AN11799

LPC5411x Low Power Modes and Wake-up Time

Rev. 1.1 — 18 July 2016 Application note

Document information

Info	Content
Keywords	LPC5411x Low Power Modes, Wake-up time, Cortex-M4, Cortex-M0+
Abstract	This application note introduces the low power modes of the LPC5411x series and wake-up implementation. The application note also provides software examples to enter the low power modes and demonstrates how to measure the power consumption and wake-up times using the LPC54114 LPCXpresso board.

Revision history

Rev	Date	Description
1.1	20160718	Updated Table 11: deep-sleep values for power Vdd = 3.3 V and 1.8 V
1	20160323	Initial version

Contact information

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

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

1. LPC5411x Introduction

The LPC5411x series MCU are dual-core microcontrollers for embedded applications featuring multiple communication interfaces with very low power consumption in active and low power modes. The LPC54114 parts have a Cortex-M4 and a Cortex M0+ core included. The LPC54113 and LPC54111 parts only have one Cortex-M4 core. Both the cores can operate at frequencies of up to 100 MHz.

The LPC5411x includes up to 256kB of on-chip flash memory and up to 192kB of on-chip SRAM. The peripheral complement includes Full Speed USB device interface, a DMIC subsystem with dual-channel PDM microphone interface and I2S, SPI flash interface, five general-purpose timers, one versatile timer with PWM and many other capabilities (SCTimer/PWM), one RTC/alarm timer, one 24-bit Multi-Rate Timer (MRT), a Windowed Watchdog Timer (WWDT), eight flexible serial communication peripherals (each of which can be a USART, SPIs, or I2C interface), and one 12-bit 5.0 Msamples/sec ADC, and a temperature sensor.

This application note introduces the various low power modes of the LPC5411x series, the steps required to enter the low power modes and wake-up implementation. This application note also provides software examples to enter and wake-up from the low power modes.

This application note covers the following topics:

1. Low power modes.
2. Wake-Up implementation.
3. Low power mode demo.

2. LPC5411x low power modes introduction

On the LPC5411x series, there are three reduced power modes: sleep, deep-sleep, and deep power-down.

In addition, there are several features unique to the LPC5411x family that allow better power consumption control and faster wake-up times. The following sections cover the power saving modes for the LPC5411x.

Section 2.4 introduces the clock gating feature of the SRAM blocks on LPC5411x which can be applied to sleep and deep-sleep modes. This feature is particularly helpful to manage the SRAM retention and low power mode current consumption optimization.

This chapter ends with a summary of the peripheral states when the part enters a particular low power mode.

2.1 Sleep mode

In sleep mode, the system clock to the CPU is stopped and execution of instructions is suspended until either a reset or an interrupt occurs.

Peripheral functions, if selected to be clocked in the AHBCLKCTRL registers, 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 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.

2.2 Deep-sleep mode

In deep-sleep mode, the system clock to the processor is disabled as in sleep mode. The main clock and therefore all peripheral clocks are disabled. Most analog blocks are powered down by default (for example, the FRO and the ADC) but can be selected to keep running through the power API if needed as wake-up sources. See Section 3.4 for details on which analog peripherals are turned off in deep-sleep mode.

The deep-sleep mode eliminates all dynamic power used by the processor itself, memory systems and related controllers, and internal buses. The processor state and registers, peripheral registers, and 64 KB of SRAM section is retained by default and the logic levels of the pins remain static. Sections of SRAM retention can be defined through the power API. Also, the flash is powered down for deep-sleep mode. The user can also optionally power on more SRAM blocks in their application for better data retention or switch to different SRAMs for data retention. Wake-up times from deep-sleep mode are longer compared to the sleep mode.

2.2.1 Wake-up to SRAM from deep-sleep mode

The flash can remain powered down during wake-up from deep-sleep mode. The wake-up routine will wake to execute from SRAM avoiding the flash start-up time, drastically reducing the wake-up time.

2.3 Deep power-down mode

For maximal power savings, the entire system is shut down except for the Power Management Unit (PMU) and the RTC domain. Only the general purpose registers in the PMU and the RTC registers are powered and can maintain their internal states. The LPC5411x does not have a dedicated wake-up pin. The part can wake up on a pulse on the reset pin or on an interrupt from the RTC alarm. On wake-up, the part reboots.

2.4 Individually clocked SRAM blocks

The clock gating feature implemented on the three SRAM blocks gives more granularities in the control of the low power mode power consumption. Table 1 shows where the four sections of the SRAM reside in the LPC5411x MCU memory map. All SRAMs have their own power switches.

Table 1. SRAM memory layout

	SRAM0	SRAM1	SRAM2	SRAMX
Size	Up to 64 KB	Up to 64 KB	Up to 32 KB	Up to 32 KB
Address range	Begins at 0x2000 0000	If present, begins at 0x2001 0000	If present, begins at 0x2002 0000	Begins at 0x0400 0000

Table 2 shows four switches in the PDRUNCFG register that allow each of the four individual SRAM sections to be retained.

Table 2. Power configuration register (PDRUNCFG0, main syscon: offset 0x610) bit description

Bit	Symbol	Description	Reset value
3:0	-	Reserved	-
4	PDEN_FRO	FRO oscillator. 0 = Powered; 1 = Powered down.	0
5	PD_FLASH	Part of flash power control. Only certain combinations of flash power configuration bits may be used.	0
6	PDEN_TS	Temp sensor. 0 = Powered; 1 = Powered down.	1
7	PDEN_BOD_RST	Brown-out Detect reset. 0 = Powered; 1 = Powered down.	0
8	PDEN_BOD_INTR	Brown-out Detect interrupt. 0 = Powered; 1 = Powered down.	1
9	-	Reserved	-
10	PDEN_ADC0	ADC0. 0 = Powered; 1 = Powered down.	1
11	PD_VDDFLASH	Part of flash power control. Only certain combinations of flash power configuration bits may be used.	0
12	LP_VDDFLASH	Part of flash power control. Only certain combinations of flash power configuration bits may be used.	0
13	PDEN_SRAM0	SRAM0. 0 = Powered; 1 = Powered down.	0
14	PDEN_SRAM1	SRAM1. 0 = Powered; 1 = Powered down.	0
15	PDEN_SRAM2	SRAM2. 0 = Powered; 1 = Powered down.	0
16	PDEN_SRAMX	SRAMX. 0 = Powered; 1 = Powered down.	0
17	PDEN_ROM	ROM. 0 = Powered; 1 = Powered down.	0
18	PD_VDDHV_ENA	Part of flash power control. Only certain combinations of flash power configuration bits may be used.	-
19	PDEN_VDDA	Vdda to the ADC, must be enabled for the ADC to work. Also see bit 23. 0 = Powered; 1 = Powered down.	1
20	PDEN_WDT_OSC	Watchdog oscillator. 0 = Powered; 1 = Powered down.	1
21	PDEN_USB_PHY	USB pin interface. 0 = Powered; 1 = Powered down.	1
22	PDEN_SYS_PLL	PLL0. 0 = Powered; 1 = Powered down.	1
23	PDEN_VREFP	Vrefp to the ADC, must be enabled for the ADC to work. Also see bit 19. 0 = Powered; 1 = Powered down.	1
24	-	Reserved	-
25	PD_FLASH_BG	Part of flash power control. Only certain combinations of flash power configuration bits may be used.	1
31:26	-	Reserved	-

2.5 Low power modes summary

Table 3 summarizes the parts of the chip that are affected by the various power saving modes.

