AN11561

LPC800 low power modes and wake-up times

Rev. 2 — 4 August 2014 Application note

Document information

Info	Content
Keywords	LPC800 , Low Power Modes, Power Consumption, Wake-up, code example, LPCXpresso
Abstract	This application note introduces the low power modes of the LPC800 series, wake-up implementation, and helpful hints to reduce power consumption. This application note also provides a software example to enter the low power modes, and demonstrates how to measure the power consumption and wake-up times using the LPC800 LPCXpresso board.

Revision history

Rev	Date	Description
2	20140804	Image print quality improved.
1	20140618	Initial version.

Contact information

For additional information, please visit: http://www.nxp.com

For sales office addresses, please send an email to: salesaddresses@nxp.com

1. Introduction

The LPC800 series are an ARM Cortex-M0+ based, low-cost 32-bit MCU family operating at CPU frequencies of up to 30 MHz. The LPC800 supports up to 16 KB of flash memory and 4 KB of SRAM.

The peripheral complement of the LPC800 includes one I^2C-bus interface, up to three USARTs, up to two SPI interfaces, one multi-rate timer, self wake-up timer, and state-configurable timer, one comparator, function-configurable I/O ports through a switch matrix, an input pattern match engine, a CRC engine, and up to 18 general-purpose I/O pins.

The LPC800 family targets a wide range of 8/16-bit applications, including lighting, consumer, climate control, fire and security applications, and motor control applications.

This application note introduces the steps required to enter the low power modes, the wake-up implementation, and helpful hints to reduce power consumption. It also provides software examples to enter and wake-up from the low power modes, and demonstrates how to measure the power consumption and wake-up times with the LPC800 LPCXpresso board.

The various topics covered in this application note are as follows:

1. Low Power Modes
2. Entering Low Power Modes
3. Wake-Up Implementation
4. Additional hints to reduce power consumption
5. Low power mode exercises using the LPC800 LPCXpresso board

2. Low power modes

On the LPC800 series, there are four reduced power modes: Sleep, Deep-Sleep, Power-down, and Deep power-down modes. The following sections cover the features and configurations for the low power modes.

2.1.1 Sleep mode

In Sleep mode, the system clock to the ARM Cortex-M0+ core is stopped, and execution of instructions is suspended until either a reset or an enabled interrupt occurs.

Peripheral functions, if selected to be clocked in the SYSAHBCLKCTRL register, continue operation during Sleep mode and may generate interrupts to cause the processor to resume execution. Sleep mode eliminates dynamic power used by the processor itself, memory systems and their related controllers, and internal buses. The processor state and registers, peripheral registers, and internal SRAM values are maintained, and the logic levels of the pins remain static.

2.1.2 Deep-sleep mode

In Deep-sleep mode, the system clock to the processor is disabled as in Sleep mode. All analog blocks are powered down, except for the BOD circuit and the watchdog oscillator, which can be selected or deselected during Deep-sleep mode in the PDSLEEPCFG register. The main clock and all peripheral clocks are disabled except for the clock to the watchdog timer if the watchdog oscillator is selected. The IRC is running, but its output is disabled. The flash is in stand-by mode.

Deep-sleep mode eliminates all power used by analog peripherals and all dynamic power used by the processor itself, memory systems and related controllers, and internal buses. The processor state and registers, peripheral registers, and internal SRAM values are maintained, and the logic levels of the pins remain static.

2.1.3 Power-down mode

In Power-down mode, the system clock to the processor is disabled as in Sleep mode. All analog blocks are powered down, except for the BOD circuit and the watchdog oscillator, which can be selected or deselected during Power-down mode. The main clock and therefore all peripheral clocks are disabled except for the clock to the watchdog timer if the watchdog oscillator is selected. The IRC itself and the flash are powered down, decreasing power consumption compared to Deep-sleep mode.

Power-down mode eliminates all power used by analog peripherals and all dynamic power used by the processor itself, memory systems and their related controllers, and internal buses. The processor state and registers, peripheral registers, and internal SRAM values are maintained, and the logic levels of the pins remain static. Wake-up times are longer compared to sleep and deep sleep modes.

2.1.4 Deep power-down mode

In Deep power-down mode, power and clocks are shut off to the entire chip with the exception of the $\overline{\text{WAKEUP}}$ pin and the self wake-up timer.

During Deep power-down mode, the contents of the SRAM and registers are not retained except for a small amount of data which can be stored in four general purpose registers (GPREG0 to GPREG3) and the deep power down control register (DPDCTRL, bits 4 to 31) of the power management unit block.

The table below provides a summary of the power control features for various low power modes:

Peripheral	Sleep mode	Deep-sleep mode	Power-down mode	Deep power-down mode
IRC	software configurable	on	off	off
IRC output	software configurable	off	off	off
Flash	software configurable	on	off	off
BOD	software configurable	software configurable	software configurable	off
PLL	software configurable	off	off	off
SysOsc	software configurable	off	off	off
WDosc/WWDT	software configurable	software configurable	software configurable	off
Digital peripherals	software configurable	off	off	off
WKT/low-power oscillator	software configurable	software configurable	software configurable	software configurable

Fig 1. Peripheral configuration features in reduced power modes

3. Entering the low power modes and wake-up implementation

This section describes the mechanism to put the LPC800 series into the four low power modes (sleep, deep-sleep, power-down, deep power-down). It will also cover the available wake-up sources for each mode.

