

Freescale Semiconductor

Application Note

Document Number: AN4470 Rev. 0, 01/2012

Using Low Power modes on Kinetis family

by: Daniel Martinez, César Manzanarez Guadalajara, México

1 Introduction

The power consumption of devices and the implications around designing for low power are common topics currently.

Freescale offers solutions with energy efficient products that can help you achieve the power consumption needed for your application.

This application note will guide you to achieve the available power consumption modes on the Kinetis family processors.

It is important to indicate that this application note is based on the Kinetis 50 MHz devices, so some low power modes might not be available in all the Kinetis family of devices this is indicated in the corresponding chapter.

Contents

1 2 2 4 5 5 6
2 4 5 5
4 5 5
5 5
5 5
5
6
6
7
8
9
9
10
11
12
13
15
18
19
20
38
42
42



© Freescale Semiconductor, Inc., 2012. All rights reserved.



Low power overview and best practices

2 Low power overview and best practices

There are many reasons why a designer of an embedded system is increasingly striving to stay within tight power consumption budgets.

One reason is money, having a portable device with a poor battery life can be the cause of losing in the market against more power efficient competitors.

There are other applications in which money goes to second term, like medical devices that are implanted in a patient's body. This device needs to run for years on battery life, replacing the battery means taking the patient to surgery, so in this scenario an efficient power solution is critical.

A low power design should take into consideration the tradeoff between power consumption and performance, and use every possible feature provided by the device to accomplish the best results.

Both software and hardware designers should be involved in a power efficient design, and estimate the application's power needs early in the planning stage to avoid any late redesigns.

Tips and tricks to consider when designing for low power:

- Set the pins to a known state: For the Kinetis family it is recommended to leave unused pines floating and configure them as disabled.
- Select and configure the desired clock mode. Keep in mind that higher core frequency means higher power consumption.
- Higher operating temperature increases power consumption.
- Disable clock to all unused modules
- Clock monitor output and debug enabled will increase power consumption and must be avoided.

In the following chapters we will study in detail each of the modules involved in the microcontroller operation and its correlation with the low power modes available in the Kinetis family.

3 System Clock Generation

The Kinetis family features a Multipurpose Clock Generator (MCG) module capable of using an external (System oscillator, RTC oscillator) or an internal source to generate the system clock.

In Figure 1 we can view the dependencies among these modules:

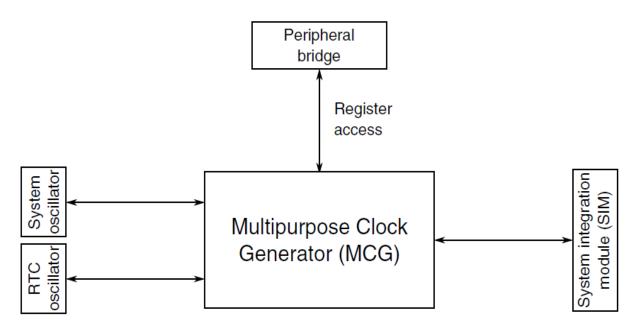


Figure 1. MCG Configuration

The system oscillator and the RTC oscillator are the external sources that you can use to feed the MCG.

The OSC module is a crystal oscillator. The module, in conjunction with an external crystal or resonator, generates a reference clock for the MCU.

The RTC oscillator module provides the clock source for the RTC. The RTC oscillator module, in conjunction with an external crystal, generates a reference clock for the RTC.

In addition you have the option of selecting between two internal reference clocks as the input source for the MCG as follows:

• Slow IRC (32 kHz) and Fast IRC (4 MHz)

These internal clocks are part of the MCG module.

The MCG module controls which clock source is used to derive the system clocks. The clock generation logic divides the selected clock source into a variety of clock domains, including the clocks for the system bus masters, system bus slaves, and flash memory.

In summary there are 4 input clock sources that can be used as the clock generator for the microcontroller, you should select the source depending on the requirements of your application.

Module dependencies:

- All modules are dependent on the MCG for one or more of their clocks
- Oscillator module relies on the MCG for the control signals.
- The oscillator must be set up correctly, from within the MCG, before it is used
- System clock dividers must be setup before increasing the MCGOUT frequency to ensure the maximum clock rate is not exceeded.

Using Low Power modes on Kinetis family, Rev. 0

System Clock Generation

- MCG is powered in all low power modes except very low leakage stop modes.
- Crystal oscillator pins are XTAL and EXTAL by default out of reset.
- There is no clock gating associated with the MCG module itself.

There are in total 9 different modes of operation for the MCG: FEI, FEE, FBI, FBE, PBE, PEE, BLPI, BLPE, and Stop.

In the following chapter you can view important details of the system clock setup in the different low power modes.

For more details in each specific module you can refer to the device Reference Manual.

3.1 Functionality of the system clock in low power modes

- STOP Normal Stop Mode
 - Can enter STOP from any MCG mode
 - Can optionally keep the PLL enabled but the output will be gated off C5[PLLSTEN]
 - Can optionally keep the internal reference clock enabled C1[IREFSTEN]
 - Can optionally keep the external reference clock enabled OSC_CR[EREFSTEN]
 - Will exit STOP in the same MCG mode when STOP was entered
 - The exception to this if STOP is entered when in PEE mode and PLLSTEN=0, in this case the MCG will be in PBE mode when STOP is exited
- VLPR Very Low Power Run Mode
 - Can enter VLPR from BLPI mode with the fast IRC selected.
 - Can enter VLPR from BLPE mode if the external clock is 4 MHz max.
 - Will exit VLPR in the same MCG mode when VLPR was entered (BLPI or BLPE)
- VLPW Very Low Power Wait Mode
 - Can enter VLPW from BLPI mode with the fast IRC selected.
 - Can enter VLPW from BLPE mode if the external clock is 4 MHz max.
 - Will exit VLPW in the same MCG mode when VLPW was entered (BLPI or BLPE)
 - Can optionally keep the external reference clock enabled (4 MHz max.)
- VLPS Very Low Power Stop
 - Can enter VLPS from any MCG mode
 - Can optionally keep the external reference clock enabled (4 MHz max.)
 - MCG is static with no clocks active (IREFSTEN and PLLSTEN have no effect)
 - Will exit VLPS in the same MCG mode when VLPS was entered
 - The exception to this if VLPS is entered when in PEE mode, in this case the MCG will be in PBE mode when VLPS is exited
- LLS Low Leakage Stop
 - Can enter LLS from any MCG mode



- Can optionally keep the external reference clock enabled in low range and low power oscillator mode (32 kHz)
- MCG is static with no clocks active (IREFSTEN and PLLSTEN have no effect)
- Will exit LLS in the same MCG mode when LLS was entered
 - The exception to this if LLS is entered when in PEE mode, in this case the MCG will be in PBE mode when LLS is exited
- VLLS0/1/2/3 Very Low Leakage Stop
 - Can enter VLLSx from any MCG mode
 - MCG is off, no clocks active, register states are not maintained
 - Will exit in FEI mode (reset state)
 - Can optionally keep the external reference clock enabled in low range and low power oscillator mode (32 kHz). Must reconfigure the oscillator before clearing the ACKISO bit.