Table 3. Peripheral states in reduced power modes

Core/Peripheral States	Sleep	Deep-sleep	Deep power-down
FRO	Configurable[1]	Configurable[2]	Off
Flash	Configurable[1]	Off	Off
BOD	Configurable[1]	Configurable[2]	Off
PLL	Configurable[1]	Off	Off
WD osc/WWDT	Configurable[1]	Configurable[2]	Off
USART	Configurable[1]	Off; but can create a wake-up interrupt in synchronous slave mode or 32 kHz clock mode	Off
SPI	Configurable[1]	Off; but can create a wake-up interrupt in slave mode	Off
I2C	Configurable[1]	Off; but can create a wake-up interrupt in slave mode	Off
RTC oscillator	Configurable[1]	Configurable[2]	Configurable[3]
Other Digital peripherals	Configurable[1]	Off	Off
Other Analog peripherals	Configurable[1]	Configurable[2]	Off

Notes for the lower modes comparison in Table 3.

1. As an additional power saving measure in active and sleep mode, the power to these components can be switched on/off via the Power Configuration Register (PDRUNCFG).

2. While the analog power is removed from most peripheral components in deep-sleep mode, most analog and digital blocks can be configured to provide a wake-up service for the CPU. See section 3.4.2 for more details.

3. The RTC oscillator can be used to wake-up the CPU via interrupt from sleep, deep-sleep, and deep power-down mode to provide a timed reset for the CPU from deep power-down mode.

3. Entering low power modes and waking up

Power to the part is supplied via two power domains. The main power domain is powered by VDD and supplies power to the core, peripheral, memories, inputs and outputs via an on-chip regulator.
A second, always-on power domain is also powered by VDD, and includes the RTC and wake-up timer. This domain always has power as long as sufficient voltage is supplied to VDD.
Power usage is controlled by settings in register within the SYSCON block, regulator settings controlled via a Power API, and the operating mode of a CPU. This application note describes how to enter and wake up from the various low power modes using the ROM power APIs.

3.1 Power control API

The Power APIs provide functions to configure the system clock and set up the system for expected performance requirements. See Table 4.

Table 4. Power API ROM calls

Function prototype	API description
`uint32_t Chip_POWER_SetPLL (uint32_t multiply_by, uint32_t input_freq);`	Power API PLL configuration routine. This API sets up basic PLL operation.
`uint32_t Chip_POWER_SetVoltage (uint32_t desired_freq);`	Power API internal voltage configuration routine.This API configures the internal regulator for the desired active operating mode and frequency. Also sets up corresponding flash wait states.
`void Chip_POWER_EnterPowerMode (POWER_MODE_T mode, uint32_t peripheral_ctrl);`	Power API power mode configuration routine.This API prepares the chip for reduced power modes: sleep, deep-sleep, or deep power-down mode. Also allows selection of which peripherals are kept alive in the reduced mode, and can therefore wake up the device from that mode.

3.1.1 Chip_POWER_EnterPowerMode

The Chip_POWER_EnterPowerMode() API prepares the part, then enters any of the low power modes. Specifically for the deep-sleep mode, the API function configures which analog components remain running in those two modes, so that an interrupt from one of the analog peripherals can wake up the part.

The Chip_POWER_EnterPowerMode() routine in Table 5 can also configure the analog peripherals for the low power modes by passing the peripheral name as the second parameter.

Table 5. Chip_POWER_EnterPowerMode routine

Routine	Chip_POWER_EnterPowerMode
Prototype	`void Chip_POWER_EnterPowerMode(POWER_MODE_T mode, uint32_t peripheral_ctrl);`
Input parameter	Param0: mode (see Fig 1.) Param1: peripheral
Result	None
Description	Defines the low power mode (either sleep, deep-sleep, or deep power-down modes) and allows controlling which peripherals are powered up in the reduced power mode.

NOTE: In addition to the analog peripherals listed with this parameter, the serial peripherals can also wake-up the chip from deep-sleep mode on an interrupt triggered by an incoming signal. This wake-up scenario is not configured using the Chip_POWER_EnterPowerMode() API. For details, please refer to the user manual chapters for USART, SPI and I2C.

3.1.2 Chip_POWER_EnterPowerMode input parameters

Param0: mode

The mode parameter defines the low power mode and prepares the chip to enter the selected mode. The following modes are valid:

```
#define POWER_SLEEP                0
#define POWER_DEEP_SLEEP           1
#define POWER_DEEP_POWER_DOWN      2
```

Fig 1. Low power modes

Param1: peripheral

The peripheral parameter is a 32-bit value that corresponds to the PDRUNCFG register (as shown in Table 2). If sleep mode (*POWER_SLEEP*) is selected with the mode parameter, the peripheral parameter is ignored.

The peripheral parameter defines which analog peripherals can wake-up the chip from sleep, deep-sleep mode. Some of these analog peripherals can wake-up the chip from deep-sleep, or deep power-down mode. For example, the RTC can be selected to remain running thus the RTC alarm wake-up the part from sleep, deep-sleep or deep power-down mode.

The LPCOpen software package defines below macros to allow easy selection of which analog peripheral to be on at deep-sleep mode.

```
/* Power control definition bits (0 = powered, 1 = powered down) */
#define SYSCON_PDRUNCFG_PD_FRO          (1 << 4)        /*!< FRO oscillator */
#define SYSCON_PDRUNCFG_PD_FLASH        (1 << 5)      /*!< Flash memory */
#define SYSCON_PDRUNCFG_PD_TS           (1 << 6)     /*!< Temperature Sensor */
#define SYSCON_PDRUNCFG_PD_BOD_RST      (1 << 7)       /*!< Brown-out Detect reset */
#define SYSCON_PDRUNCFG_PD_BOD_INTR     (1 << 8)       /*!< Brown-out Detect interrupt
*/
#define SYSCON_PDRUNCFG_PD_ADC0         (1 << 10)     /*!< ADC0 */
#define SYSCON_PDRUNCFG_PD_SRAM0        (1 << 13)       /*!< SRAM0 */
#define SYSCON_PDRUNCFG_PD_SRAM1        (1 << 14)       /*!< SRAM1 */
#define SYSCON_PDRUNCFG_PD_SRAM2        (1 << 15)       /*!< SRAM2 */
#define SYSCON_PDRUNCFG_PD_SRAMX        (1 << 16)       /*!< SRAMX */
#define SYSCON_PDRUNCFG_PD_ROM          (1 << 17)     /*!< ROM */
#define SYSCON_PDRUNCFG_PD_VDDA_ENA     (1 << 19)         /*!< Vdda to the ADC, must
be enabled for the ADC to work */
#define SYSCON_PDRUNCFG_PD_WDT_OSC      (1 << 20)     /*!< Watchdog oscillator */
#define SYSCON_PDRUNCFG_PD_USB_PHY      (1 << 21)     /*!< USB Phy */
#define SYSCON_PDRUNCFG_PD_SYS_PLL      (1 << 22)       /*!< PLL0 */
#define SYSCON_PDRUNCFG_PD_VREFP        (1 << 23)       /*!< Vrefp to the ADC, must be
enabled for the ADC to work */
```

