_st<br>Power domain core power status<br>1'b0: power on<br>1'b1: power off |

**PMU_IDLE_REQ**
Address: Operational Base + offset (0x000c)
PMU Noc idle req control

| Bit | Attr | Reset Value | Description |
|---|---|---|---|
| 31:8 | RO | 0x0 | reserved |
| 7 | RW | 0x0 | idle_req_crypto_cfg<br>software configcrypto domain flush trasaction request<br>1'b1: idle req<br>1'b0: not idle req |

| Bit | Attr | Reset Value | Description |
|---|---|---|---|
| 6 | RW | 0x0 | idle_req_msch_cfg<br>software configmsch domain flush trasaction request<br>1'b1: idle req<br>1'b0: not idle req |
| 5 | RW | 0x0 | idle_req_sys_cfg<br>software configsystem domain flush trasaction request<br>1'b1: idle req<br>1'b0: not idle req |
| 4 | RW | 0x0 | idle_req_core_cfg<br>software configcore domain flush trasaction request<br>1'b1: idle req<br>1'b0: not idle req |
| 3 | RW | 0x0 | idle_req_gpu_cfg<br>software configgpu domain flush trasaction request<br>1'b1: idle req<br>1'b0: not idle req |
| 2 | RW | 0x0 | idle_req_vio_cfg<br>software configvio domain flush trasaction request<br>1'b1: idle req<br>1'b0: not idle req |
| 1 | RW | 0x0 | idle_req_video_cfg<br>software configvideo domain flush trasaction request<br>1'b1: idle req<br>1'b0: not idle req |
| 0 | RW | 0x0 | idle_req_peri_cfg<br>software configperi domain flush trasaction request<br>1'b1: idle req<br>1'b0: not idle req |

**PMU_IDLE_ST**

Address: Operational Base + offset (0x0010)

PMU Noc idle status

| Bit | Attr | Reset Value | Description |
|---|---|---|---|
| 31:24 | RO | 0x0 | reserved |
| 23 | RW | 0x0 | idle_ack_crypto<br>crypto domain flush transaction acknowledge<br>1'b0: no ack<br>1'b1: ack |
| 22 | RW | 0x0 | idle_ack_msch<br>msch domain flush transaction acknowledge<br>1'b0: no ack<br>1'b1: ack |

| Bit | Attr | Reset Value | Description |
|---|---|---|---|
| 21 | RW | 0x0 | idle_ack_sys<br>sys domain flush transaction acknowledge<br>1'b0: no ack<br>1'b1: ack |
| 20 | RW | 0x0 | idle_ack_core<br>core domain flush transaction acknowledge<br>1'b0: no ack<br>1'b1: ack |
| 19 | RW | 0x0 | idle_ack_gpu<br>gpu domain flush transaction acknowledge<br>1'b0: no ack<br>1'b1: ack |
| 18 | RW | 0x0 | idle_ack_vio<br>vio domain flush transaction acknowledge<br>1'b0: no ack<br>1'b1: ack |
| 17 | RW | 0x0 | idle_ack_video<br>video domain flush transaction acknowledge<br>1'b0: no ack<br>1'b1: ack |
| 16 | RW | 0x0 | idle_ack_peri<br>peri domain flush transaction acknowledge<br>1'b0: no ack<br>1'b1: ack |
| 15:8 | RO | 0x0 | reserved |
| 7 | RW | 0x0 | IDLE_CRYPTO<br>crypto domain flush transaction finish(idle)<br>1'b0: no finish<br>1'b1: finish |
| 6 | RW | 0x0 | IDLE_MSCH<br>msch domain flush transaction finish(idle)<br>1'b0: no finish<br>1'b1: finish |
| 5 | RW | 0x0 | IDLE_SYS<br>sys domain flush transaction finish(idle)<br>1'b0: no finish<br>1'b1: finish |
| 4 | RW | 0x0 | IDLE_CORE<br>core domain flush transaction finish(idle)<br>1'b0: no finish<br>1'b1: finish |
| 3 | RW | 0x0 | IDLE_GPU<br>gpu domain flush transaction finish(idle)<br>1'b0: no finish<br>1'b1: finish |
| 2 | RW | 0x0 | IDLE_VIO<br>vio domain flush transaction finish(idle)<br>1'b0: no finish<br>1'b1: finish |

| Bit | Attr | Reset Value | Description |
|---|---|---|---|
| 1 | RW | 0x0 | IDLE_VIDEO<br>video domain flush transaction finish(idle)<br>1'b0: no finish<br>1'b1: finish |
| 0 | RW | 0x0 | IDLE_PERI<br>peri domain flush transaction finish(idle)<br>1'b0: no finish<br>1'b1: finish |