4 Description and usage of low power modes

The operating modes described in this chapter apply for the Kinetis family devices.

The available modes of operation are the following:

- Run
- Wait
- Stop
- VLPR (Very Low Power Run)
- VLPW (Very Low Power Wait)
- VLPS (Very Low Power Stop)
- LLS (Low Leakage Stop)
- VLLS3 (Very Low Leakage Stop3)
- VLLS2 (Very Low Leakage Stop2)
- VLLS1 (Very Low Leakage Stop1)
- VLLS0 (Very Low Leakage Stop0)¹
- BAT (Backup battery only)

In this chapter we will discuss details about all modes of operation relevant to the topic of this application note. That is all modes except the Run and BAT.

NOTE

For a full description of the operating modes refer to the device reference manual, search for chapter "Power modes".

^{1.} This mode of operation is **present only in the 50 MHz devices.** It has two modes of operation: Power On reset detect circuit enabled or disabled.



Description and usage of low power modes

4.1 Run mode

This is the default mode out of reset.

Allows maximum performance of the chip.

Mode of operation details:

- Selected after any Reset
- On chip voltage regulator is on, full capability
- Stack pointer (SP), Program Counter (PC) and link register are set
- ARM processor exits reset and reads the start SP
- ARM processor reads the start PC from vector table
- Reduce power by clearing clock gating bits in SCGCx

4.2 Wait mode

Allows peripherals to function while the core is in sleep mode, reducing power.

NVIC remains sensitive to interrupts; peripherals continue to be clocked.

Mode of operation details:

- ARM core enters Sleep Mode
- ARM core is clock gated (HCLK = OFF)
- NVIC remains sensitive to interrupts (FCLK = ON)
- Peripherals continue to be clocked
- Reduce power by clearing clock gating bits in SCGCx
- On interrupt the ARM core exits Sleep Mode: Resume processing

Table 1. How to enter Wait mode

From	То	Trigger conditions
Run		Execute WAIT(); - This means that sleep-now or sleep-on-exit modes entered with SLEEPDEEP clear
Wait	Run	Interrupt or Reset

4.3 Stop mode

Places chip in static state. Lowest power mode that retains all registers while maintaining LVD protection. NVIC is disabled; AWIC is used to wake up from interrupt; peripheral clocks are stopped.

- ARM core enters DeepSleep Mode
- ARM core is clock gated (HCLK = OFF)



- NVIC is disable (FCLK = OFF)
- WIC is used to wake up from interruptions
- Platform and peripheral clock are stopped
- MCG module can be configured to leave reference clocks running
- All SRAM is operating (content retained and I/O states held)

Table 2. How to enter Stop mode

From	То	Trigger Conditions
Run	Stop	Execute STOP(); This means that sleep-now or sleep-on-exit modes entered with SLEEPDEEP set
Stop	Run	Interrupt or Reset – Interrupt goes to ISR (no LLWU)

4.4 VLPR (Very Low Power Run)

On-chip voltage regulator is in a low power mode that supplies only enough power to run the chip at a reduced frequency. Reduced frequency Flash access mode (1 MHz); LVD off; internal oscillator provides a low power 4 MHz source for the core, the bus and the peripheral clocks.

- On chip voltage regulator is in a mode that supplies only enough power to run the MCU in a reduced frequency
- Core and Bus clock limited to 4Mhz
- Flash frequency limited to 1Mhz
- Reduce power by clearing clock gating bits in SCGCx
- Flash programming and erasing is not allowed
- FlexMemory (EEPROM) programming is not allowed
- Clock Monitor:
 - CME0 and CME1 should also be set to a logic 0 before entering VLPR or VLPW power modes if the MCG is in BLPE mode.

Description and usage of low power modes

From	То	Trigger Conditions
Run	VLPR	Reduce system bus and core frequency to 2 MHz or less Flash access frequency limited to 1 MHz, AVLP = 1 Set RUNM = 10 Note: Poll VLPRS bit before executing VLPR specific code (You can wait ~5 μs instead of waiting for VLPRS)
VLPR	Run	Set RUNM = 00 or Interrupt with LPWUI = 1 or Reset Note: Poll REGONS bit before increasing frequency.

Table 3. How to enter VLPR mode

4.5 VLPW (Very Low Power Wait)

Same as VLPR but with the core in sleep mode to further reduce power; NVIC remains sensitive to interrupts (FCLK = ON). On-chip voltage regulator is in a low power mode that supplies only enough power to run the chip at a reduced frequency.

- ARM core enters Sleep Mode
- ARM core is clock gated (HCLK = OFF)
- NVIC remains sensitive to interrupts (FCLK = ON)
- On chip voltage regulator is in a mode that supplies only enough power to run the MCU in a reduced frequency
- System and Bus clock limited to 4Mhz
- Reduce power by clearing clock gating bits in SCGCx
- Clock Monitor:
 - CME0 and CME1 should also be set to a logic 0 before entering VLPR or VLPW power modes if the MCG is in BLPE mode.



From	То	Trigger Conditions
VLPR	VLPW	Execute WAIT();
VLPW	VLPR	Interrupt with LPWUI = 0
VLPW	RUN	Interrupt with LPWUI = 1 or Reset

Table 4. How to enter VLPW mode

4.6 VLPS (Very Low Power Stop)

Places chip in static state with LVD operation off. Lowest power mode with ADC and pin interrupts functional. Peripheral clocks are stopped, but LPTimer, RTC, CMP, TSI can be used. NVIC is disabled (FCLK =OFF), AWIC is used to wake up from interrupt.

On-chip voltage regulator is in a low power mode that supplies only enough power to run the chip at a reduced frequency. All SRAM is operating (content retained and I/O states held).

Mode of operation details:

- ARM core enters DeepSleep Mode
- ARM core is clock gated (HCLK = OFF)
- NVIC is disable (FCLK = OFF)
- WIC is used to wake up from interruptions
- Platform and peripheral clock are stopped
- MCG module can be configured to leave reference clocks running
- On chip voltage regulator is in a mode that supplies only enough power to run the MCU in a reduced frequency
- All SRAM is operating (content retained and I/O states held)

Table 5. How to enter VLPS mode

From	То	Trigger Conditions
VLPR	VLPS	STOPM = 000 or 010, execute STOP();
VLPS	VLPR	Interrupt with LPWUI = 0
RUN	VLPS	AVLP=1, STOPM =010, execute STOP();
VLPS	RUN	Interrupt with LPWUI= 1 or Reset

4.7 LLS (Low Leakage Stop)

State retention power mode. Most peripherals are in state retention mode (with clocks stopped), but LLWU, LPTimer, RTC, CMP, TSI can be used. NVIC is disabled, LLWU is used to wake up.

Description and usage of low power modes

NOTE

The LLWU interrupt must not be masked by the interrupt controller to avoid a scenario where the system does not fully exit stop mode on an LLS recovery.

All SRAM is operating (content retained and I/O states held).