Fig 2. PDRUNCFG register bit macro in LPCOpen package

Changing the contents of PDRUNCFG register should only be completed by writing to the PDRUNCFGSET and/or PDRUNCFGCLR register. The LPCOpen package has API calls to take care of this procedure. See Fig 3.

```
Chip_SYSCON_PowerDown(SYSCON_PDRUNCFG_PD_VREFP);
```

Fig 3. Power on/off analog blocks with LPCOpen package

3.2 Switch to FRO 12MHz clocking before entering low power modes

Before entering any of the four low power modes, the application code should switch to 12 MHz FRO unless it is already set as a source. On wake-up from sleep and deep-sleep modes, the application code should recover the intended clock source. Fig 4 shows a sample implementation using the LPCOpen package.

```
/* save the clock source, power down the PLL */
    saved_clksrc = Chip_Clock_GetMainClockSource();


     /* Disable PLL, if previously enabled, prior to sleep */
     if (saved_clksrc == SYSCON_MAINCLKSRC_PLLOUT) {
         Chip_Clock_SetMainClockSource(SYSCON_MAINCLKSRC_FRO12MHZ);
         Chip_SYSCON_PowerDown(SYSCON_PDRUNCFG_PD_SYS_PLL);
     }
     else if (saved_clksrc == SYSCON_MAINCLKSRC_FROHF) {
         saved_clkRate = Chip_Clock_GetFROHFRate();
         Chip_Clock_SetMainClockSource(SYSCON_MAINCLKSRC_FRO12MHZ);
         LPC_SYSCON->FROCTRL &= ~(SYSCON_FROCTRL_MASK |
SYSCON_FROCTRL_HSPDCLK);
         Chip_SYSCON_SetFLASHAccess(SYSCON_FLASH_1CYCLE);
     }
```

Fig 4. Switch to FRO clocking and recover saved system clock

The following sections assumes that the application code is switched to 12 MHz FRO.

3.3 Entering and waking up from sleep mode

3.3.1 Entering sleep mode

Based on the Chip_POWER_EnterPowerMode API, the sleep mode can be easily entered.

```
Chip_POWER_EnterPowerMode(curr_pwr,
                          (SYSCON_PDRUNCFG_PD_SRAM0 |
                          SYSCON_PDRUNCFG_PD_SRAM1 |
                          SYSCON_PDRUNCFG_PD_SRAM2 |
                          SYSCON_PDRUNCFG_PD_SRAMX ));
```

Fig 5. Entering sleep mode

3.3.2 Waking up from sleep mode

Any interrupt can wake-up the part from sleep mode. In this application note, the following three methods are used to wake-up the part from sleep mode.

- RTC interrupt
- uTICK interrupt
- PININT

3.4 Entering and waking up from deep-sleep mode

3.4.1 Entering deep-sleep mode

Chip_POWER_EnterPowerMode() API in Fig 3 can be used to enter the deep-sleep mode. LPCOpen also provides another layer of API with the same argument inputs as the Chip_POWER_EnterPowerMode() API. See Fig 6.

NOTE: To enter deep-sleep mode with default configuration, enter second parameter as 0 in the function displayed in Fig 6.

Depending on the application, certain peripherals, for example the SRAM0, can be kept active by supplying the bit position to the power API as illustrated in Fig 6.

```
Chip_POWER_EnterPowerMode(curr_pwr,
                          SYSCON_PDRUNCFG_PD_SRAM0);
```

Fig 6. Entering deep-sleep mode

The analog peripherals shown in Table 6 are turned off when the deep-sleep mode is entered with the second argument of Chip_POWER_EnterPowerMode() set to 0.

Table 6. Analog peripherals that default to OFF in deep-sleep mode

Symbol in the PDRUNCFG register	Description
SYSCON_PDRUNCFG_PD_FRO	FRO oscillator
SYSCON_PDRUNCFG_PD_TS	Temperature Sensor
SYSCON_PDRUNCFG_PD_BOD_INTR	Brown-out Detect interrupt
SYSCON_PDRUNCFG_PD_ADC0	ADC0
SYSCON_PDRUNCFG_PD_ROM	ROM
SYSCON_PDRUNCFG_PD_VDDA_ENA	Vdda to the ADC, must be enabled for the ADC to work
SYSCON_PDRUNCFG_PD_USB_PHY	USB Phy
SYSCON_PDRUNCFG_PD_SYS_PLL	PLL0
SYSCON_PDRUNCFG_PD_VREFP	Vrefp to the ADC, must be enabled for the ADC to work
SYSCON_PDRUNCFG_PD_FLASH	Flash memory
SYSCON_PDRUNCFG_PD_BOD_RST	Brown-out Detect reset
SYSCON_PDRUNCFG_PD_SRAM1	SRAM1

Symbol in the PDRUNCFG register	Description
SYSCON_PDRUNCFG_PD_SRAM2	SRAM2
SYSCON_PDRUNCFG_PD_SRAMX	SRAMX
SYSCON_PDRUNCFG_PD_WDT_OSC	Watchdog oscillator

3.4.2 Waking up from deep-sleep mode

The STARTERP0 and STARTERP1 registers enable an interrupt for wake-up from deep-sleep mode. The particular interrupt also needs to be enabled in the NVIC domain.

Some interrupts are typically used in sleep mode only and will not occur during deep-sleep modes because relevant clocks are stopped. However, it is possible to enable those clocks (significantly increasing power consumption in the reduced power mode), making these wake-up sources possible.

Whether peripheral interrupts can occur during deep-sleep mode depends on the peripheral, its configuration, and system set-up.

The general procedure to configure waking up from deep-sleep mode is described in Table 7.