3.1 System Control Register (SCR)

The SCR register controls features of entry to and exit from low power modes.

The bit assignments are shown in Fig 2.

System control register (SCR, address 0xE000 ED10) bit description

Bit	Symbol	Description	Reset value
0	-	Reserved.	0
1	SLEEPONEXIT	Indicates sleep-on-exit when returning from Handler mode to Thread mode: 0 = do not sleep when returning to Thread mode. 1 = enter sleep, or deep sleep, on return from an ISR to Thread mode. Setting this bit to 1 enables an interrupt driven application to avoid returning to an empty main application.	0
2	SLEEPDEEP	Controls whether the processor uses sleep or deep-sleep as its low power mode: 0 = sleep 1 = deep sleep.	0
3	-	Reserved.	0
4	SEVONPEND	Send Event on Pending bit: 0 = only enabled interrupts or events can wake-up the processor, disabled interrupts are excluded 1 = enabled events and all interrupts, including disabled interrupts, can wake up the processor. When an event or interrupt enters pending state, the event signal wakes up the processor from WFE. If the processor is not waiting for an event, the event is registered and affects the next WFE. The processor also wakes up on execution of an SEV instruction.	0
31:5	-	Reserved.	0

Fig 2. System Control Register (SCR, 0xE000ED10)

If the SLEEPDEEP bit in the Cortex-M0+ System Control Register (SCR) is 0, sleep mode can be selected.

If the SLEEPDEEP bit in the Cortex-M0+ System Control Register (SCR) is 1, Deep-sleep, Power-down, and Deep power-down modes can be selected.

3.2 Power Control Register (PCON)

After setting the SLEEPDEEP bit mentioned above, the PCON register (bits 2:0) must be set appropriately to select the desired low power-down modes. It also provides the flags for the low power modes respectively. See Fig 3 below.

Power control register (PCON, address 0x4002 0000) bit description

Bit	Symbol	Value	Description	Reset value
2:0	PM		Power mode	000
		0x0	Default. The part is in active or sleep mode.	
		0x1	Deep-sleep mode. ARM WFI will enter Deep-sleep mode.	
		0x2	Power-down mode. ARM WFI will enter Power-down mode.	
		0x3	Deep power-down mode. ARM WFI will enter Deep-power down mode (ARM Cortex-M0+ core powered-down).	
3	NODPD		A 1 in this bit prevents entry to Deep power-down mode when 0x3 is written to the PM field above, the SLEEPDEEP bit is set, and a WFI is executed. This bit is cleared only by power-on reset, so writing a one to this bit locks the part in a mode in which Deep power-down mode is blocked.	0
7:4	-	-	Reserved. Do not write ones to this bit.	0
8	SLEEPFLAG		Sleep mode flag	0
		0	Active mode. Read: No power-down mode entered. Part is in Active mode. Write: No effect.	
		1	Low power mode. Read: Sleep, Deep-sleep or Power-down mode entered. Write: Writing a 1 clears the SLEEPFLAG bit to 0.	
10:9	-	-	Reserved. Do not write ones to this bit.	0
11	DPDFLAG		Deep power-down flag	0
		0	Not Deep power-down. Read: Deep power-down mode **not** entered. Write: No effect.	0
		1	Deep power-down. Read: Deep power-down mode entered. Write: Clear the Deep power-down flag.	
31:12	-	-	Reserved. Do not write ones to this bit.	0

Fig 3. Power Mode Control Register (PCON)

3.3 Wait For Interrupt (WFI) instruction

Execution of the WFI instruction will cause immediate entry to any of the low reduced power modes based on the SLEEPDEEP bit and PCON register settings mentioned above.

The WFI instruction is a Cortex-M0+ instruction which cannot be directly accessible by ANSI C. The CMSIS (Cortex Microcontroller Software Interface Standard) provides an intrinsic function to generate a WFI instruction and is supported by C compiler.

If a C compiler does not support the WFI intrinsic function, then the user will have to use assembly code to execute WFI instruction.

3.4 Programming steps to enter sleep mode

The following steps must be performed to enter sleep mode:

1. The PM bit in the PCON register (Fig 4) must be set to the default value 0x0.
2. The SLEEPDEEP bit in the ARM Cortex-M0+ SCR register must be set to zero.
3. Use the ARM Cortex-M0+ Wait-For-Interrupt (WFI) instruction.

Fig 4 below shows code example to enter sleep mode.

If using NXP's LPCOpen software platform, user can call the Chip_PMU_SleepState(LPC_PMU) function to enter sleep mode.

```
/*SLEEP MODE*/

/*Clear SLEEPDEEP BIT*/
 SCB->SCR &= ~(1<<2);

 /*Select sleep mode*/
  LPC_PMU->PCON = 0x0;

  /*Call ARM WFI function*/
  __WFI();
```

Fig 4. Code example (Sleep mode)

3.5 Wake-up from Sleep mode

Sleep mode is exited automatically when an interrupt enabled by the NVIC arrives at the processor or a reset occurs. Please see the LPC81x User Manual for details on which interrupt sources are connected to NVIC. After wake-up due to an interrupt, the microcontroller returns to its original power configuration defined by the contents of the PDRUNCFG and the SYSAHBCLKDIV registers. If a reset occurs, the microcontroller enters the default configuration in active mode.