Mode of operation details:

- ARM core enters Deep Sleep Mode
- ARM core is clock gated (HCLK = OFF)
- NVIC is disable (FCLK = OFF)
- LLWU is used to wake up from interruptions
- Platform and peripheral clock are stopped
- MCG module can be configured to leave reference clocks running
- On chip voltage regulator is in a mode that supplies only enough power to run the MCU in a reduced frequency
- All SRAM is operating (content retained and I/O states held)
- Most of peripherals are in state retention mode (cannot operate)

Table 6. How to enter LLS mode

From	То	Trigger Conditions
Run	LLS	Set ALLS in PMPROT, PMCTRL_STOPM = 011, Execute STOP();
LLS	Run	Wakeup from enabled LLWU pin or module source or Reset pin
VLPR	LLS	Set ALLS in PMPROT, PMCTRL_STOPM = 011, Execute STOP();

4.8 VLLS3 (Very Low Leakage Stop3)

Most peripherals are disabled (with clocks stopped), but LLWU, LPTimer, RTC, CMP, TSI can be used. NVIC is disabled, LLWU is used to wake up. SRAM_U and SRAM_L remain powered on (content retained and I/O states held).

- ARM core enters SleepDeep Mode
- ARM core is clock gated (HCLK = OFF)
- NVIC is disable (FCLK = OFF)
- LLWU should configure by user to enable the desire wake up source
- Platform and peripheral clock are stopped
- MCG module can be configured to leave reference clocks running



- On chip voltage regulator is in a mode that supplies only enough power to run the MCU in a reduced frequency
- All SRAM is operating (content retained and I/O states held)
- Most modules are disabled

From	То	Trigger Conditions
Run	VLLS (3,2,1,0)	Set AVLLSx in PMPROT, PMCTRL_STOPM = 100 VLLSM = 011 for VLLS3, 010 for VLLS2, 001 for VLLS1, 000 for VLLS0 Note: For VLLS0 you must set or clear PORPO to disable or enable (respectively) the POR detect circuit. Execute STOP();
VLLS (3,2,1,0)	Run	Wakeup from enabled LLWU input source or Reset. All wakeup goes through Reset sequence. Check SRS for source of wakeup. Check VLLSM for mode
VLPR	VLLS(3,2,1,0)	Set AVLLSx in PMPROT, PMCTRL_STOPM = 100 VLLSM = 011 for VLLS3, 010 =for VLLS2, 001 for VLLS1, 000 for VLLS0 Note: For VLLS0 you must set or clear PORPO to disable or enable (respectively) the POR detect circuit. Execute STOP();

Table 7. How to enter VLLS3 mode

4.9 VLLS2 (Very Slow Leakage Stop2)

Most peripherals are disabled (with clocks stopped), but LLWU, LPTimer, RTC, CMP, TSI can be used. NVIC is disabled; LLWU is used to wake up.

SRAM_L is powered off. A portion of SRAM_U remains powered on (content retained and I/O states held).

Mode of operation details:

- ARM core enters SleepDeep Mode
- ARM core is clock gated (HCLK = OFF)
- NVIC is disable (FCLK = OFF)
- LLWU should configure by user to enable the desire wake up source
- Platform and peripheral clock are stopped

Using Low Power modes on Kinetis family, Rev. 0



Description and usage of low power modes

- MCG module can be configured to leave reference clocks running
- On chip voltage regulator is in a mode that supplies only enough power to run the MCU in a reduced frequency
- Only partial SRAM is operating (content retained and I/O state held)
- Most modules are disabled

From	То	Trigger Conditions
Run	VLLS (3,2,1,0)	Set AVLLSx in PMPROT, PMCTRL_STOPM = 100 VLLSM = 011 for VLLS3, 010 for VLLS2, 001 for VLLS1, 000 for VLLS0 Note: For VLLS0 you must set or clear PORPO to disable or enable (respectively) the POR detect circuit. Execute STOP();
VLLS(3,2,1,0)	Run	Wakeup from enabled LLWU input source or Reset. All wakeup goes through Reset sequence. Check SRS for source of wakeup. Check VLLSM for mode
VLPR	VLLS(3,2,1,0)	Set AVLLSx in PMPROT, PMCTRL_STOPM = 100 VLLSM = 011 for VLLS3, 010 for VLLS2, 001 for VLLS1, 000 for VLLS0 Note: For VLLS0 you must set or clear PORPO to disable or enable (respectively) the POR detect circuit. Execute STOP();

Table 8. How to enter VLLS2 mode

4.10 VLLS1 (Very Low Leakage Stop1)

Most peripherals are disabled (with clocks stopped), but LLWU, LPTimer, RTC, CMP, TSI can be used. NVIC is disabled; LLWU is used to wake up.

All of SRAM_U and SRAM_L are powered off. The 32-byte system register file and the 32-byte VBAT register file remain powered for customer-critical data.

- ARM core enters SleepDeep Mode
- ARM core is clock gated (HCLK = OFF)
- NVIC is disable (FCLK = OFF)
- LLWU should configure by user to enable the desire wake up source



- Platform and peripheral clock are stopped
- MCG module can be configured to leave reference clocks running
- On chip voltage regulator is in a mode that supplies only enough power to run the MCU in a reduced frequency
- All SRAM is powered down, only 32 byte register content retained and I/O states held
- Most modules are disabled

From	То	Trigger Conditions
Run	VLLS (3,2,1,0)	Set AVLLSx in PMPROT, PMCTRL_STOPM = 100 VLLSM = 011 for VLLS3, 010 for VLLS2, 001 for VLLS1, 000 for VLLS0 Note: For VLLS0 you must set or clear PORPO to disable or enable (respectively) the POR detect circuit. Execute STOP();
VLLS(3,2,1,0)	Run	Wakeup from enabled LLWU input source or Reset. All wakeup goes through Reset sequence. Check SRS for source of wakeup. Check VLLSM for mode.
VLPR	VLLS (3,2,1,0)	Set AVLLSx in PMPROT, PMCTRL_STOPM = 100 VLLSM = 011 for VLLS3, 010 for VLLS2, 001 for VLLS1, 000 for VLLS0 Note: For VLLS0 you must set or clear PORPO to disable or enable (respectively) the POR detect circuit. Execute STOP();

Table 9.	How to	enter VL	LS1 mode

4.11 VLLS0 (Very Low Leakage Stop0)

LLWU, RTC. All SRAM powered off.

- ARM core enters SleepDeep Mode
- ARM core is clock gated (HCLK = OFF)
- NVIC is disabled (FCLK = OFF)
- LLWU should configure by user to enable the desire wake up source
- Platform and peripheral clocks are stopped



Module operation in Low Power modes

- MCG module can be configured to leave reference clocks running
- On chip voltage regulator is in a mode that supplies only enough power to run the MCU in a reduced frequency
- All SRAM powered off.
- Most modules are disabled except LLWU and RTC

Table 10. How to enter VLLS0 mode

From	То	Trigger Conditions
Run	VLLS (3,2,1,0)	Set AVLLSx in PMPROT, PMCTRL_STOPM = 100 VLLSM = 011 for VLLS3, 010 for VLLS2, 001 for VLLS1, 000 for VLLS0 Note: For VLLS0 you must set or clear PORPO to disable or enable (respectively) the POR detect circuit. Execute STOP();
VLLS (3,2,1,0)	Run	Wakeup from enabled LLWU input source or Reset. All wakeup goes through Reset sequence. Check SRS for source of wakeup. Check VLLSM for mode.
VLPR	VLLS (3,2,1,0)	Set AVLLSx in PMPROT, PMCTRL_STOPM = 100 VLLSM = 011 for VLLS3, 010 for VLLS2, 001 for VLLS1, 000 for VLLS0 Note: For VLLS0 you must set or clear PORPO to disable or enable (respectively) the POR detect circuit. Execute STOP();

5 Module operation in Low Power modes

In this chapter we can view the microcontroller's module status among the different power modes.