**PMU_PWRMODE_CON**

Address: Operational Base + offset (0x0014)

PMU configuration register in power mode flow

| Bit | Attr | Reset Value | Description |
|---|---|---|---|
| 31:24 | RO | 0x0 | reserved |
| 23 | RW | 0x0 | clr_peri<br>issue idle_req_peri in low power mode<br>1'b0: not issue<br>1'b1: issue |
| 22 | RW | 0x0 | clr_video<br>issue idle_req_video in low power mode<br>1'b0: not issue<br>1'b1: issue |
| 21 | RW | 0x0 | clr_vio<br>issue idle_req_vio in low power mode<br>1'b0: not issue<br>1'b1: issue |
| 20 | RW | 0x0 | clr_gpu<br>issue idle_req_gpu in low power mode<br>1'b0: not issue<br>1'b1: issue |
| 19 | RW | 0x0 | clr_core<br>issue idle_req_core in low power mode<br>1'b0: not issue<br>1'b1: issue |
| 18 | RW | 0x0 | clr_sys<br>issue idle_req_sys in low power mode<br>1'b0: not issue<br>1'b1: issue |
| 17 | RW | 0x0 | clr_msch<br>issue idle_req_msch in low power mode<br>1'b0: not issue<br>1'b1: issue |
| 16 | RW | 0x0 | clr_crypto<br>issue idle_req_crypto in low power mode<br>1'b0: not issue<br>1'b1: issue |
| 15:10 | RW | 0x0 | ddr0io_ret_de_req<br>ddr0io retention de-assert request<br>1'b0: de-assert request<br>1'b1: not de-assert request |

| Bit | Attr | Reset Value | Description |
|---|---|---|---|
| 9 | RW | 0x0 | pmu_int_en<br>pmuint enable<br>1'b1: enable<br>1'b0: disable |
| 8 | RW | 0x0 | ddr_gating_en<br>ddrc auto gating in low power mode<br>1'b0: disable<br>1'b1: enable |
| 7 | RW | 0x0 | sref_enter_en<br>DDR enter self-refresh enable in low power mode<br>1'b0: disable DDR enter self-refresh<br>1'b1: enable DDR enter self-refresh |
| 6 | RW | 0x0 | wait_osc_24m<br>wait 24MHz OSC when wakeup in low power mode<br>1'b0: disable<br>1'b1: enable |
| 5 | RW | 0x0 | pmu_use_lf<br>pmu domain clock switch to low clock enable<br>1'b0: not switch to low clock<br>1'b1: switch to low clock |
| 4 | RW | 0x0 | core_pd_en<br>core power off enable in low power mode<br>1'b0: core power on<br>1'b1: core power off |
| 3 | RW | 0x0 | global_int_disable<br>Global interrupt disable<br>1'b0: enable global interrupt<br>1'b1: disable global interrupt |
| 2 | RW | 0x0 | clk_bus_src_gate_en<br>config bus clock source gating enable in idle mode<br>1'b0: enable<br>1'b1: disable |
| 1 | RW | 0x0 | clk_core_src_gate_en<br>A7 core clock source gating enable in idle mode<br>1'b0: enable<br>1'b1: disable |
| 0 | RW | 0x0 | power_mode_en<br>power mode flow enable<br>1'b0: disable<br>1'b1: enable |

**PMU_PWR_STATE**

Address: Operational Base + offset (0x0018)

PMU Low power mode state

| Bit | Attr | Reset Value | Description |
|---|---|---|---|
| 31:15 | RO | 0x0 | reserved |