3.6 Programming steps to enter Deep-sleep and Power-down modes

The following steps must be performed to either enter Deep-sleep mode or Power-down mode:

1. The PM bits in the PCON register (Fig 3) must be set to 0x1 for Deep-sleep mode or 0x2 for Power-down mode.
2. User can optionally select the power configuration in the PDSLEEPCFG register (Fig 5) to control the behavior of the Watchdog oscillator (WDTOSC) and the Brown-Out Detect (BOD) circuit when the device enters Deep-sleep or Power-down modes. The watchdog oscillator can be used as a clock source to the Windowed watchdog Timer (WWDT) to generate an interrupt or reset. The BOD can also be enabled to generate an interrupt or reset and can be used to protect the device from a low voltage event. Enabling the WDTOSC or the BOD circuit causes an additional current drain.

Deep-sleep configuration register (PDSLEEPCFG, address 0x4004 8230) bit description

Bit	Symbol	Value	Description	Reset value
2:0			Reserved.	0b111
3	BOD_PD		BOD power-down control for Deep-sleep and Power-down mode	1
		0	Powered	
		1	Powered down	
5:4			Reserved.	11
6	WDTOSC_PD		Watchdog oscillator power-down control for Deep-sleep and Power-down mode. Changing this bit to powered-down has no effect when the LOCK bit in the WWDT MOD register is set. In this case, the watchdog oscillator is always running.	1
		0	Powered	
		1	Powered down	
15:7	-		Reserved	0b111111111
31:16	-	-	Reserved	0

Fig 5. PDSLEEPCFG Register Settings

3. Ensure that the IRC is powered in the PDRUNCFG register and switch the system clock source (Fig 10) to the IRC in the MAINCLKSEL register before entering deep sleep or power-down modes.
4. Select the power configuration after wake-up in the PDAWAKECFG register. The bits in this register can be programmed to determine the state the chip must enter after it wakes up from deep-sleep or power-down modes.
5. Write one to the SLEEPDEEP bit in the ARM Cortex-M0+ SCR register.
6. Use the ARM Cortex-M0+ Wait-For-Interrupt (WFI) instruction.

Fig 6 and Fig 7 below show the code example to enter deep-sleep and power-down modes.

If using NXP's LPCOpen software platform, user can call the Chip_PMU_DeepSleepState(LPC_PMU) function to enter deep-sleep mode, and the Chip_PMU_PowerDownState(LPC_PMU) function to enter power-down mode.

```
SCB->SCR |= (1<<2);

/*Select deep-sleep mode*/
LPC_PMU->PCON = 0x1;

/*Call ARM WFI function*/
__WFI();
```

Fig 6. Code example (Deep-sleep mode)

```
SCB->SCR |= (1<<2);

/*Select power-down mode*/
LPC_PMU->PCON = 0x2;

/*Call ARM WFI function*/
__WFI();
```

Fig 7. Code example (Power-down mode)

3.7 Wake-up from Deep Sleep and Power-down modes

The microcontroller can wake up from Deep-sleep and Power-down modes using the wake-up sources mentioned below. Please see the LPC81x user manual for further details to configure the desired wake-up source.

Deep-sleep and Power-down	Pin interrupts	Enable pin interrupts in NVIC and STARTERP0 registers.
	BOD interrupt	• Enable interrupt in NVIC and STARTERP1 registers. • Enable interrupt in BODCTRL register. • BOD powered in PDSLEEPCFG register.
	BOD reset	• Enable reset in BODCTRL register. • BOD powered in PDSLEEPCFG register.
	WWDT interrupt	• Enable interrupt in NVIC and STARTERP1 registers. • WWDT running. Enable WWDT in WWDT MOD register and feed. • Enable interrupt in WWDT MOD register. • WDOsc powered in PDSLEEPCFG register.
	WWDT reset	• WWDT running. • Enable reset in WWDT MOD register. • WDOsc powered in PDSLEEPCFG register.
	Self Wake-up Timer (WKT) time-out	• Enable interrupt in NVIC and STARTERP1 registers. • Enable low-power oscillator in the DPDCTRL register in the PCON block. • Select low-power clock for WKT clock in the WKT CTRL register. • Start the WKT by writing a time-out value to the WKT COUNT register.
	Interrupt from USART/SPI/I2C peripheral	• Enable interrupt in NVIC and STARTERP1 registers. • Enable USART/I2C/SPI interrupts. • Provide an external clock signal to the peripheral. • Configure the USART in synchronous slave mode and I2C and SPI in slave mode.

Fig 8. Wake-up Sources (Deep-sleep and Power-down modes)

3.8 Programming steps to enter Deep power-down mode

The following steps must be performed to enter Deep power-down mode:

1. If using the WAKEUP pin as a wake-up source, pull the WAKEUP pin externally HIGH.
2. If using the self wake-up timer as a wake-up source, enable the low-power oscillator to run in Deep power-down mode by setting bits 2 and 3 in the Deep power-down register (DPDCTRL) register to 1.
3. Ensure that bit 3 in the PCON register (Fig 3) is cleared.
4. Write 0x3 to the PM bits in the PCON register (Fig 3).
5. User can optionally store data in the general purpose registers (GPREG0 to GPREG3) and in the DPDCTRL register (bits 31:4).
6. Write one to the SLEEPDEEP bit in the ARM Cortex-M0+ SCR register.
7. If using the self wake-up timer as a wake-up source, the low power oscillator needs to be enabled in the DPDCTRL (bits 2:3) and selected as a clock source for the self wake-up timer using WKT CTRL register. Start the self wake-up timer by writing to WKT COUNT register. Please see the LPC81x user manual (WKT chapter) for details regarding the WKT CTRL and COUNT registers.
8. Use the ARM Cortex-M0+ Wait-For-Interrupt (WFI) instruction.