In order to choose the most proper low power mode for your application you must first make a list of the modules that need to be enabled.

The following table is useful to decide what low power mode your application should enter based on the modules that you require to be enabled.

Meaning of the terms used in the table:

• FF: Full functionality



- Static: Module register states and associated memories are retained
- Powered: Memory is powered to retain contents
- Low power: Flash has a low power state that retains configuration registers to support faster wakeup
- Off: Modules are powered off; module is in reset state upon wakeup
- Wakeup: Modules can serve as a wakeup source for the chip

For further details you can refer to the device Reference Manual, chapter "Module Operation in Low Power Modes".

Modules	Stop	VLPR	VLPW	VLPS	LLS	VLLSx			
Core modules									
NVIC	static	FF	FF	static	static	OFF			
	System modules								
Mode Controller	FF	FF	FF	FF	FF	FF			
LLWU ¹	static	static	static	static	FF	FF ²			
Regulator	ON	low power	low power	low power	low power	low power in VLLS2/3, OFF in VLLS0/1			
LVD	ON	disabled	disabled	disabled	disabled	disabled			
Brown-out Detection	ON	ON	ON	ON	ON	ON in VLLS1/2/3, optionally disabled in VLLS0 ³			
DMA	static	FF	FF	static	static	OFF			
Watchdog	FF	FF	FF	FF	static	OFF			
EWM	static	FF	static	static	static	OFF			
			Clocks						
1kHz LPO	ON	ON	ON	ON	ON	ON In VLLS1/2/3, OFF in VLLS0			
System oscillator (OSC)	OSCERCLK optional	OSCERCLK max of 4MHz crystal	OSCERCLK max of 4MHz crystal	OSCERCLK max of 4MHz crystal	limited to low range/low power	limited to low range/low power in VLLS1/2/3, OFF In VLLS0			
MCG	static - MCGIRCLK optional; PLL optionally on but gated	4 MHz IRC	4 MHz IRC	static - no clock output	static - no clock output	OFF			
Core clock	OFF	4 MHz max	OFF	OFF	OFF	OFF			
System clock	OFF	4 MHz max	4 MHz max	OFF	OFF	OFF			
Bus clock	OFF	4 MHz max	4 MHz max	OFF	OFF	OFF			

Module operation in Low Power modes

Memory and memory Interfaces							
Modules	Stop	VLPR	VLPW	VLPS	LLS	VLLSx	
Flash	powered	1 MHz max access - no pgm	low power	low power	OFF	OFF	
Portion of SRAM_U ⁴	low power	low power	low power	low power	low power	low power in VLLS3,2	
Remaining SRAM_U and all of SRAM_L	low power	low power	low power	low power	low power	low power in VLLS3	
FlexMemory	low power	low power ⁵	low power	low power	low power	OFF	
Register files ⁶	powered	powered	powered	powered	powered	powered	
EzPort	disabled	disabled	disabled	disabled	disabled	disabled	
		Com	munication inter	faces			
USB FS/LS	static	static	static	static	static	OFF	
USB DCD	static	FF	FF	static	static	OFF	
USB Voltage Regulator	optional	optional	optional	optional	optional	optional	
UART	static, wakeup on edge	125 kbps	125 kbps	static, wakeup on edge	static	OFF	
SPI	static	1 Mbps	1 Mbps	static	static	OFF	
I ² C	static, address match wakeup	100 kbps	100 kbps	static, address match wakeup	static	OFF	
I ² S	FF with external clock ⁷	FF	FF	FF with external clock ⁷	static	OFF	
	Security						
CRC	static	FF	FF	static	static	OFF	



Timers							
FTM	static	FF	FF	static	static	OFF	
PIT	static	FF	FF	static	static	OFF	
PDB	static	FF	FF	static	static	OFF	
LPTMR	FF	FF	FF	FF	FF	FF ⁸	
RTC - 32kHz OSC ⁶	FF	FF	FF	FF	FF	FF	
CMT	static	FF	FF	static	static	OFF	
	Analog						
16-bit ADC	ADC Internal clock only	FF	FF	ADC Internal clock only	static	OFF	
CMP ⁹	HS or LS compare	FF	FF	HS or LS compare	LS compare	LS compare in VLLS1/2/3, OFF in VLLS0	
6-bit DAC	static	FF	FF	static	static	static	
VREF	FF	FF	FF	FF	static	OFF	
Human-machine interfaces							
GPIO	wakeup	FF	FF	wakeup	static, pins latched	OFF, pins latched	
TSI	wakeup	FF	FF	wakeup	wakeup ¹⁰	wakeup ^{10, 11}	

Figure 2. Module operation in low power modes

- 1. Using the LLWU module, the external pins available for this chip do not require the associated peripheral function to be enabled. It only requires the function controlling the pin (GPIO or peripheral) to be configured as an input to allow a transition to occur to the LLWU.
- 2. Since LPO clock source is disabled, filters will be bypassed during VLLS0.
- 3. The VLLSCTRL[PORPO] bit in the SMC module controls this option.
- 4. A 8 KB portion of SRAM_U block is left powered on in low power mode VLLS2.
- 5. FlexRAM enabled as EEPROM is not writable in VLPR and writes are ignored. Read accesses to FlexRAM as EEPROM while in VLPR are allowed. There are no access restrictions for FlexRAM configured as traditional RAM.
- 6. These components remain powered in BAT power mode.
- 7. Use an externally generated bit clock or an externally generated audio master clock (including EXTAL).
- 8. System OSC and LPO clock sources are not available in VLLS0.
- CMP in Stop or VLPS supports high speed or low speed external pin to pin or external pin to DAC compares. CMP in LLS or VLLSx only supports low speed external pin to pin or external pin to DAC compares. Windowed, sampled & filtered modes of operation are not available while in stop, VLPS, LLS, or VLLSx modes.
- 10. TSI wakeup from LLS and VLLSx modes is limited to a single selectable pin.
- 11. System OSC and LPO clock sources are not available in VLLS0.

6 Allowed power mode transitions

This chapter is a brief overview of the allowed power mode transitions. It is important to keep in mind that there are restrictions for switch among all power modes.

