Sleep and Wake up by GPIO on LS1028

This document introduces basic concept of Power Management, LS1028 RCW configuration to enable GPIO, Linux Kernel source and device tree modification to support GPIO wakeup, Kernel configuration to enable sleep feature and GPIO wakeup driver, export GPIO pin and enable interrupt, Order system to sleep and trigger GPIO interrupt to wake up the system.

1. Power Management

QorIQ Processors have features to minimize power consumption at several different levels. All processors support sleep mode (LPM20/SWLPM20). Some processors also support deep sleep mode (LPM35).

• Processors enter low-power state (LPM20/SWLPM20, LPM35)
— LPM20/SWLPM20 mode: most parts of processor clocks are shut down
— LPM35 mode: power is removed to cores, cache and IP blocks of the processor, such as DIU, eLBC, PEX, eTSEC, USB, SATA, eSDHC
The wake-up event sources caused quitting from low-power mode are listed as below:
— Wake on LAN (WoL) using magic packet
— Wake by MPIC timer or FlexTimer
— Wake by Internal and external interrupts, such as GPIO

2. LS1028 RCW configuration to enable GPIO

For GPIO wakeup feature, some GPIO pins are muxed with other signal on SoC/board. Therefore, the user needs to ensure that the correct RCW image is used.
In this document enabling GPIO3_DAT12 on LS1028ARDB.
Reset Configuration Word Status Register 27 (RCWSR27) is as the following.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RESERVED_895_862		GTX_CLK125_PMUX		SPI3_PMUX		ENETC_RCW								USB3_CLK_FSEL	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	USB3_CLK_FSEL				USB_PWRFAULT_PM...		USB_DRVVBUS_PMUX		RESERVE...	EC1_SAI3_6_PMUX			EC1_SAI4_5_PMUX			RESERVE...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Field	Function
3-1 EC1_SAI4_5_PMUX	EC1 Pin Configuration Configures the functionality of the EC1 pins 0b000 - EC1_TXD[3:2], EC1_TX_EN, EC1_GTX_CLK, EC1_RXD[3:2] 0b001 - GPIO3_DAT[12:11], GPIO3_DAT[8:7], GPIO3_DAT[5:4] 0b010 - SAI5_TX_DATA, SAI4_TX_DATA, SAI5_TX_SYNC, SAI4_TX_SYNC, SAI5_TX_BCLK, SAI4_TX_BCLK 0b011 - SAI5_RX_DATA, SAI4_RX_DATA, SAI5_RX_SYNC, SAI4_RX_SYNC, SAI5_RX_BCLK, SAI4_RX_BCLK 0b100 - FTM1_CH7, FTM1_CH6, FTM1_FAULT, FTM1_EXTCLK, FTM1_CH3, FTM1_CH2 0b101 - EC1_1588_PULSE_OUT1, EC1_1588_ALARM_OUT1, SWITCH_1588_DAT[1:0], EC1_1722_DAT[1], GPIO3_DAT[4] 0b110 - Reserved 0b111 - Reserved

Modify rcw/ls1028ardb/R_SQPP_0x85bb/rcw_1500_gpu600.rcw, change EC1_SAI4_5_PMUX to 1.

```
#include <../ls1028asi/ls1028a.rcwi>
…
CLK_OUT_PMUX=2
EC1_SAI4_5_PMUX=1
EC1_SAI3_6_PMUX=5
USB3_CLK_FSEL=39
…
```

3. Linux Kernel source and device tree modification to support GPIO wakeup

In the Linux kernel source file drivers/gpio/gpio-mpc8xxx.c, apply the following patch to mpc8xxx_probe() to enable the GPIO irq wake function.

```
---
 drivers/gpio/gpio-mpc8xxx.c | 2 ++
 1 file changed, 2 insertions(+)

diff --git a/drivers/gpio/gpio-mpc8xxx.c b/drivers/gpio/gpio-mpc8xxx.c
index 2360f01..a3dedc8 100644
--- a/drivers/gpio/gpio-mpc8xxx.c
+++ b/drivers/gpio/gpio-mpc8xxx.c
@@ -425,6 +425,8 @@ static int mpc8xxx_probe(struct platform_device *pdev)
                goto err;
        }

+       device_init_wakeup(&pdev->dev, true);
```

```
+       enable_irq_wake(mpc8xxx_gc->irqn);
        return 0;
 err:
        iounmap(mpc8xxx_gc->regs);
--
2.7.4
```