Please note that the $\overline{\text{RESET}}$ /PIO0_5 pin needs to be always externally pulled HIGH to achieve desired Deep power-down current consumption.

Please refer to the provided software examples with this application note to see how deep power-down mode is configured.

3.9 Wake-up from Deep power-down modes

The microcontroller can wake up from deep power-down mode by using the PIO0_4/WAKEUP pin or using the self wake-up timer. Pulling the PIO0_4/WAKEUP pin LOW wakes up the LPC81x from Deep power-down mode. Please note that the if the self-wake-up timer is used as wake-up source, the $\overline{\text{WAKEUP}}$ functionality on the PIO0_4/WAKEUP pin can be disabled using the DPDCTRL register and this pin can then be used as a general purpose IO pin.

Deep power-down	WAKEUP pin PIO0_4	Enable the WAKEUP function in the DPDCTRL register in the PMU.
	WKT time-out	• Enable the low-power oscillator in the DPDCTRL register in the PMU. • Enable the low-power oscillator to keep running in Deep power-down mode in the DPDCTRL register in the PMU. • Select low-power clock for WKT clock in the WKT CTRL register. • Start WKT by writing a time-out value to the WKT COUNT register.

Fig 9. Wake-up Sources (deep power-down modes)

3.10 Additional tips to reduce power consumption

The current consumption can be further reduced by considering the following points:

3.10.1 CPU clock rate

The CPU clock rate can be controlled as needed, which allows a trade-off of power versus processing speed based on application requirements.

1. Changing clock sources: Internal Oscillator (12 MHz ± 1.5 %), System Oscillator (1 MHz to 25 MHz), Watchdog Oscillator (9.4 kHz to 2.3 MHz ± 40 %).
2. Reconfiguring PLL values, and/or lowering the PLL's output frequency (FCCO, 156 MHz to 320 MHz) can also save power.
3. An 8-bit system AHB clock divider register (SYSAHBCLKDIV) allows a range of options, including slowing CPU operation to a low rate for temporary power savings without turning off SYS PLL.