The following figure is intended to help you visualize the power mode transitions allowed:

Using Low Power modes on Kinetis family, Rev. 0



Allowed power mode transitions

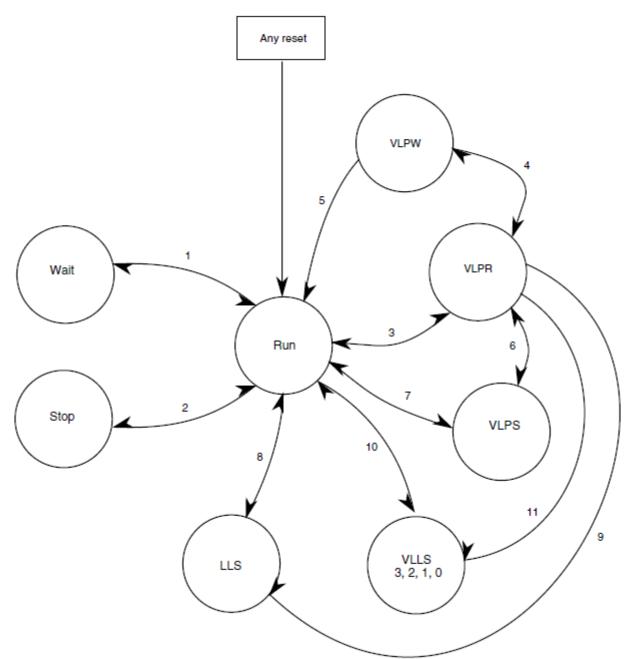


Figure 3. Power mode state transition diagram

As given in the figure above not all mode transitions are allowed, for example if the device is in VLLSx mode and you want to change to Stop mode you must first bring back the device to Run mode and then switch to Stop mode.

Any reset always brings the chip back to the normal run state.

For more details refer to chapter "Power mode transitions" in the device Reference Manual.



In this chapter we will review the necessary conditions to enter the available power modes based on the demo software.

The objective of this demo is to get familiar with the steps needed to enter the available power modes in the Kinetis family of devices.

It is important to clarify that this demo uses only one mode to exit from any operation mode, but there are many ways to exit from any given operation mode as previously explained in this document.

This demo is prepared to run on TWR-K20D50M with a K20DX128 device.

Low Power exit modes:

External pin SW2 has been enabled to wake up via the LLWU from any LLS and VLLSx mode, all wakeup interrupts from these modes must go through LLWU_ISR.

All other modes (non-leakage) can be taken out of low power mode via an interrupt from any of the available port enabled pins, in the case of this demo that is SW2 or SW3.

The following is required:

- 1. IAR Embedded Workbench for ARM v6.30 or CodeWarrior MCU V10.1 (with *MCU 10.1 Kinetis 50MHz Service Pack* installed)
- 2. TWR-K20D50M tower board.
- P&E OSBDM OSJTAG Virtual Serial Toolkit. You can download it from the following site: http://www.pemicro.com/osbdm/index.cfm
- 4. Demo project: "low_power_demo".

This toolkit contains the P&E terminal utility that you need to visualize the menu options via serial over USB.

Before running the demo you need to configure your IDE to the proper settings depending on the programming/debugging interface that you will use.

NOTE

If your preferred debug interface is already installed you can disregard the following section and go directly to chapter 7.2 "Steps to run the demo".

7.1 Flash programming and debugging settings

In this section we indicate the settings needed to program and debug you application into the Kinetis board using IAR or CodeWarrior IDEs with any of the following debug interfaces: OSJTAG, P&E Multilink, Segger J-Link.

IAR and OSJTAG/P&E Multilink

Flashloader *FlashK20Xxxx* must be used for these connections.



Unzip iar_FlashK20Xxxx_8k_ram.zip located in the demo project zip file and follow the instructions below.

1. Place all of the files in this zip (except "readme.txt") in your *IAR Systems**Embedded Workbench.6.0* x\arm\config\flashloader\Freescale folder

NOTE

This applies for the current IAR version IAR 6.30.4, for newer versions these steps will not be necessary.

- 2. Open your IAR workspace file and select a flash configuration for the project
 - a) Open IAR Embedded Workbench IDE.
 - b) Open the *low_power_demo* workspace.

To do this you can drag and drop the "low_power_demo.eww" located on the following path:

{Demo Installation path}\build\iar\low_power_demo

3. In the project options select Debugger->Download.

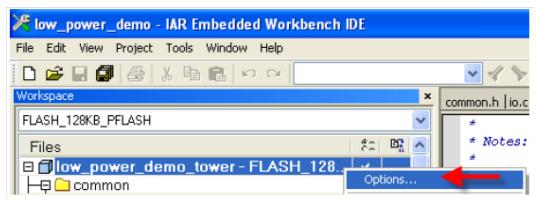


Figure 4.

- 4. Check the "Use flash loader(s)" box.
- 5. Check the "Override default .board file box".
- 6. Provide the path to the FlashK20Xxxx.board file. This will be given as follows: \$TOOLKIT_DIR\$\config\flashloader\Freescale\FlashK20Xxxx.board.

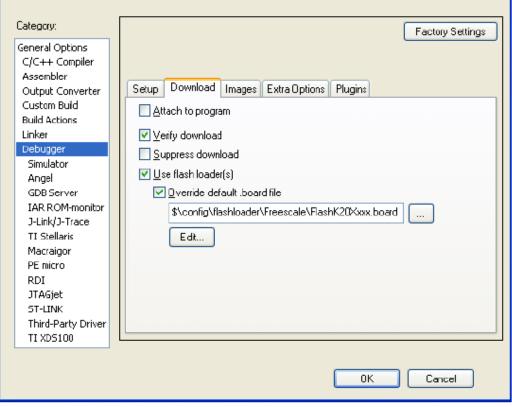


Figure 5.

7. On debugger select driver PE micro.

Category:	Factory Settings
General Options	
C/C++ Compiler	
Assembler	
Output Converter	Setup Download Image: Extra Options Plugins
Custom Build	
Build Actions	Driver 🛛 🗹 🗹
Linker	PE micro 😽 main
Debugger	
Simulator	S <u>e</u> tup macros
Angel	Use macio file(s)
GDB Server	
IAR ROM-monitor	
J-Link/J-Trace	
TI Stellaris	
Macraigor PE existence	Devi <u>c</u> e description file
PE micro	Ovenide default
RDI	
JTAGjet	\$TOOLKIT_DIR\$\CONFIG\debugger\Freescale\iok20xxxxV2.

Figure 6.

Using Low Power modes on Kinetis family, Rev. 0

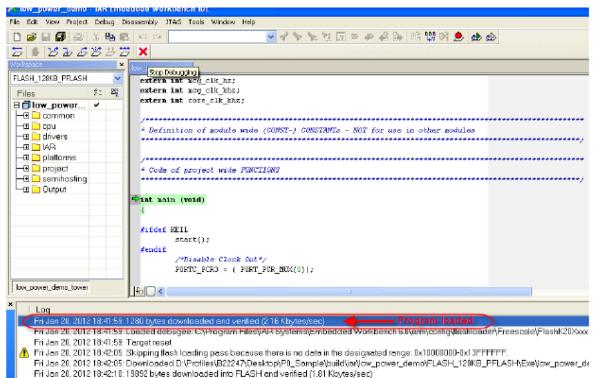


8. To program and debug the application click on the debug icon.





9. Verify that the program was successfully downloaded.



Fri Jan 20, 2012 18:42:10: Loaded debugge: D\Profiles\B22247\Desktop\P0_Sample\build\jaAlow_power_demo\FLASH_128KB_PFLASH\Exe\low_pow Fri Jan 20, 2012 18:42:10: CPU reset by debugger. Fri Jan 20, 2012 18:42:10: CPU reset by debugger.