In the Linux kernel source file arch/arm64/boot/dts/freescale/fsl-ls1028a.dtsi, add property fsl,rcpm-wakeup = <&rcpm 0x0 0x0 0x0 0x0 0x200 0x0 0x0> to node gpio3: gpio@2320000 to apply proper programming on IPPDEXPCRn for GPIO wakeup

```
                rcpm: power-controller@1e34040 {
                        compatible = "fsl,ls1028a-rcpm", "fsl,qoriq-rcpm-2.1+";
                        reg = <0x0 0x1e34040 0x0 0x1c>;
                        #fsl,rcpm-wakeup-cells = <7>;
                        little-endian;
                };

                ftm_alarm0: timer@2800000 {
                        compatible = "fsl,ls1028a-ftm-alarm";
                        reg = <0x0 0x2800000 0x0 0x10000>;
                        fsl,rcpm-wakeup = <&rcpm 0x0 0x0 0x0 0x0 0x4000 0x0 0x0>;
                        interrupts = <GIC_SPI 44 IRQ_TYPE_LEVEL_HIGH>;
                        status = "disabled";
                };

                ftm_alarm1: timer@2810000 {
                        compatible = "fsl,ls1028a-ftm-alarm";
                        reg = <0x0 0x2810000 0x0 0x10000>;
                        fsl,rcpm-wakeup = <&rcpm 0x0 0x0 0x0 0x0 0x4000 0x0 0x0>;
                        interrupts = <GIC_SPI 45 IRQ_TYPE_LEVEL_HIGH>;
                        status = "disabled";
                };

                gpio3: gpio@2320000 {
                        compatible = "fsl,ls1028a-gpio","fsl,qoriq-gpio";
                        reg = <0x0 0x2320000 0x0 0x10000>;
                        interrupts = <GIC_SPI 37 IRQ_TYPE_LEVEL_HIGH>;
                        gpio-controller;
                        #gpio-cells = <2>;
                        interrupt-controller;
                        #interrupt-cells = <2>;
                        little-endian;
                        fsl,rcpm-wakeup = <&rcpm 0x0 0x0 0x0 0x0 0x200 0x0 0x0>;
                };
```

The RCPM performs all device-level tasks associated with device run control
and power management.

Required properties:
- reg : Offset and length of the register set of the RCPM block.
- #fsl,rcpm-wakeup-cells : The number of IPPDEXPCR register cells in the
    fsl,rcpm-wakeup property.
- compatible : Must contain a chip-specific RCPM block compatible string
    and (if applicable) may contain a chassis-version RCPM compatible
    string. Chip-specific strings are of the form "fsl,<chip>-rcpm",
    such as:
    * "fsl,p2041-rcpm"
    * "fsl,p5020-rcpm"
    * "fsl,t4240-rcpm"

    Chassis-version strings are of the form "fsl,qoriq-rcpm-<version>",
    such as:
    * "fsl,qoriq-rcpm-1.0": for chassis 1.0 rcpm
    * "fsl,qoriq-rcpm-2.0": for chassis 2.0 rcpm
    * "fsl,qoriq-rcpm-2.1": for chassis 2.1 rcpm
    * "fsl,qoriq-rcpm-2.1+": for chassis 2.1+ rcpm

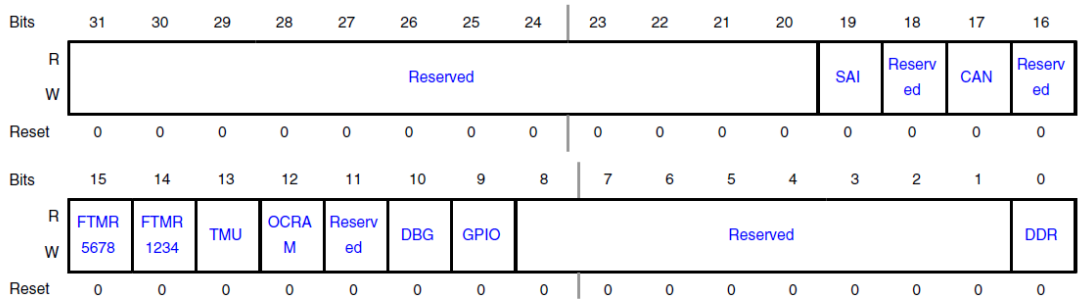
- fsl,rcpm-wakeup: Consists of a phandle to the rcpm node and the IPPDEXPCR
    register cells. The number of IPPDEXPCR register cells is defined in
    "#fsl,rcpm-wakeup-cells" in the rcpm node. The first register cell is
    the bit mask that should be set in IPPDEXPCR0, and the second register
    cell is for IPPDEXPCR1, and so on.

    Note: IPPDEXPCR(IP Powerdown Exception Control Register) provides a
    mechanism for keeping certain blocks awake during STANDBY and MEM, in
    order to use them as wake-up sources.

IPPDEXPCRn provides a mechanism for preventing clock gating certain IP during device sleep in order to make certain IP a
wake-up source. For example,
• Wake on LAN (magic packet) - Requires ethernet controller preventing clock gating during LPM20
• Wake on USB (unplug/plug event) - Requires USB controller preventing clock gating during LPM20
• Wake on eSDHC (card detect event) - Requires eSDHC controller preventing clock gating during LPM20
• Wake on GPIO - Requires GPIO controller preventing clock gating during device sleep

IP Powerdown Exception Control Register 4 (IPPDEXPCR4) definition is as the following.