Table 7. Configure wake-up from deep-sleep mode

• Setup the wake-up peripheral source
• Enable the corresponding interrupt in the NVIC
• Enable the corresponding interrupt in the STARTERP0 or STARTERP1 register
• Enable the corresponding analog block if needed with the PDRUNCFG register
• Enable the interrupt at NVIC as well – do not forget

Table 8. Start enable register 0 (STARTER0, main syscon: offset 0x680) bit description

Bit	Symbol	Description	Reset Value
0	WDT, BOD	WWDT and BOD interrupt wake-up.	0
1	DMA[1]	DMA wake-up.	0
2	GINT0	Group interrupt 0 wake-up.	-
3	GINT1	Group interrupt 1 wake-up.	0
4	PIN_INT0	GPIO pin interrupt 0 wake-up.	0
5	PIN_INT1	GPIO pin interrupt 1 wake-up.	0
6	PIN_INT2	GPIO pin interrupt 2 wake-up.	0
7	PIN_INT3	GPIO pin interrupt 3 wake-up.	0
8	UTICK	Micro-tick Timer wake-up.	0
9	MRT[1]	Multi-Rate Timer wake-up.	0
10	CT32B0[1]	Standard counter/timer CT32B0 wake-up.	0
11	CT32B1[1]	Standard counter/timer CT32B1 wake-up.	0
12	SCT0[1]	SCT0 wake-up.	0
13	CT32B3[1]	Standard counter/timer CT32B2 wake-up.	0

Bit	Symbol	Description	Reset Value
14	FLEXCOMM0	Flexcomm0 peripheral interrupt wake-up.	0
15	FLEXCOMM1	Flexcomm1 peripheral interrupt wake-up.	0
16	FLEXCOMM2	Flexcomm2 peripheral interrupt wake-up.	0
17	FLEXCOMM3	Flexcomm3 peripheral interrupt wake-up.	0
18	FLEXCOMM4	Flexcomm4 peripheral interrupt wake-up.	0
19	FLEXCOMM5	Flexcomm5 peripheral interrupt wake-up.	0
20	FLEXCOMM6	Flexcomm6 peripheral interrupt wake-up.	0
21	FLEXCOMM7	Flexcomm7 peripheral interrupt wake-up.	0
22	ADC0_SEQA[1]	ADC0 sequence A interrupt wake-up.	0
23	ADC0_SEQB[1]	ADC0 sequence B interrupt wake-up.	0
24	ADC0_THCMP[1]	ADC0 threshold and error interrupt wake-up.	0
25	D-MIC	Digital microphone interrupt wake-up.	0
26	HWVAD	Hardware voice activity detect interrupt wake-up.	0
27	USB_NEEDCLK	USB activity interrupt wake-up.	0
28	USB	USB function interrupt wake-up.	0
29	RTC	RTC interrupt alarm and wake-up timer.	0
30	-	Reserved.	-
31	MAILBOX	Mailbox interrupt wake-up.	0

Table 9. Start enable register 1 (STARTER1, main syscon: offset 0x684) bit description

Bit	Symbol	Description	Reset Value
0	PINT4	GPIO pin interrupt 4 wake-up.	0
1	PINT5	GPIO pin interrupt 5 wake-up.	0
2	PINT6	GPIO pin interrupt 6 wake-up.	0
3	PINT7	GPIO pin interrupt 7 wake-up.	0
4	CT32B2[1]	Standard counter/timer CT32B2 wake-up.	0
5	CT32B4[1]	Standard counter/timer CT32B4 wake-up.	0
31:6	-	Reserved	-

Note:

1. Typically used in sleep mode only since the peripheral clock must be running for it to function.

3.5 Entering and waking up from deep power-down mode

3.5.1 Entering deep power-down mode

Chip_POWER_EnterPowerMode() API in Fig 3 can be used to enter the deep-sleep mode.

```
Chip_POWER_EnterPowerMode(curr_pwr,
                          (SYSCON_PDRUNCFG_PD_SRAM0 |
                           SYSCON_PDRUNCFG_PD_SRAM1 |
                           SYSCON_PDRUNCFG_PD_SRAM2 |
                           SYSCON_PDRUNCFG_PD_SRAMX));
```

Fig 7. Entering deep power-down mode with RTC active

3.5.2 Wake-up from deep power-down mode

Use the following two methods to wake-up from the deep power-down mode

- RTC timer.
- Reset pin low pulse can wake-up the part from the deep power-down mode.

3.6 Other power consumption reduction considerations

3.6.1 Turning off unused analog peripherals

Other analog peripherals can be powered down for additional power savings.

3.6.2 Clock gating the digital peripherals

For example, when the IO configuration is done, the user can shut down the IOCON clock for additional power savings in all low power modes.

3.6.3 Handle unused pins in sleep/deep-sleep modes

Configure unused pins to GPIO with pull-up and pull-down disabled. In addition, set the pin to output mode with proper level based on the application for reduced leakage current. The function ConfigureUnusedPins() in the wake-up project takes care of setting up the unused pins for LPCXpresso board.

3.7 Debugging session considerations with low power modes

The debugging session can stay active in sleep mode but not in deep-sleep and deep power-down mode. A reconnect to the debugger from deep-sleep and deep power-down mode can be achieved using several methods depending on the application of the user.

3.7.1 No wake-up scheme provided in user's application

In this case, the user can reset the part if the application does not directly put the part into low power modes. The wake-up sample projects presented in this application note does not put the part into low power mode unless the user selects the low power mode and the wake-up method from the terminal window.

3.7.2 Wake-up scheme provided in user's application

The user can activate the wake-up action to release LPC5411x from the low power modes to allow reconnection to the debugger.

4. Low power mode demos

This application note provides examples to enter the low power modes and waking up from pin interrupt for sleep and deep sleep modes. The RTC alarm functionality is used to wake up from the deep power-down mode. On the LPC5411x, the part can also wake-up from deep power-down mode from the reset pin. The examples are based on the LPCOpen software platform of NXP. Keil and LPCXpresso tool chains are also supported.

4.1 Software set-up

Three IDEs were used to verify the sample projects:

- Keil 5.12 with Patch for LPC5411x (included in the AN package).
- IAR Embedded Workbench with patch for LPC5411x.
- LPCXpresso 8.1 (can download from http://www.lpcware.com).

Zip file “\lpc5411x_lowpower_modes.zip” contains the software for implementation based on Keil and IAR IDEs. Unzip the project folder to a folder for testing. CMSIS-DAP on board debugger is used for the Keil and IAR projects The LQFP54102 development is preconfigured with CMSIS-DAP capability. The CMSIS-DAP configuration tool is also available on http://www.lpcware.com.

Zip file “\lpc5411x_xpresso_lowpower_modes.zip” contains the software implementation based on the LPCXpresso IDE. Import these projects to an LPCXpresso workspace for testing. For the LPCXpresso projects, the CMSIS-DAP on board debugger is used.

For the low power mode demo, the board library is bypassed. By default, the chip library takes the FRO as input to PLL and sets the main clock to 100 MHz Before entering any low power mode, the application switches the main clock to 12 MHz FRO and powers down the PLL. See Section 3.2.

4.2 Hardware

The LPCXpresso LPC54114 development board is used in this implementation for the low power mode entry and exit example.