Using Low Power modes on Kinetis family, Rev. 0



10. At this point we have successfully downloaded the application, now we need to exit and close the debugger by clicking on the X mark as shown in the following screen below:

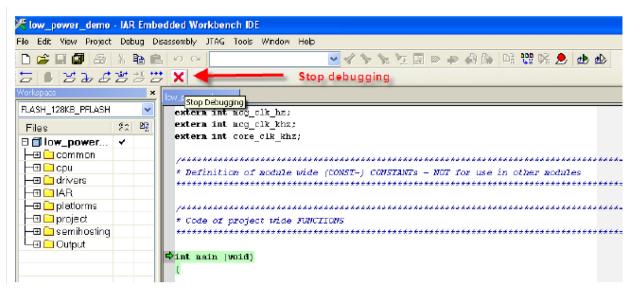


Figure 9.

NOTE

The purpose of stopping the debug session is due to higher current consumption measurements when the debugger is active.

IAR and J-link

For proper operation you must use: J-Link DLL V4.36i

You can download it from the following site:

http://www.segger.com/jlink-software.html?step=1&file=JLink_436kl

IAR must be configured as follows:

- 1. Open your IAR workspace file and select a flash configuration for the project.
- 2. Open IAR Embedded Workbench IDE.
- 3. Open the *low_power_demo* workspace.

To do this you can drag and drop the "low_power_demo.eww" located on the following path:

{Demo Installation path}\build\iar\low_power_demo

4. In the project options select General options.



🔀 low_power_demo - IAR Embedded Workbench I	DE	
File Edit View Project Tools Window Help		
		- 🗸 📏
Workspace	×	common.h io.c
FLASH_128KB_PFLASH	~	*
Files	8:: 📴 🔼	* Notes:
□ low_power_demo_tower - FLASH_128		*
	Options	



5. Select Device *MK20DX128xxx5* on the project options as shown below:

Category:	
General Options	
C/C++ Compiler	
Assembler	
Output Converter	Target Output Library Conliguration Library Options MISRA-C:200
Custom Build	
Build Actions	- Processor variant
Linker	
Debugger	🔿 Core Cortex-M4 🗸
Simulator	
Angel	O Device Freescale MK20DX128xxx5 □
GDB Server	
IAR ROM-monitor	

Figure 11.

6. In the Debugger category: Uncheck "Use flash loaders".



Category:	Factory Settings
General Options	
C/C++ Compiler	
Assembler	
Output Converter	Setup Download Images Extra Options Plugins
Custom Build	Attach to program
Build Actions	
Linker	✓ Verify download
Debugger	Suppress download
Simulator	
Angel	Use flash loader(s)
GDB Server	<u>O</u> verride default .board file
IAR ROM-monitor	\$TOOLKIT_DIR\$\config\flashloader\Freescale\Flas
J-Link/J-Trate	
TI Stellaris	E dit
Macraigor	
PE micro	



7. In J-link/J-trace Connection window: Select JTAG interface.

Category:					Factory Setting
General Options					
C/C++ Compiler					
Assembler					
Output Converter	Setup Connection	Breakpoints			
Custom Build	Communication-				
Build Actions	O USB:	Device 0	~	Serial no:	
Linker	O ICP/IP:	IP address			
Debugger		IF address	×		
Simulator	IP address:	aaa.bbb.ccc.ddd		Serial no:	
Angel	Interface	⊂JTAG scan chai]	
GDB Server IAR ROM-monitor	Interrace			with multiple to each	
J-Link/J-Trace	⊙ <u>J</u> TAG			with multiple targets	
TI Stellaris		TAP numb	er:	0	
Macraigor	U <u>s</u> wD	Scan d	hain d	contains non-ARM o	levices
PE micro		Preci	edin	g bits: 0	
RDI					
JTAGjet	Log <u>c</u> ommunic	ation			
ST-LINK					
Third-Party Driver	\$PRUJ_DIR\$	cspycomm.log			

Figure 13.

8. To program and debug the application click on the debug icon.



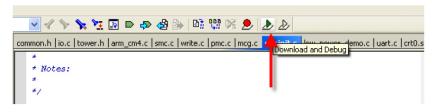
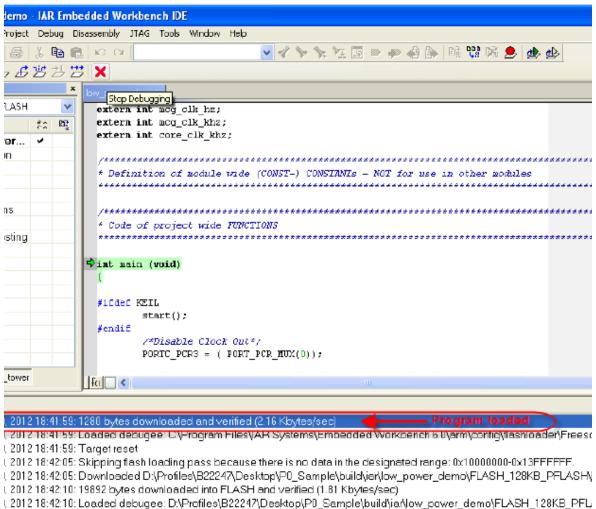


Figure 14.

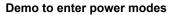
9. Verify that the program was successfully downloaded.



- 1, 2012 18:42:10: CPU reset by debugger.
- , 2012 18:42:10:

Figure 15.

10. At this point we have successfully downloaded the application, now we need to exit and close the debugger by clicking on the X mark as shown in the following screen:





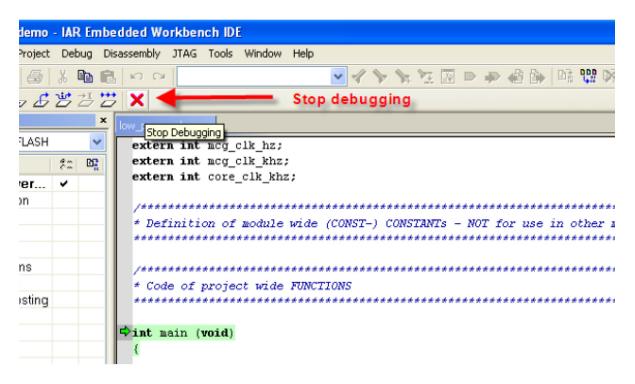


Figure 16.

NOTE

The purpose of stopping the debug session is due to higher current consumption measurements when the debugger is active.

CW MCU V10.1 and OSJTAG/ P&E Multilink

For proper operation with CW10.1, it must be configured as follows:

Update the flash programmer:

1. Click on Flash programmer and import "Flash Task".



ile Edit Refactor Navigate Search Project	
	Run Profiler PEMicro Processor Expert Window Help
📬 • 🗐 🗁 🗄 🗄 🖉 💣 🗄 🖾 🗄	😥 • 🖩 • @ • 🗹 • 🎯 • 🔨 • 👀 • @
🖸 CodeWarrior Projects 🐰	Flash Programmer
📳 🞝 🔶 🗢 🕷	E & P File Name V
File Name 🔺	Size Type
💼 adc_demo 💼 hello_world 🛅 hscmp	Click on Flash programmer (drop down
	arrow) and select "Import Flash Task"

Figure 17.