| Bit | Attr | Reset Value | Description |
|---|---|---|---|
| 14 | RW | 0x0 | TRANS_RESTORE<br>noc trans restore<br>1'b0: state not happened<br>1'b1: state happened |
| 13 | RW | 0x0 | CORE_PWRUP<br>pd core power up state<br>1'b0: state not happened<br>1'b1: state happened |
| 12 | RW | 0x0 | SREF_EXIT<br>ddr exit self-refresh<br>1'b0: state not happened<br>1'b1: state happened |
| 11 | RW | 0x0 | CORE_CLK_EN<br>pd_core source clock enable<br>1'b0: state not happened<br>1'b1: state happened |
| 10 | RW | 0x0 | BUS_CLK_EN<br>config bus source clock enable<br>1'b0: state not happened<br>1'b1: state happened |
| 9 | RW | 0x0 | CLOCK_HF<br>pd_pmu switch to normal clock<br>1'b0: state not happened<br>1'b1: state happened |
| 8 | RW | 0x0 | WAIT_24M<br>wait 24M osc stable<br>1'b0: state not happened<br>1'b1: state happened |
| 7 | RW | 0x0 | WAIT_WAKEUP<br>wati wakeup state<br>1'b0: state not happened<br>1'b1: state happened |
| 6 | RW | 0x0 | CORE_PWRDN<br>pd core down state<br>1'b0: state not happened<br>1'b1: state happened |
| 5 | RW | 0x0 | CLOCK_LF<br>pd_pmu switch to low speed clock<br>1'b0: state not happened<br>1'b1: state happened |
| 4 | RW | 0x0 | BUS_CLK_DIS<br>config bus source clock disable<br>1'b0: state not happened<br>1'b1: state happened |
| 3 | RW | 0x0 | CORE_CLK_DIS<br>pd_core source clock disable<br>1'b0: state not happened<br>1'b1: state happened |
| 2 | RW | 0x0 | SREF_ENTER<br>ddrselfrefresh enter<br>1'b0: state not happened<br>1'b1: state happened |

| Bit | Attr | Reset Value | Description |
|---|---|---|---|
| 1 | RW | 0x0 | TRANS_NO_FIN<br>transfer no finish<br>1'b0: state not happened<br>1'b1: state happened |
| 0 | RW | 0x0 | NORMAL<br>normal state<br>1'b0: state not happened<br>1'b1: state happened |

**PMU_OSC_CNT**
Address: Operational Base + offset (0x001c)
24MHz OSC stabilization counter threshold

| Bit | Attr | Reset Value | Description |
|---|---|---|---|
| 31:0 | RW | 0x05dc0 | osc_stabl_cnt_thresh<br>24MHz OSC stabilization counter threshold |

**PMU_CORE_PWRDWN_CNT**
Address: Operational Base + offset (0x0020)
CORE domain power down waiting counter in sleep mode

| Bit | Attr | Reset Value | Description |
|---|---|---|---|
| 31:0 | RW | 0x05dc0 | core_pwrdwn_cnt_thresh<br>CORE domain power down waiting counter threshold |

**PMU_CORE_PWRUP_CNT**
Address: Operational Base + offset (0x0024)
CORE domain power up waiting counter in sleep mode

| Bit | Attr | Reset Value | Description |
|---|---|---|---|
| 31:0 | RW | 0x05dc0 | core_pwrup_cnt_thresh<br>CORE domain power up waiting counter threshold |

**PMU_SFT_CON**
Address: Operational Base + offset (0x0028)
PMU Software control in normal mode

| Bit | Attr | Reset Value | Description |
|---|---|---|---|
| 31:5 | RO | 0x0 | reserved |
| 4 | RW | 0x0 | clk_core_src_gating_cfg<br>pd core source clock disable<br>1'b1: disable<br>1'b0: enable |
| 3 | RW | 0x0 | clk_bus_src_gating_cfg<br>config bus source clock disable<br>1'b1: disable<br>1'b0: enable |
| 2 | RW | 0x0 | upctl_c_sysreq_cfg<br>software config enter DDR self-refresh by lowpower interface<br>1'b1: request enter self-refresh<br>1'b0: not enter self-refresh |

| Bit | Attr | Reset Value | Description |
|---|---|---|---|
| 1 | RW | 0x0 | pmu_lf_ena_cfg<br>software config PMU domain clock switch to low clock<br>1'b1: switch to low speed clock<br>1'b0: not switch |
| 0 | RW | 0x0 | low_clk_sel<br>low clock in low power mode select<br>1'b0: clock pvtm<br>1'b1: 32KHz clock |

**PMU_DDR_SREF_ST**

Address: Operational Base + offset (0x002c)

PMU DDR self refresh status

| Bit | Attr | Reset Value | Description |
|---|---|---|---|
| 31:2 | RO | 0x0 | reserved |
| 1 | RW | 0x0 | upctl_c_sysack<br>DDR enter self-refresh acknowledge<br>1'b0: no ack<br>1'b1: ack |
| 0 | RW | 0x0 | upctl_c_active<br>DDR enter self-refresh<br>1'b0: no active<br>1'b1: active |

**PMU_INT_CON**

Address: Operational Base + offset (0x0030)