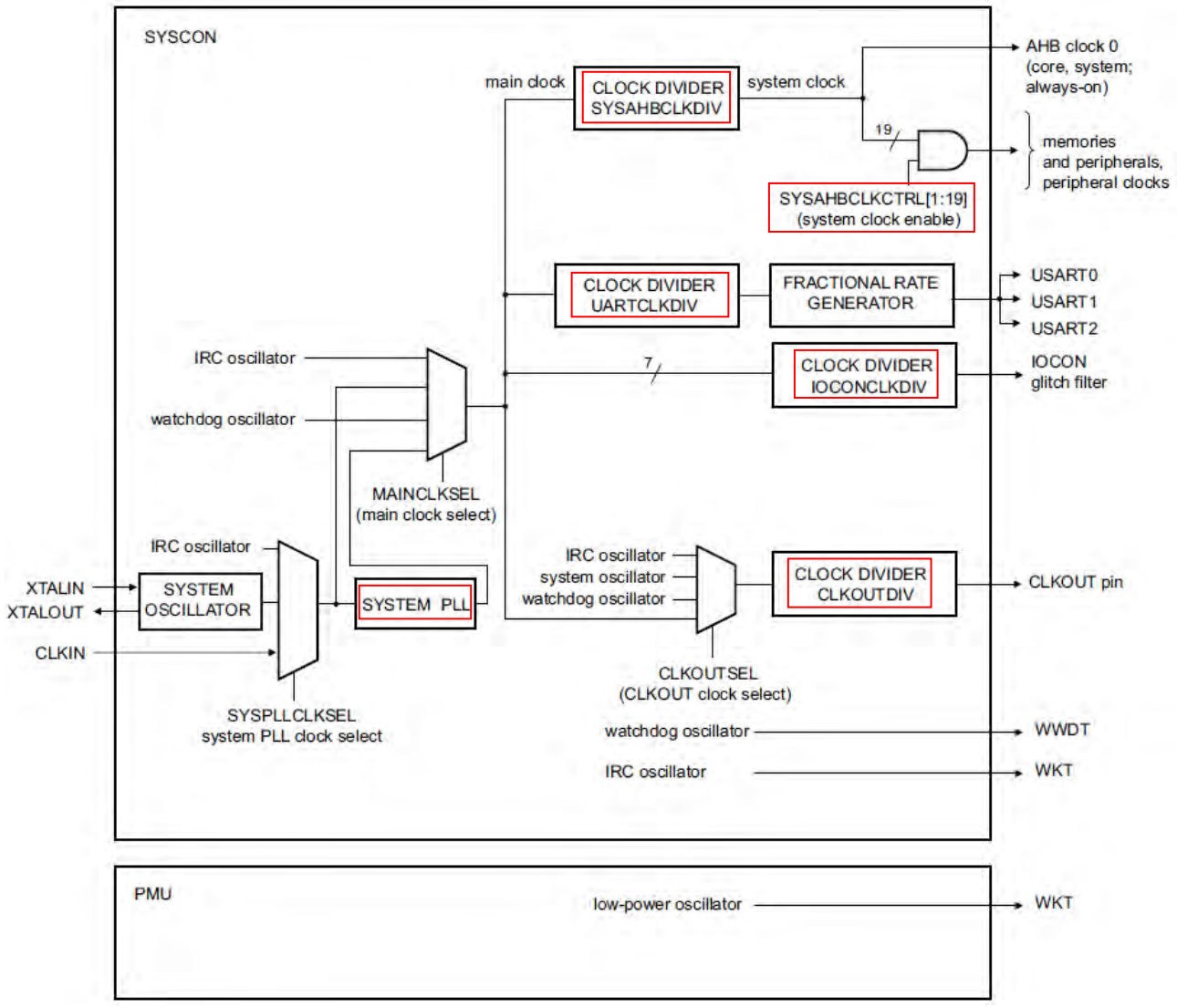


Fig 10. Clock Generation Unit (CGU) diagram

3.10.2 System AHB Clock Control Register (AHBCLKCTRL)

As shown in Fig 10, the SYSAHBCLKCTRL register enables the clocks to individual system and peripheral blocks. Depending upon application requirements, the user can use this register to reduce power by disabling clocks to unused peripherals. Please see the LPC81xM User Manual for details.

3.10.3 Clock dividers

As shown in Fig 10, depending on application requirements, the user can use the divider registers (UARTCLKDIV, IOCONCLKDIV, CLKOUTDIV) to reduce the peripheral clock frequency or shut down the clock to reduce power. Please see the LPC81x User Manual for details.

3.10.4 Power-down Configuration register (PDRUNCFG)

PDRUNCFG Register allows the user to power on or off individual analog blocks, allowing elimination of power consumption by analog peripherals that are not needed. Please see the LPC81xM User Manual for details.

3.10.5 Miscellaneous

3.10.5.1 Software

Most embedded applications terminate with a while(1) loop, and they service interrupts whenever needed. In this case, code is still constantly fetched from the on-chip flash and executed which adds to the power consumption. A better solution would be to switch to the sleep power saving mode and then wait for interrupts. An interrupt from a peripheral would then wake the device from Sleep mode. Considerable power savings can be achieved by keeping the core in Sleep mode while it is waiting for interrupts.

3.10.5.2 Port pins

The general purpose port pins (except the I2C pins) on the LPC81x have programmable internal pull-ups enabled by default. Additional steps should be considered to reduce current consumption in the low power modes by configuring the port pins appropriately to prevent contention. It is recommended to disable all internal pull-ups and internal pull-downs, configure the IOs as GPIO outputs, and depending upon what is externally connected to the port pin, drive the pins as output low or output high.

Please note that in deep power-down mode, state of the port pins does not affect the current consumption and the steps mentioned above do not need to be considered in this mode.

3.10.6 I2C pins

When not using the I2C peripheral, the I2C pins (PIO0_10 and PIO0_11) needs to be appropriately terminated either via software or externally using pull-up or pull-down resistor.

3.10.7 RESET_N/PIO0_5 pin

The $\overline{\text{RESET}}$ /PIO0_5 pin needs to be externally pulled up via 10k to 47k resistor when in deep power-down mode. If the pin is left floating, user will see an increase in deep power-down current consumption.

3.10.8 Packages (Un-bonded pins)

The LPC81x parts are available in various packages options. For GPIO pins which are not available on smaller packages, these non-bonded out pins should be defined in a well-defined state using the steps mentioned in section 3.10.5.2.

3.10.9 Debug notes

The user should be aware of certain limitations during debugging. The most important is that, due to limitations of the Cortex-M0+ integration, the LPC81x cannot wake up in the usual manner from Sleep, Deep Sleep, or Deep power-down modes. It is recommended not to use these modes during debug. Once an application is downloaded via SWD interface, the USB to SWD debug adapter should be removed from the target board. Another issue is that debug mode changes the way in which reduced power modes are handled by the Cortex-M0+ CPU. This causes power modes at the device level to be different from normal modes operation. These differences mean that power measurements should not be made while debugging; the results will be higher than during normal operation in an application.

4. Low power mode demos

4.1 Objective

This application note provides three low power mode examples and is based on NXP's LPCOpen software platform. The examples support all three IDE tools (LPCXpresso, Keil, and IAR).

1. periph_low_power: Low Power Modes (sleep, deep sleep, power-down)

This example allows the user to enter the low power modes and wake-up using general purpose port pin.

2. periph_dpd_wkt: Using the self wake-up timer and deep power-down mode

This example puts the device in deep power-down mode and automatically wakes up using the self wake-up timer.

3. periph_dpd: Using the wake-up pin and deep power-down mode

This example allows the user to enter deep power-down mode and wake-up using the WAKEUP pin.

To reprogram the device, the user must put the device in ISP mode by pulling the PIO0_12 pin low and power cycling the board. Thereafter, the device can be erased and programmed again.