2. Open the ARM folder, select *K20DX128.xml* and click Open.

Open	? 🛛
Look jn:	🖻 ABM 🗲 🔤 📝 🔝 -
My Recent Documents Desktop My Documents	K9F2G08X0A.xml K20DN512Z.xml K50DX256Z.xml K70FX512.xml K10DN32,xml K20DX32,xml K51DN256Z.xml K70FX512.xml K10DN64,xml K20DX128.xml K51DN512Z.xml K51DN512Z.xml K10DN512Z,xml K20DX128.xml K51DX256Z.xml K51DX256Z.xml K10DN512Z,xml K20DX128.xml K51DX256Z.xml K51DX256Z.xml K10DX32,xml K20DX256Z.xml K53DN512Z.xml K53DX256Z.xml K10DX64,xml K20FX512.xml K53DX256Z.xml K53DX256Z.xml K10DX128,xml K20FX512.xml K60DN512Z.xml K60DN126Z.xml K10DX128,xml K20FX512.xml K60DN512Z.xml K60DN512Z.xml K10DX256Z,xml K30DX128Z.xml K60DX256Z.xml K60FN1M0. K10DX256Z,xml K30DX128Z.xml K60FN1M0. Type K10FX512,xml K30DX128Z.xml K60FN1M0. Type K10FX512,xml K40DN128Z.xml K60FN1M0. Size: Size: K20DN64,xml K40DX128Z.xml K61FN1M0. Size: Size: Size: K20DN128,xml K50DN512Z.xml K61FN1M0. K61FN1M0. Size: Size:
	File <u>n</u> ame: K20DX128.xml
My Network	Files of type: *.xml Cancel

Figure 18.

3. Select FLASH configuration.



C/C++ - CodeWarrior Development Studio		
File Edit Refactor Navigate Search Project Run Profiler	PEMicro · P	rocessor Ex
<mark>□ - □ △ ゐ ☎ ∦</mark> ▲ ∦ - Ⅲ - ■ - □ □ □ ↓ - २ - २ - २ -	• 🚳 •	• 💣 • @
CodeWarrior Projects		- 8
🔠 縄 🗇 🔿 👰 🗐 🤣 🔑 File	Name	
File Name 🔺	Size	Туре
<pre> i adc_demo i hello_world i hello_world</pre>		
E 🤗 🔹 MK20DX128_INTERNAL_FLASH		
Binaries MK20DX128_INTERNAL_RAM		
Includes MK20DX128_INTERNAL_FLASH		_
FLASH_OSJTAG.launch	24 KB	
FLASH_OSJTAG.pemicro	1 KB	
FLASH_Segger.launch	17 KB	
MK20DX128_INTERNAL_FLASH		
Figure 19.		

4. Build the project by clicking on the hammer icon as shown below:



🥙 C/C++ - CodeWarrior Development Studio		
File Edit Refactor Navigate Search Project Run Profiler	PEMicro · Processor E:	xpert Window Help
<mark>12 - 8 ≙ 8 22 <i>8</i> 14 <i>9</i> - 111 - 8 11 12 - 12 - 12 - 10 + 10 - 10 - 10 - 10 - 10 - 10 - 10</mark>	- 🚳 - 💽 - (
CodeWarrior Projects	- 8	
🔠 🛃 🗇 🗘 🛱 🗦 🌮 File	Name 🎽	Build project
File Name 🔺	Size Type	
dt_demo finite demo finite demo		
💼 hscmp = 🖉 low_power_demo : MK20DX128_INTERNAL_FLASH		
Binaries		
Includes		
FLASH_OSJTAG.launch	24 KB	
FLASH_OSJTAG.pemicro	1 KB	
ELASH_Segger.launch	17 KB	
MK20DX128_INTERNAL_FLASH		
Communication Communication Communication Communication Communication		
	18 KB	
RAM_OSITAG.iaunth	10KB 1KB	
RAM_Segger.launch	16 KB	
SaAnalysispointsManager.apconfig	1 KB	
E Cources		

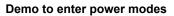


5. Click on the drop down arrow to select the corresponding debug configuration.

SC/C++ - CodeWarrior Development Studio			
File Edit Refactor Navigate Search Project Run Profiler PEMicro	Processor E:	zpert Window Help	
11 - 2 - 4 30 12 8 8 14 1 9 - 10 - 10 11 12 - 9 - 10 - 10 - 10 - 10 - 10 - 10 - 10	9 - 💽 - (♂ • % • ⊗ • @ •	🏇 - 💽 - 🍇 - 👘 🥭 🖉
E CodeWarrior Projects 😒			C 2 RAM_OSJTAG
File Name Size Image: adc_demo Image: adc_demo Im	с Турс	Select Debug configurations	Debug As Debug Configurations Organize Fevorites
E 😂 low_power_demo : MK2CDX126_INTERNAL_FLASH E 💥 Bharies			

Figure 21.

6. Select FLASH_OSJTAG, go to Debugger tab, open Download section, uncheck "Perform Standard Download".



eate, manage, and run conf	-					1
Xownload an application to a target, th	han debug or run tha appl	lication.				- A
1 🗊 🗶 🕞 🏇 •	Name: FLASH_CGJTAG	s 🎋 Debugger 🛛 🦉 Sau	rce) 📧 Envit	ranmant) 📼 Co	mmon) 💣 Trace and Profile)	
CodeWarrior Attach	Debugger options					
CodeWarrior Connect	Debug Exceptions	Interrupts Download P	C Other I	Executables Sy	mbolics OS Awareness	
- C RLASH_OGJTAG	Select download opti	ons; subsequent of tions are	used for rest	art and when svi	mbolics are cached:	
	Perform Standa					
C RAM_Segger	A	3	5ut	bsequent		
🎰 🕨 Launch Group	Program Secti		Downlo	oad Verify	Scient All	
Select	Unchecktal		v 		Deselect Al	
FLASH_OSJTAG	(C) idalized Da					
0	Uninitialized Da					
	Nobe: Standard	download availability depend	is on <u>cannect</u>	<u>ian type</u>		
	Execute Tasks					
	Name	Task Type	First	Subsequ	Add	
	CopyOfK20DX12	8_Ta Flash Programme	. 🗹	V	Remove	
					1 Section 1 M	
					Up	

Figure 22.

7. Check the box for Execute Tasks, select the current task and click on Remove.

NP



manage, and run conf d an application to a target, ti	hen debug or run the applicati	ion.		3
🗶 🔁 🏇 -	Name: FLASH_OSUTAG			
er text	🕒 Maio 🔯 Arguments 🗟	E Debunner Day Sou	rce] 🌆 Environment] 🔚 Co	mmm 🕷 Trace and Profile
CodeWarrior Attach	- Debugger options	,,,,		
CodeWarrior Connect CodeWarrior Download	Debug Exceptions Int	terrupte Download pg	C Other Executables Sy	mbolice OS Awareness
E FLASH_OGJTAG	Select download ontions:	subsequent potions are	used for restart and when sy	nholics are rached:
 FLASH_Segger RAM_OSJTAG 	Perform Standard D		and the research and which sys	
RAM_CODTAG		First	Subsequent	
Launch Group	Program Section	Download Verify	Download Verify	Select All
	Executable			Deselect All
	Constant Data Initialized Data	V		
	Uninitialized Data			
	Notes Standard dow	nload availability depend	s on connection type	
	Diecute Tasks			
	Name	Task Type	First Subsequ	Add
Check box		a Hash Programme		Remove
(5)				Remove current ta
-				6
				0
thed B of S tems				Apply Rev
thed a or a tems				

Figure 23.