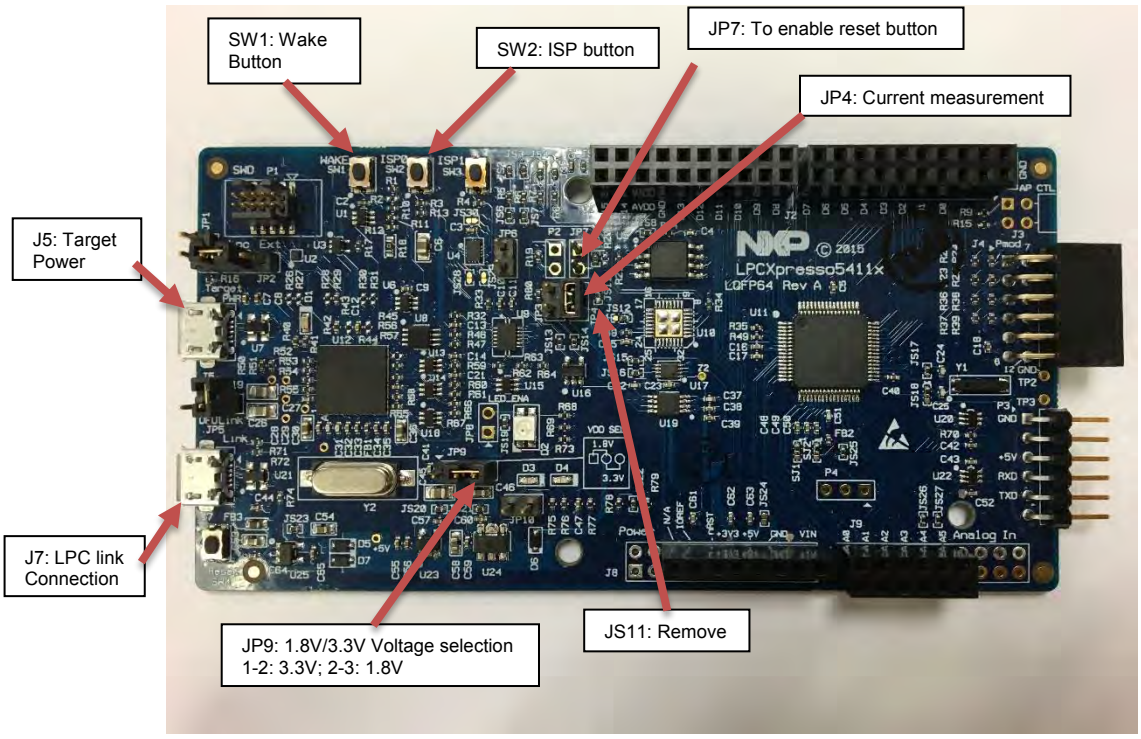

Fig 8. LPC54114 LPCXpresso development board

See http://www.lpcware.com for the LPCXpresso LPC54114 (OM13089) development board schematic information.

To allow the LPC54114 low power mode power consumption to be measured, remove JS11 and JS9, install connector at JP4 and insert current meter at JP4. See Fig 8. Install connector at JP7 to use the SW4 reset button to reset the board and remove the connection while measuring the power.

In addition, remove the JS components to reduce the leakage current. If none of these components are removed, an extra leakage current of about 140 uA is observed on top of the specified power consumption for the different low power modes. See the LQFP64 LPCXpresso board silk screen for location of the JS components. See Table 10.

Table 10. Leakage current reduction (remove below JS component)

Component	Function
JS11	Bridge to link MCU
JS9	Bridge to link nRST

This development board allows LPC54114 to operate at 3.3 V or 1.8 V. To switch between 3.3 V and 1.8 V operation, switch jumper on JP7. 1-2 position selects 3.3 V and 2-3 position selects 1.8 V.

4.3 Low power mode power consumption measurement steps

Use the following steps to measure the low power consumption.

1. Connect current meter to JP4 as shown in Fig 9.

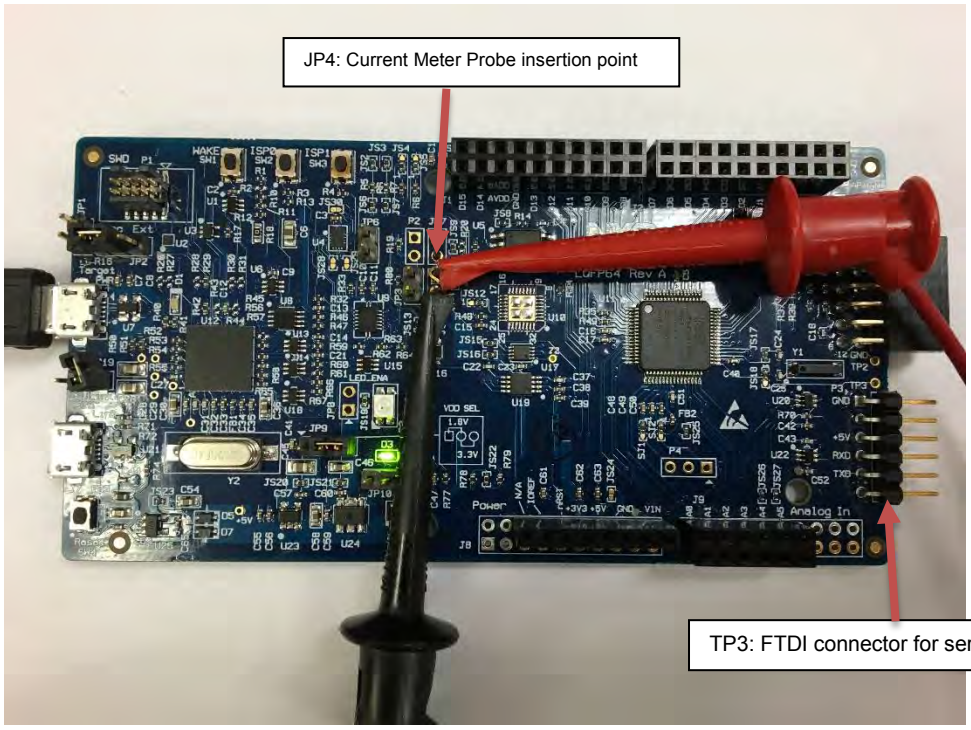

Fig 9. Current meter connection with LPCXpresso board LPC54114

2. Connect power and Link2 on board debugger from J7 to PC through USB cable.
3. Connect FTDI cable to output pins of TP3 for serial output. (connect GND, TX and RX pins)
4. Compile ‘lib_chip_5411x’, 'lib_board_lpcpresso_54114’ and ‘periph_pmu’ projects from ‘LPC5411x_power_measurement’ software package with –O0 compiler optimization.
5. Download the compiled project to the LPCXpresso board using J7 link USB connector.
6. After downloading successfully to the flash, remove the USB connector from the J7 and insert it to the J5 USB port for execution.
7. Start a serial terminal (for example, Tera Term), select appropriate COM port and set baud rate to 115200.