4.2 Requirements

1. LPCXpresso IDE or Keil IDE or IAR IDE
2. LPCXpresso Target Board (LPC800) (see Fig 11).

4.3 LPCXpresso target board (LPC800)

4.3.1 To measure the current

The LPCXpresso LPC812 Board requires the following set-up to measure the current:

1. The LPCXpresso LPC812 board includes an integrated SWD debugger (LPC-Link). By default, the power is shared between the LPC800 and LPC-Link. Trace connection between J4.1 and J4.2 needs to be disconnected by unsoldering the solder bump to separate the power supply. Thereafter, the user can connect an ammeter between J4.1 and J4.2 to measure the current. See Fig 11 below.
2. The trimming potentiometer needs to be de-soldered to achieve the correct current consumption. See Fig 11 below.

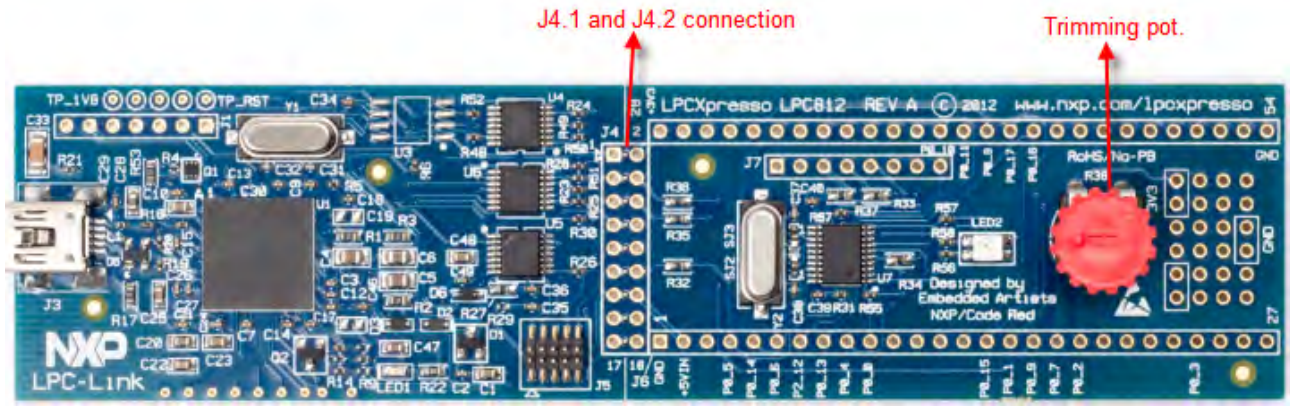


Fig 11. LPCXpresso LPC812 board

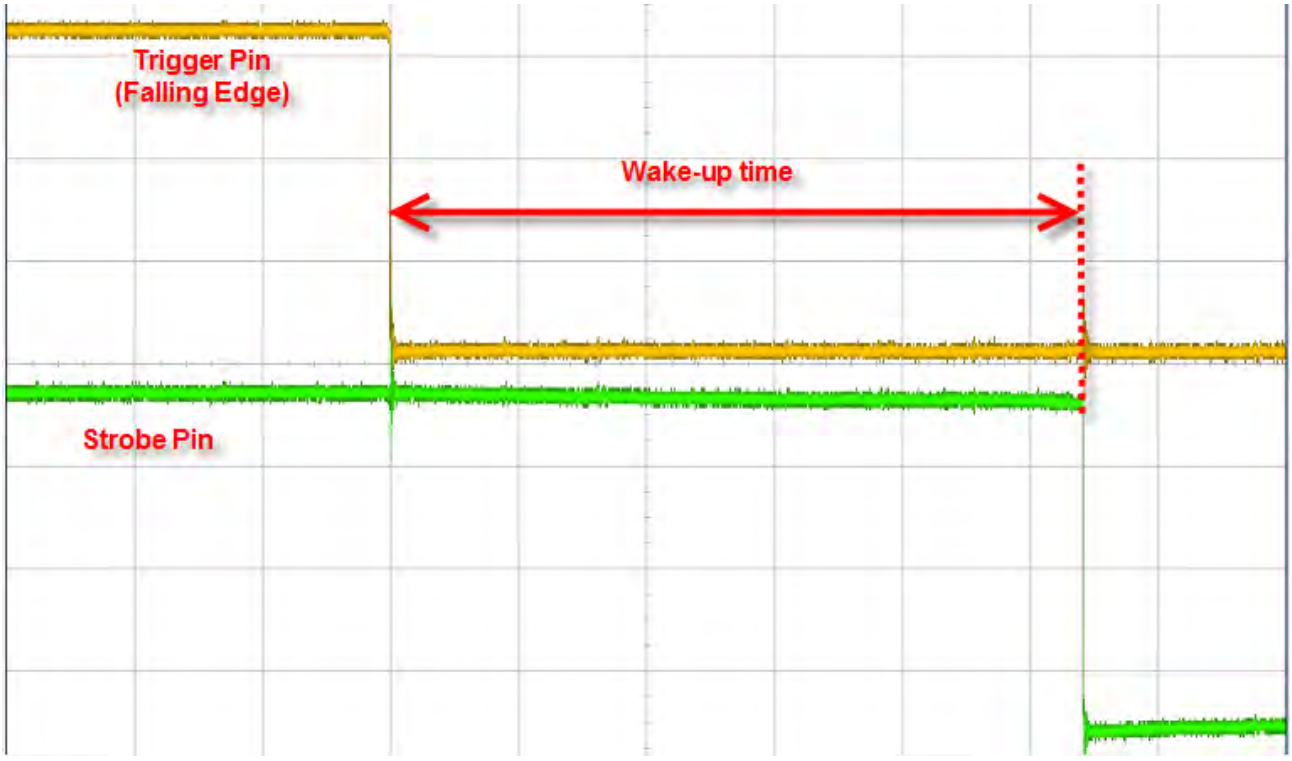
4.3.2 periph_low_power example

In active mode, the LPC800 is running at 12 MHz (Internal RC Oscillator). LED (on port pin PIO0_7) is ON in active mode.

1. Connect the USB cable to power up the LPC812 LPCXpresso board.
2. Download the code to flash.
3. Disconnect/connect the USB cable to power cycle the LPC812 LPCXpresso board.
4. The low power modes can be entered as follows:
   a. Sleep Mode – Externally pull port pin PIO0_15 low
   b. Deep-Sleep Mode – Externally pull port pin PIO0_13 low
   c. Power-down Mode – Externally pull port pin PIO0_14 low

LED (on port pin PIO0_7) is OFF in the low power modes.

5. Externally pull-down port pin PIO0_4 low to wake the LPC800 from the low power modes. LED (on port pin PIO0_7) is ON upon wake-up.