PMU interrupt configuration register

| Bit | Attr | Reset Value | Description |
|---|---|---|---|
| 31:6 | RO | 0x0 | reserved |
| 5 | RW | 0x0 | pd_core_int_en<br>Power domain core power switch interrupt enable<br>1'b0: disable<br>1'b1: enable |
| 4 | RW | 0x0 | pd_gpu_int_en<br>Power domain gpu power switch interrupt enable<br>1'b0: disable<br>1'b1: enable |
| 3 | RW | 0x0 | pd_video_int_en<br>Power domain video power switch interrupt enable<br>1'b0: disable<br>1'b1: enable |
| 2 | RW | 0x0 | pd_vio_int_en<br>Power domain vio power switch interrupt enable<br>1'b0: disable<br>1'b1: enable |

| Bit | Attr | Reset Value | Description |
|---|---|---|---|
| 1 | RW | 0x0 | wakeup_int_en<br>wakeup status interrupt enable<br>1'b0: disable<br>1'b1: enable |
| 0 | RW | 0x0 | armint_wakeup_int_en<br>ARM interrupt wakeup status interrupt enable<br>1'b0: disable<br>1'b1: enable |

**PMU_INT_ST**

Address: Operational Base + offset (0x0034)

PMU interrupt status register

| Bit | Attr | Reset Value | Description |
|---|---|---|---|
| 31:6 | RO | 0x0 | reserved |
| 5 | RW | 0x0 | pd_core_int_st<br>Power domain core power switch status<br>1'b0: no power switch happen<br>1'b1: power switch happen |
| 4 | RW | 0x0 | pd_gpu_int_st<br>Power domain gpu power switch status<br>1'b0: no power switch happen<br>1'b1: power switch happen |
| 3 | RW | 0x0 | pd_video_int_st<br>Power domain video power switch status<br>1'b0: no power switch happen<br>1'b1: power switch happen |
| 2 | RW | 0x0 | pd_vio_int_st<br>Power domain vio power switch status<br>1'b0: no power switch happen<br>1'b1: power switch happen |
| 1 | RW | 0x0 | pwrmode_wakeup_event_trig<br>power mode flow wakeup<br>1'b0: no wakeup<br>1'b1: wakeup |
| 0 | RW | 0x0 | armint_wakeup_event_trig<br>ARM interrupt wake-up enent trigger<br>1'b0: no wakeup<br>1'b1: wakeup |

**PMU_SYS_REG0**

Address: Operational Base + offset (0x0038)

PMU system register0

| Bit | Attr | Reset Value | Description |
|---|---|---|---|
| 31:0 | RW | 0x00000000 | pmu_sys_reg0<br>PMU system register0 |

**PMU_SYS_REG1**

Address: Operational Base + offset (0x003c)

PMU system register1

| Bit | Attr | Reset Value | Description |
|---|---|---|---|
| 31:0 | RW | 0x00000000 | pmu_sys_reg1<br>PMU system register1 |

**PMU_SYS_REG2**
Address: Operational Base + offset (0x0040)
PMU system register2

| Bit | Attr | Reset Value | Description |
|---|---|---|---|
| 31:0 | RW | 0x00000000 | pmu_sys_reg2<br>PMU system register2 |

**PMU_SYS_REG3**
Address: Operational Base + offset (0x0044)
PMU system register3

| Bit | Attr | Reset Value | Description |
|---|---|---|---|
| 31:0 | RW | 0x00000000 | pmu_sys_reg3<br>PMU system register3 |

# 14.6 Timing Diagram

## 14.6.1 Each domain power switch timing

The following figure is the each domain power down and power up timing.