Diagram



Field	Function
12 OCRAM	OCRAM and TZPC Powerdown Exception 0b - OCRAM and TZPC are powerdown during device sleep 1b - OCRAM and TZPC are not powerdown during device sleep
11 —	Reserved
10 DBG	DBG IP Powerdown Exception 0b - All DBG IPs are powerdown during device sleep 1b - All DBG IPs are not powerdown during device sleep
9 GPIO	GPIO1-3 Powerdown Exception 0b - GPIO1-3 are powerdown during device sleep 1b - GPIO1-3 are not powerdown during device sleep
8-1 —	Reserved
0 DDR	DDR Powerdown Exception 0b - DDR is powerdown during device sleep 1b - DDR is not powerdown during device sleep

4. Kernel configuration to enable Sleep feature and GPIO wakeup driver

Kernel configure tree view options

Kernel configure tree view options	Description
`Power management options  -->` `  [*] Suspend to RAM and standby`	Enable sleep feature

`Device Drivers --->`
`  -*- GPIO Support  -->`
`  [*] /sys/class/gpio/…(sysfs interface)`
`  Memory mapped GPIO drivers  --->`
`  [*] MPC512x/MPC8xxx/QorIQ GPIO support`
	Enable GPIO driver (for GPIO wakeup case only)

Compile-time configuration options

CONFIG_SUSPEND

CONFIG_GPIO_MPC8XXX

5. export GPIO pin and enable interrupt

Boot to Linux console, execute following commands to export specific GPIO pin in the Linux user space and enable interrupt, order system to sleep (Suspend-RAM/Suspend-to-Idle).
echo 428 > /sys/class/gpio/export # Export related GPIO pin in userspace
echo input > /sys/class/gpio/gpio428/direction
echo falling > /sys/class/gpio/gpio428/edge

GPIOs are defied as they are inside the kernel, using integers in the range 0 .. INT_MAX.
/sys/class/gpio
/export : asks the kernel to export a GPIO to userspace
/unexport: to return a GPIO to the kernel
/gpioN: for each exported GPIO #N
/value: always readable, writes fail for input GPIOs
/direction: r/w as in, out (default low); write(high, low)
/edge: r/w as none, falling, rising, both
/gpiochipN: for each gpiochip; #N is its first GPIO
/base: (r/o) same as N
/label: (r/o) descriptive, not necessarily unique
/ngpio: (r/o) number of GPIOs; numbered N to N + (ngpio - 1)

GPIO block mapping could be get as the following.
root@ls1028ardb:~# cat /sys/kernel/debug/gpio
gpiochip2: GPIOs 416-447, parent: platform/2320000.gpio, 2320000.gpio:

gpiochip1: GPIOs 448-479, parent: platform/2310000.gpio, 2310000.gpio:

gpiochip0: GPIOs 480-511, parent: platform/2300000.gpio, 2300000.gpio:

GPIO3_DAT12 is GPIO428(416+12)

6. Order system to sleep (Suspend-RAM/Suspend-to-Idle).

echo mem > /proc/power/state # Suspend-to-RAM
echo freeze > /proc/power/state # Suspend-to-Idle

7. Trigger GPIO interrupt to wake up the system.

On LS1028ARDB, short J11 pin 1 and 11 to trigger GPIO interrupt (as shown in the following figure), to wake up the system.

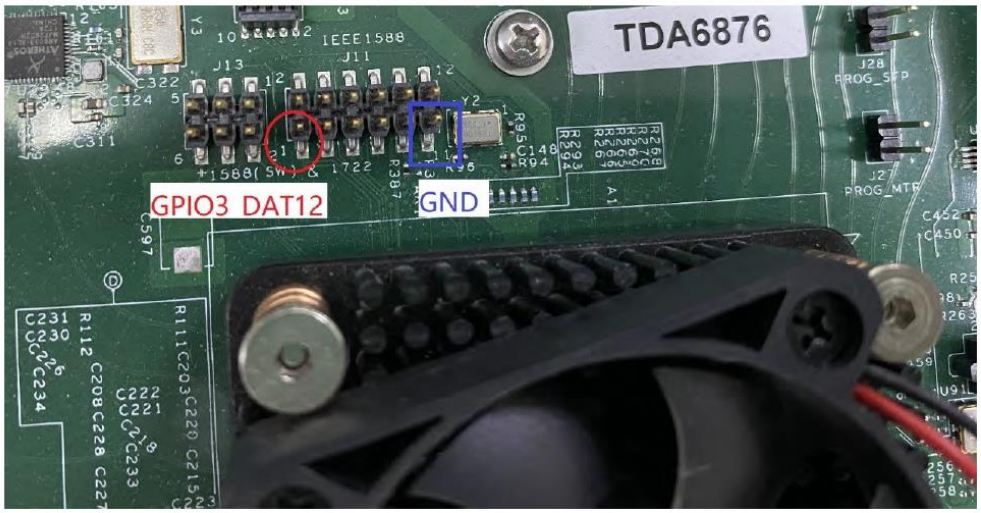


Figure 245. GPIO