6. The wake-up times can be measured by using the following pins:
   a. Trigger pin (PIO0_4) - Used to get the device out of the low power modes (sleep, deep-sleep, power-down). The pin is triggered externally low (falling edge) to wake the device up.
   b. Strobe pin (PIO0_7) – After wake-up, the device returns into run mode, and this pin is set low within the WAKEUP_IRQHANDLER() subroutine.
7. As shown in Fig 12, the wakeup time is the difference between the falling edge of the trigger pin (yellow waveform) and the rising edge of the Strobe pin (green waveform). Please see table for results.



See Table 1 for wake-up times.

Fig 12. Wake-up time measurements (sleep, deep-sleep, power-down)

4.3.3 periph_dpd_wkt example

In active mode, the LPC800 is running at 12 MHz (Internal RC Oscillator). LED (on port pin PIO0_7) is ON in active mode.

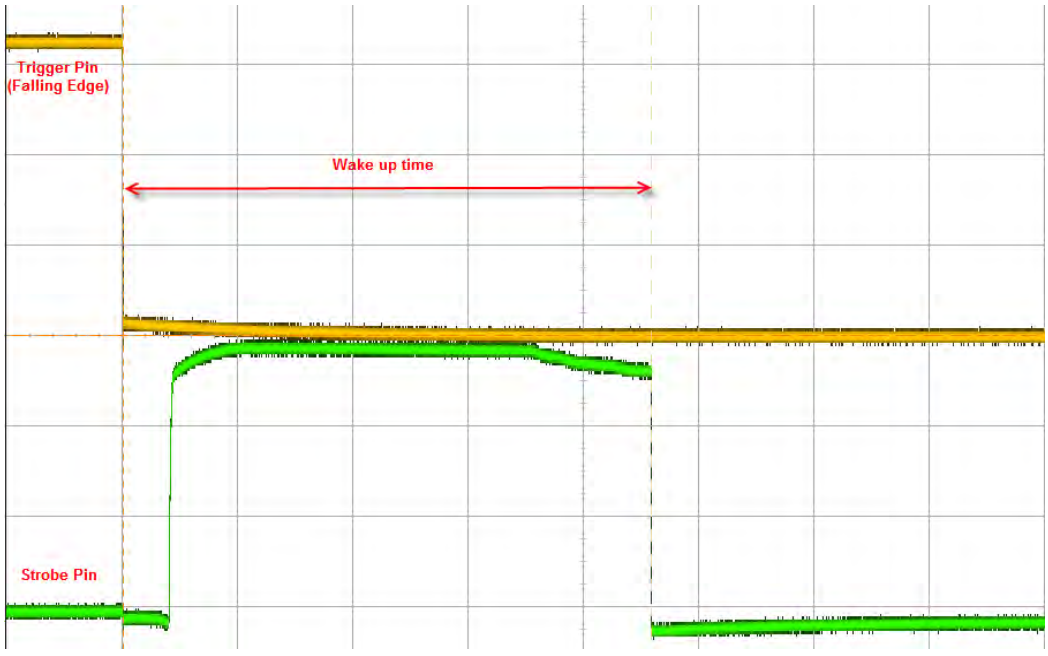
The low power oscillator is used as a clock source to the self wake-up timer.

1. Connect the USB cable to power up the LPC812 LPCXpresso board.
2. Download the code to flash
3. Disconnect/connect the USB cable to power cycle the LPC812 LPCXpresso board.
4. The device immediately enters deep power-down mode, and automatically wakes-up using the self wake-up timer. Upon wake-up, port pin PIO0_7 toggles.
5. The wake-up time for this exercise is not measured since the variation could vary due to ± 40 % spread of the low power oscillator.

4.3.4 periph_dpd example

In active mode, the LPC800 is running at 12 MHz (Internal RC Oscillator).