8. Restart the board using SW4 Reset switch and serial terminal will display output. See Fig 10.

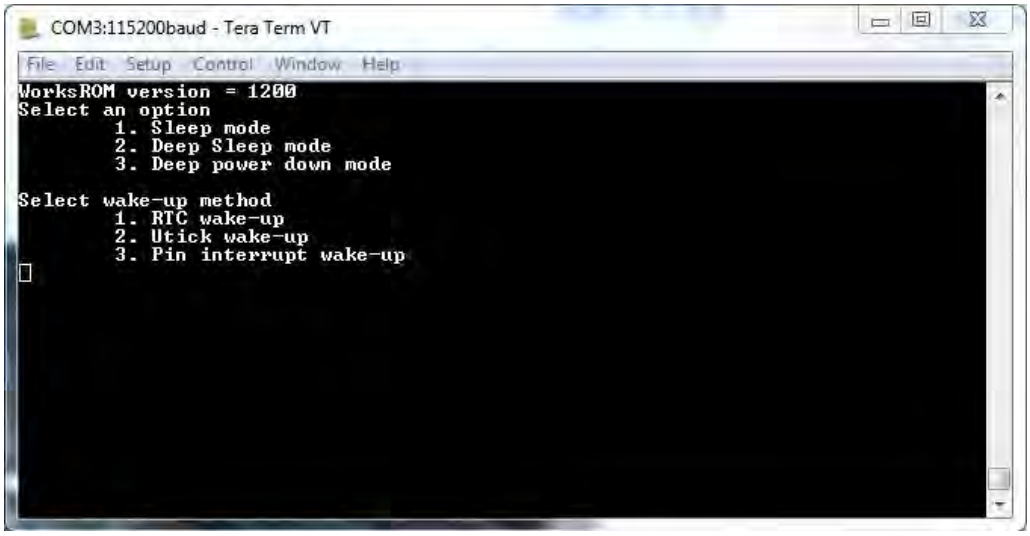

Fig 10. Serial output

9. Enter a number between 1 to 3 from the keyboard based on the low power mode user wants to select.
10. Enter a number between 1 to 3 from keyboard to select the wake-up method.
11. Take current measurement from the current meter.

    NOTE:

    1. In the example project, only the RTC interrupt is available to wake-up for the continuous execution from deep power down mode. The chip wakes up after 2 seconds from the deep power-down mode. The user can change the wake-up time by changing the third parameter from the function call 'Chip_RTC_SetWake (LPC_RTC, 2000); at line 498.
    2. If uTick wake-up method is selected, the chip will wake-up after 5 seconds from any of the low power mode except deep power-down mode.
    3. If Pin Interrupt wake-up method is selected, press wake-up button SW1 on the board to wake-up the chip from the any of the low power modes except deep power-down mode.
12. Repeat steps 9 to 11 to take the next measurement after the chip wakes-up from the low power mode.

NOTE: For the measurements of deep power-down mode (RTC GND), change the JS18 from position 1-2 to position 2-3 and ground TP3.

4.4 Wake-up time measurement steps

In this application note, the wake-up time from sleep and deep-sleep modes are measured from the wake-up I/O line (PIO0_24) being pulled low to the first instruction executed in the wake-up interrupt service routine. This first instruction in the interrupt service routine pulls another I/O line (PIO0_2) from LOW to HIGH. For Cortex-M4, the maximum interrupt latency is 12 cycles with the zero wait state configuration. Therefore, the measured wake-up time is 1 us longer than the true wake-up time of LPC54114.

In addition, an example of entering low power mode from SRAM and wake-up to execution in SRAM is provided. This mechanism features an ultra-fast wake-up time from deep sleep mode.

The LPC5411x does not have a dedicated wake-up pin active in deep power-down mode. Wake-up from deep power-down is achieved by providing a low pulse on the reset pin or use the RTC alarm timer. This application note provides an example of using the RTC alarm timer to wake-up the part from deep power-down mode.

4.4.1 Wake-up from sleep/deep-sleep mode from flash execution

Use the following steps for the wake-up time measurement with flash target.

4. Install jumper JP4 as shown in Fig 11.
5. Connect P0_24 wake-up pin to oscilloscope trigger input. P0_24 can be accessed from pin 10 at J4 as shown in Fig 11.
6. Connect P0_2 to another oscilloscope trace. P0_2 can be access from pin 1 at J9 as shown in Fig 11.
7. Connect both the oscillator probes with GND on J8.
8. Connect power and the on board debugger to a PC through J5.
9. Compile ‘lib_chip_5411x’, 'lib_board_lpcpresso_54114’ and ‘periph_pmu’ projects from ‘LPC5411x_wakeup_measurement’ software package with –O3 compiler optimization.
10. Switch the USB connection from J7 to J5 (for target power only).
11. Select low-power mode through terminal as shown in Fig 10 to perform wake-up measurement.
12. Press wake-up button.
13. For wake-up time from sleep, deep-sleep modes, measure time passed on the oscilloscope between wake-up pin (PIO0_24) going low to PIO0_2 going high as shown in Fig 12.