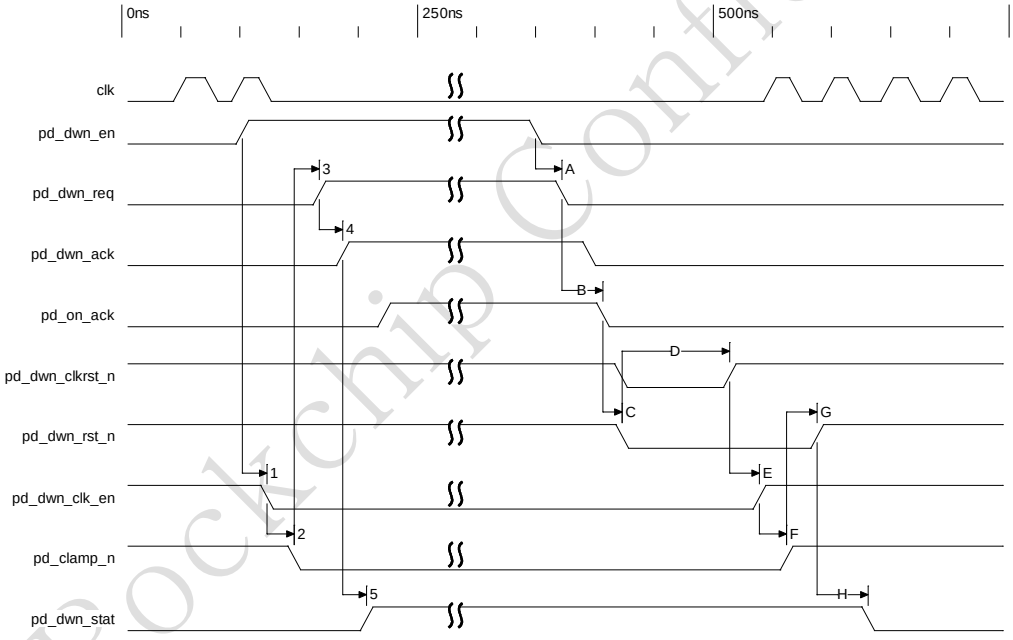


Fig.14-3 Each Domain Power Switch Timing

## 14.6.2 External wakeup PAD timing

The PMU supports a lot of external wakeup sources, such as SD/MMDC, USBDEV, SIM0/1 detect wakeup, GPIO0 wakeup source and so on. All these external wakeup sources must meet the timing requirement (at least 200us) when the wakeup event is asserted. The following figure gives the timing information.

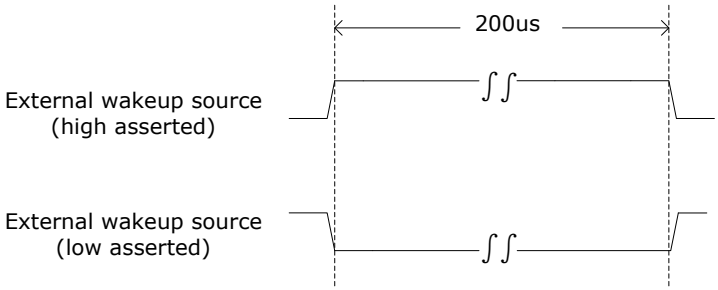


Fig.14-4 External Wakeup Source PAD Timing

## 14.7 Application Notes

### 14.7.1 Recommend configurations for power mode.

The PMU is a design with great flexibilities, but just for facilities and inheritances, a group of recommend configurations will be shown below for software. And for convenience, we will define several modes.

The RK3128can support following 5 recommended power modes:

- normal
- idle mode
- deep idle mode
- sleep mode
- power off mode

The following table lists the detailed description of the modes.

Table 14-4 Power Domain Status Summary in all Work Mode

| Item | mode1(idle) | mode2(sleep) | configurable |
|---|---|---|---|
| Core | standy | power down | |
| Logic Clock | enable | disable | Yes |
| Logic Clock Source | 24MHz | 32KHz/PVTM | Yes |
| PLL | working | power down | Yes |
| DDR | working | self refresh | Yes |

**Normal mode**

In this mode, you can power off/on or enable/disable the following power domain to save power: PD_VIO/PD_VIDEO/PD_GPU

**Idle mode**

This mode is used when the core do not have load for a shot while such as waiting for interrupt and the software want to save power by gating Cortex-A7 source clock.

In idle mode, core1/2/3 of Cortex-A7 should be either power off or in WFI/WFE state. The core0 of A7 should be in WFI/WFE state. The configurations of core clock source gating and disable global interrupt are presented. The Cortex-A7 can waked up by an interrupt.

**Sleep mode**

The sleep mode can power off all power domains except TOP Logic. The VD_CORE is turned off externally, and other domains power off by software.

In sleep mode the clock of PD_PMU can be switched from 24MHz to low speedclock optionally by hardware.The low speed clock can be selected from clock pvtm and 32KHz clock.

In sleep mode all PLLs power down mandatorily to save power by software.

In sleep mode OSC can be disabled optionally by software.

In sleep mode DDR self-refresh can be issued by hardware mandatorily.

### 14.7.2 System Register

PMU support 4 words register: PMU_SYS_REG0, PMU_SYS_REG1, PMU_SYS_REG2, PMU_SYS_REG3. These registers are always on no matter what low power mode. So software can use these registers to retain some information which is useful after wakeup from any mode.

### 14.7.3 Configuration Constraint

In order to shut down the power domains correctly, the software must obey the rules bellow:

- Send NIU request to the NIU in power domain that you want to shut down.
- Querying PMU_IDLE_ST register to get the information until the pacific NIU is in idle state.
- Send power request to the power domain through PMU_PWRDN_CON register.
- Querying PMU_PWRDN_ST register to make sure the pacific power domain is power down.

The power domains controlled only by software are showing below:

PD_VIO, PD_GPU, PD_VIDEO.

So you must power off these power domains before enter low power mode if you need.