1. Externally pull port pin PIO0_4/WAKE-UP high to 3.3 V.
2. Connect the USB cable to power up the LPC812 LPCXpresso board.
3. Download the code to flash.
4. Disconnect/connect the USB cable to power cycle the LPC812 LPCXpresso board.
5. The device immediately enters deep power-down mode.
6. Externally pull-down port pin PIO0_4 low to wake the LPC800. Upon wake-up, port pin PIO0_7 toggles.
7. The wake-up time for this exercise can only be measured using the LPCXpresso example by using the following pins:
   a. Trigger pin (PIO0_4) - Used to get the device out of the deep power-down mode. The pin is triggered externally low (falling edge) to wake the device up.
   b. Strobe pin (PIO0_13) – After wake-up, the device goes through a reset process where this pin will be initially pulled up due to the internal pull-up, and thereafter, gets set low within the RESET_IRQHANDLER() subroutine.
8. As shown in Fig 13, the wakeup time is the difference between the falling edge of the trigger pin (yellow waveform) and the falling edge of the Strobe pin (green waveform).



See Table 1 for wake-up times.

Fig 13. Wake-up time measurement (deep power-down mode)

4.3.5 Power measurements and Wake-up measurements

Table 1 shows the typical power consumption and wake-up measurements:

Table 1. Typical power consumption (3.3 V, Temp = 25 °C)

Low power modes	Ivdd current	Wake-up time
Sleep mode (IRC (12 MHz), all peripherals off)	0.7 mA	2.6 μs
Deep-sleep mode (Watchdog Osc off and BOD off)	150 μA	4 μs
Power-down Mode (Watchdog Osc off and BOD off)	0.9 μA	50 μs
Deep Power-down Mode (note 1) (low power oscillator and self wake-up time disabled)	170 nA	215 μs
Deep Power-down Mode (low power oscillator and self wake-up time enabled)	1 μA	-

5. Conclusion

The LPC800 series provides great flexibility and various options for users to achieve low power consumption and wake-up flexibility. As shown in Table 1, the user has various low power mode options to choose from in order to achieve the desired power consumption by trading up or down wake-up time of the device. This flexibility allows a trade-off between power consumption and wake-up speed based on the user's application requirements. In addition, user has the option of using the self wake-up timer feature and would not require an external component to wake the device up.

6. Legal information

6.1 Definitions

Draft — The document is a draft version only. The content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included herein and shall have no liability for the consequences of use of such information.

6.2 Disclaimers

Limited warranty and liability — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information.

In no event shall NXP Semiconductors be liable for any indirect, incidental, punitive, special or consequential damages (including - without limitation - lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory.

Notwithstanding any damages that customer might incur for any reason whatsoever, NXP Semiconductors' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms and conditions of commercial sale of NXP Semiconductors.

Right to make changes — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Suitability for use — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors accepts no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

Applications — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

Customers are responsible for the design and operation of their applications and products using NXP Semiconductors products, and NXP Semiconductors accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the NXP Semiconductors product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP Semiconductors does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary testing for the customer's applications and products using NXP Semiconductors products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). NXP does not accept any liability in this respect.

Export control — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from national authorities.

6.3 Trademarks

Notice: All referenced brands, product names, service names and trademarks are property of their respective owners.

7. Contents

1. Introduction 3
2. Low power modes 4
2.1.1 Sleep mode 4
2.1.2 Deep-sleep mode 4
2.1.3 Power-down mode 4
2.1.4 Deep power-down mode 5
3. Entering the low power modes and wake-up implementation 5
3.1 System Control Register (SCR) 5
3.2 Power Control Register (PCON) 6
3.3 Wait For Interrupt (WFI) instruction 7
3.4 Programming steps to enter sleep mode 8
3.5 Wake-up from Sleep mode 8
3.6 Programming steps to enter Deep-sleep and Power-down modes 8
3.7 Wake-up from Deep Sleep and Power-down modes 10
3.8 Programming steps to enter Deep power-down mode 11
3.9 Wake-up from Deep power-down modes 12
4.1 Additional tips to reduce power consumption 12
4.1.1 CPU clock rate 12
4.1.2 System AHB Clock Control Register (AHBCLKCTRL) 13
4.1.3 Clock dividers 13
4.1.4 Power-down Configuration register (PDRUNCFG) 14
4.1.5 Miscellaneous 14
4.1.5.1 Software 14
4.1.5.2 Port pins 14
4.1.6 I2C pins 14
4.1.7 RESET_N/PIO0_5 pin 14
4.1.8 Packages (Un-bonded pins) 14
4.1.9 Debug notes 14
5. Low power mode demos 15
5.1 Objective 15
5.2 Requirements 15
5.3 LPCXpresso target board (LPC800) 15
5.3.1 To measure the current 15
5.3.2 periph_low_power example 16
5.3.3 periph_dpd_wkt example 17
5.3.4 periph_dpd example 18
5.3.5 Power measurements and Wake-Up measurements 19
6. Conclusion 19
7. Legal information 20
7.1 Definitions 20
7.2 Disclaimers 20
7.3 Trademarks 20
8. Contents 21

For more information, please visit: http://www.nxp.com
For sales office addresses, please send an please send an email to: salesaddresses@nxp.com

Date of release: 4 August 2014
Document identifier: AN11561