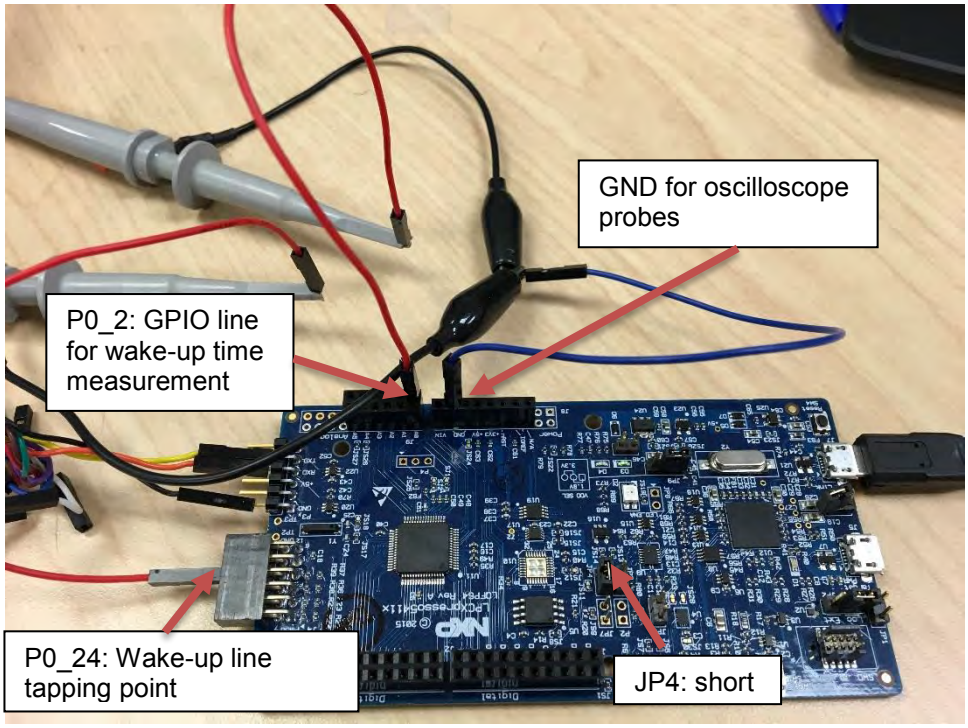

Fig 11. Connect PIO0_24 and PIO0_2 for wake-up time measurement

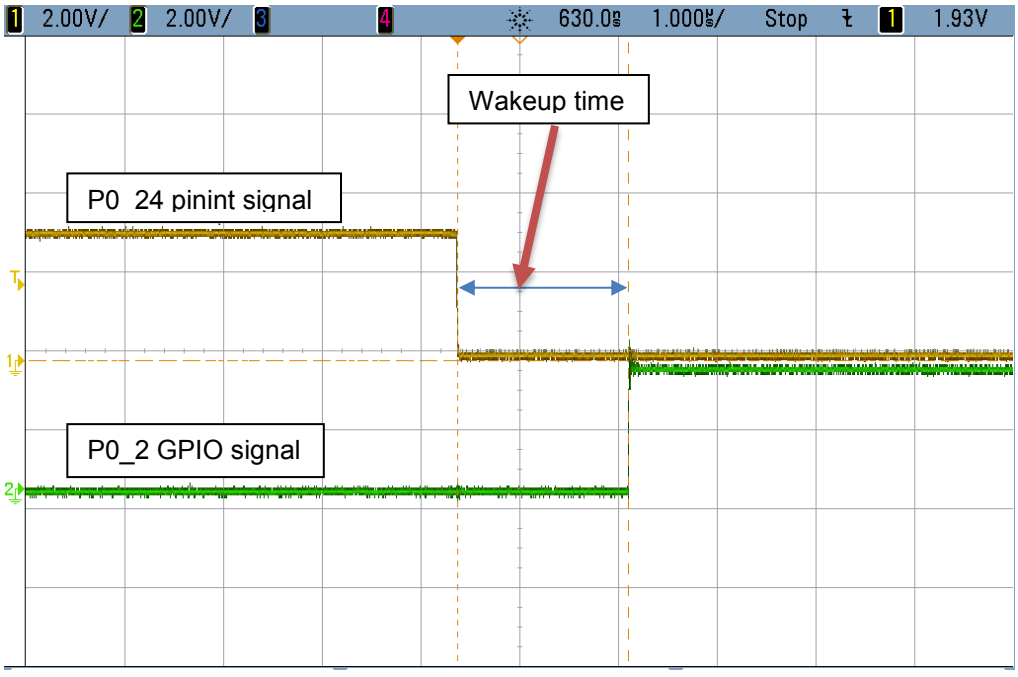

Fig 12. Wake-up time measurements from sleep and deep sleep Modes

4.4.2 Wake-up from deep-sleep mode to SRAM execution

Turning off the flash whenever possible not only reduces the flash leakage current consumption (which is about 200 uA), but also allows a special case of entering the deep-sleep mode and exit to SRAM execution with a wake-up time in the region of 20 us. This is about 50 us wake-up time improvement compared to the regular deep-sleep mode where you enter and wake-up with flash execution.

On LPC5411x, SRAM0, SRAM1 and SRAM2 are placed on different AHB matrix ports. This allows user programs to potentially obtain better performance by dividing RAM usage among the 3 ports. Such SRAM execution has several beneficial features the user might want to explore in their application.

Only Keil project is implemented to demonstrate the wake-up from low power modes to SRAM execution. To measure the wake-up time with the SRAM target:

1. Connect the current meter to JP4 and power-up the board from J5.
2. Attach Ulink-ME to P1.
3. Select the RAM_SRAM0_SRAM0 from the Keil IDE, see Fig 13.

Fig 13. Select RAM_SRAM0_SRAM0 target (Keil Example)

4. Compile ‘lib_chip_5411x’, 'lib_board_lpcpresso_54114’ and ‘periph_pmu’ projects from ‘LPC5411x_wakeup_measurement’ software package with –O3 compiler optimization.
5. Run the project in debug mode.
6. Select low power mode and wake-up method from the terminal, see Fig 10.
7. Stop IDE.
8. Press WAKE button SW1 to wake-up from the low power mode.

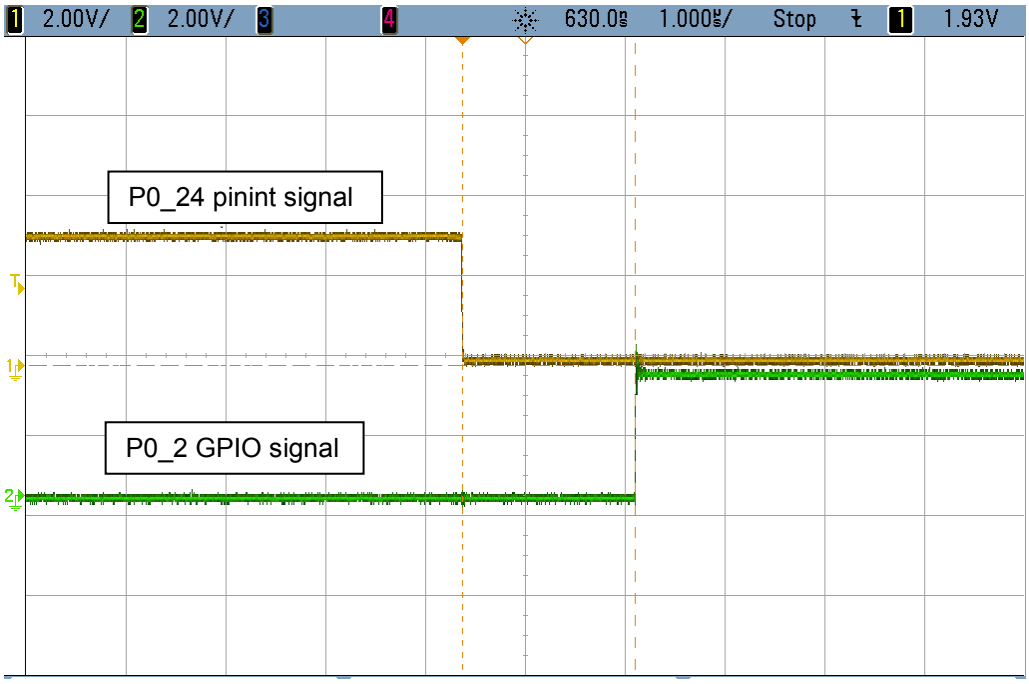

Fig 14. Wake-up from deep sleep mode to SRAM execution

4.4.3 Wake-up time measurement from deep power-down mode

The reset method is used to measure the wake-up time from the deep power-down mode. To measure the wake-up time, oscillate the PIO0_2 GPIO pin as soon as the system restarts. In this project, GPIO pin toggle code resides in ‘sysinit.c’ file, first code to execute after the system reset. The time is measured from the upper edge of nRESET pin to the lower edge of GPIO pin.

Wakeup time of the deep power-down mode can only be measured while executing the code from flash and not from SRAM, because SRAM does not retain data after the reset.

To measure the wake-up time from deep power-down mode in a flash based project:

1. Short JP4 with a jumper.

2. Connect USB connector to J5.
3. Connect FTDI-USB cable with P3. Only GND, TX and RX pins are required.
4. Connect oscillator source probe to nRST pin, pin 10 on J8.
5. Connect another oscillator probe to PIO0_2 on J9 pin 1.
6. Enable GPIO of code in sysinit.c by changing line 82 to #if 1 from #if 0 to enable GPIO toggling at startup.
7. Compile ‘lib_chip_5411x’, 'lib_board_lpcpresso_54114’ and ‘periph_pmu’ projects from ‘LPC5411x_wakeup_measurement’ software package with –O3 compiler optimization.
8. Switch the USB connector from J5 to J7.
9. Select Deep Power-down mode and Reset wake-up method from the terminal, see Fig 10.
10. Press Reset button, SW4, to measure the wake-up time. See Fig 15.

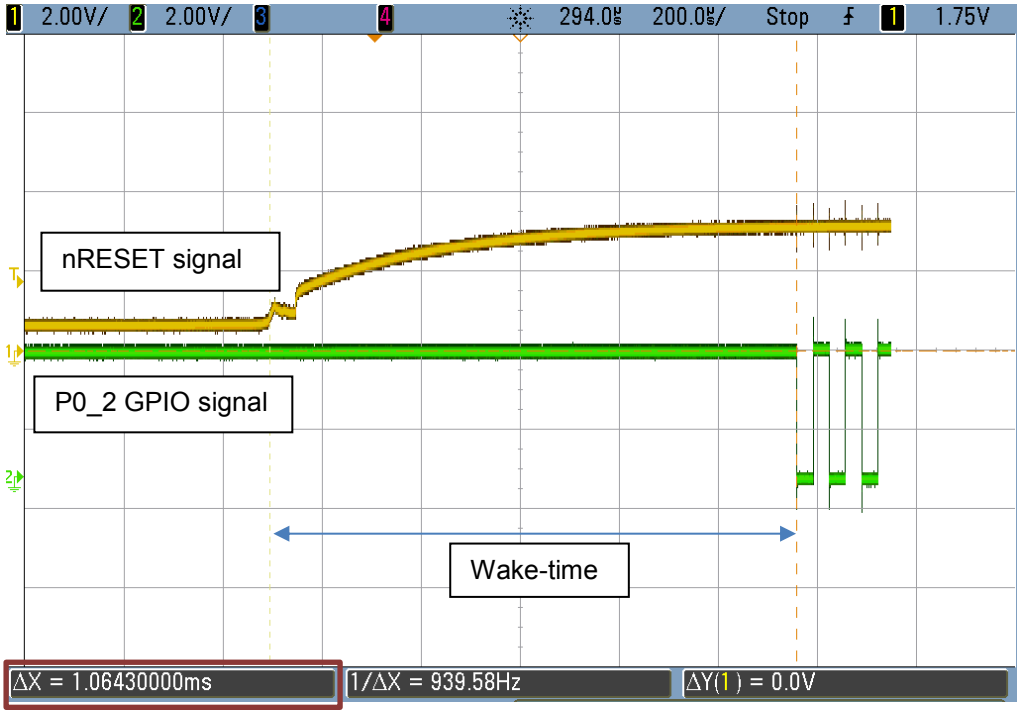

Fig 15. Wake-up from deep power-down mode

4.5 Typical low power mode power consumption and wake-up times

Table 11 shows the measurement results taken on LPCXpresso LPC5411x LQFP64 development board. These wake-up times and power consumption measurement results are very close to those specified in the product datasheet.

Table 11. Low power mode power consumption and wake-up time (VDD=3.3 V, room temperature)

Low Power Mode	SRAM retention	Power VDD = 3.3V	Wake-up time VDD = 3.3 V	Power VDD = 1.8V	Wake-up time VDD = 1.8 V
Sleep	Default entire 192kB (configurable)	1.02 mA	1.59 uS	1.02 uA	1.63 uS
Deep Sleep	64kB SRAM0	12 uA	17.66 uS	10 uA	18.22 uS
	128kB SRAM0+SRAM1	15 uA	17.68 uS	13 uA	18.23 uS
	160kB SRAM0+SRAM1+SRAM2	16 uA	17.59 uS	14 uA	18.19 uS
	192kB SRAM0+SRAM1+SRAM2+SRAMX	-	17.58 uS	-	18.20 uS
Deep Power-down	No SRAM retention/RTC enabled	736 nA	1.02 mS	384 nA	1.21 mS
	No SRAM retention/RTC disabled	360 nA		315 nA	

5. Conclusion

The LPC5411x series MCU provides great flexibility and multiple options for users to achieve low power consumption and wake-up flexibility. This flexibility allows a trade-off between power consumption and wake-up speed based on the application requirements of the user.

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 competent authorities.

Evaluation products — This product is provided on an "as is" and "with all faults" basis for evaluation purposes only. NXP Semiconductors, its affiliates and their suppliers expressly disclaim all warranties, whether express, implied or statutory, including but not limited to the implied warranties of non-infringement, merchantability and fitness for a particular purpose. The entire risk as to the quality, or arising out of the use or performance, of this product remains with customer.

In no event shall NXP Semiconductors, its affiliates or their suppliers be liable to customer for any special, indirect, consequential, punitive or incidental damages (including without limitation damages for loss of business, business interruption, loss of use, loss of data or information, and the like) arising out the use of or inability to use the product, whether or not based on tort (including negligence), strict liability, breach of contract, breach of warranty or any other theory, even if advised of the possibility of such damages.

Notwithstanding any damages that customer might incur for any reason whatsoever (including without limitation, all damages referenced above and all direct or general damages), the entire liability of NXP Semiconductors, its affiliates and their suppliers and customer's exclusive remedy for all of the foregoing shall be limited to actual damages incurred by customer based on reasonable reliance up to the greater of the amount actually paid by customer for the product or five dollars (US$5.00). The foregoing limitations, exclusions and disclaimers shall apply to the maximum extent permitted by applicable law, even if any remedy fails of its essential purpose.

6.3 Licenses

Purchase of NXP <xxx> components
<License statement text>

6.4 Patents

Notice is herewith given that the subject device uses one or more of the following patents and that each of these patents may have corresponding patents in other jurisdictions.

<Patent ID> — owned by <Company name>

6.5 Trademarks

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

<Name> — is a trademark of NXP B.V.

7. List of figures

8. List of tables

9. Contents

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

Date of release: 18 July 2016
Document identifier: AN11799