8. Click on Add button and select K20DX128.



P Debug Configurations		X
Create, manage, and run cor Download an application to a target,		Ś
Image: Second	Name: FLASH_OSJTAG Add Download Task Add Download Task Trace and Profile Target Tasks CopyOR(200X128_Task_Mem1 CopyOR(200X128_Task_Mem1 CopyOR(200X128_Task_Mem1 CopyOR(200X128 Receive an initial launch Execute an initial launch Execute an successive runs CK Cancel Add Add Add Add Remove Task Type First Subsequ Add Remove Task Type First Subsequ Add Remove Task CK Cancel CK CACCANCEL CK CK CACCANCEL CK CK CK CK CK CK CK CK CK C	

Figure 24.

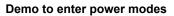
9. Click Apply and Debug.



eate, manage, and run con ownload an application to a target,	-	stion.				Ŕ
Vpe filer text C CodeWarter Attach C CodeWarter Connect C CodeWarter Counted C CodeWarter Counted C FLASH_Segger C RAM_Segger Launch Group	Debugger options Debug Exceptions Debug Exceptions Perform Standard Program Sector Exactable Constant Dete Initialized Deta Uninitiated Deta Notes Standard de	First Download Verify	Other sed for res Down	Executables Syn start and when syn ibsequent iond Verfy	nbolics 05 Awareness	
	Execute Tasks					
	Name K20DXL28	Task Type Flash Programme	First	Subsequ	Add	
					Remove	
					- Dava	
					Acoly	Revert

Figure 25.

10. Verify download.



Debug - low_power_demo/Sources/low_power_demo/low_power_c	lemo.c - CodeWarrior Devel	opment	Studio				
File Edit Reflector Navigabe Search Project Run Profiler PEMoro Processo	r Expert Window Help						
[월•][월][梁 <i>왕</i>][종][編][》·開•][종 •]@ [월-][-박야-아-	• * • O • 0 •			1 🥔	* • \$	E 🌮	Debug 🔤 C/C++
🕸 Debug 🕄 🔪 🦉 🎲 😌 🕪 💷 🛤 🔍 🔅 🖻	5 1 元 2 3 伊田市	<i>m</i> • [∞]		99• Veri	S % Bro	🛙 Cac 🛛 🕅 Re	g 🔁 Mod 🗖 🗖
E RLASH_OSITAG [Codel/Verior Download]					Æ <	4 🖂 🔀 •	. # × % ∼
B- # ARM Processors, k20_jow_power_damo.afx (Suspended) B- # Thread (30: 0x0) (Suspended: Signal Halt' received, Description: User In	alted thread.)			Name		Value	
3 main() bt/Profiles(222247)Desktop(P0_Sample)sr.dprojects(Jow_D) 2 start() ot/Profiles(222247)Desktop(P0_Sample)sr/dpo/start.cr49 10 LoumyFn() bt/Profiles(222247)Desktop(P0_Sample)sr/dpo/com- Dignet(Jest)22247)Desktop(P0_Sample)bulkpul/gw_Dower_deno(//K2005)	ower_denic/jow_power_demo.c:10 0x0000072e grt0.s:40 0x00000a58						
				<			3
							A
							-
¢			>	<			(2)
🗷 юм_рониг_dense. 🗵				amentaly	22 🚦 Outine		
int main (woid)		^			Enter location her	• •	8 h 🕟 🔨
Program execution stop	ne at main	-			nain:		<u>~</u>
#ifdef KEIL	o atman		000 110	024e4:			F FORT F
start(); If you are here the board	d has been successful	ly 👘		1024e6:			- I PORT_P
#endit programmed		~	000	D24e8:	BOVE CO.;	¥0x200c	<u> </u>
		3	Ļ				
🕒 Concole 🖾 🖉 Tasks 🚚 Remote Systems 🔞 Target Tasks 🛣 Problems	🕐 Executables 📙 Memory Brow	eser 🚺 f	femory		× .	ii 🚮 🗹	•••••
RACH_COILAG Info: Windows NT Generated.							10
Info: PGE Interface detected - Flash Version 30.14 Info: Device is X20D0128. Info: Mode is In-Circuit Debug.							
Info: CPU reset by debugger.							
							×
SI E 114		1					2
				i.			1

Figure 26.

11. In order to measure current consumption properly, terminate debug session by clicking on the red square button.

NP



Debug - low_power_demo/Sources/low_power_demo/low_power_demo.c - CodeWarrior Development Studio
File Edit Refactor Navigate Search Project Run Profiler PEMicro ProcessorExpert Window Help
i mi - □ ≧ Ba i ma ∦ i ∲ i ka i ∦ - □ - i @ - 救 - D - Qa - i g - □
(参 Debug 23) 🦓 🤣 🕪 🗉 🔳 🖬 🔍 👁 💀 🖉 🖶 📅 🏙 👖 - 🍸 🗖
FLASH_OSJTAG [CodeWarrior Download] ARM Processors, k20_low_power_demo.afx (SuspeTerminate (Ctrl+FZ)) Thread [ID: 0x0] (Suspended: Signal Halt' received. Description: User halted thread.) 3 main() D:\Profiles\B22247\Desktop\P0_3 ample\src\projects\low_power_demo.low_power_demo.c;104 0x000024e4
2 start() D: \Profiles\B22247\Desktop\P0_i emple\src\cpu\start.c:48 0x0000072e 1 DummyFn1() D:\Profiles\B22247\Desktop\P0_Sample\src\cpu\cw_crt0.s:40 0x00000a53
🛄 D:\Profiles\B22247\Desktop\P0_Sample\build\cw\bw_power_demo\MK20DX128_INTERNAL_FLASH\k20_low_power_demo.afx (1)
Click to terminate debug
session

Figure 27.

CW MCU V10.1 and JLink

Follow the same steps as indicated above (for CW MCU V10.1 and OSJTAG/ P&E Multilink), except that in step 6, you must select **FLASH_Segger**.

Select FLASH_Segger, go to Debugger tab, open Download section, uncheck "Perform Standard Download".

🔑 Debug Configurations	X
Create, manage, and run configurations	1
Download an application to a target, then debug or run the application.	3
C T	
🗅 🗄 🗶 🖻 🍜 - Name: FLASH_Segger	
hope Filter last	_
CodeWarrier Attach	
CodeWantor Connect CodeWantor Connect	<u> </u>
CodeWarrior Download Debug Exceptions Interrupts Download PIC Other Executables Symbolics OS Awareness	
FLASH_OSDTAG Select download options; subsequent options are used for restart and when symbolics are cached: FLASH_Sequent	
C RAM_COUTAG	
RAM_Segger 🛖 Fist Subsequent	
Program Section Download 3)prfy Download Verify Scient Al	
Select Uncheckeratele Deselect Al	
FLASH_Segger (4) Initialized Data	=
Note: Standard download availability depends on connection type.	
▼ Execute Tasks	
Nome Task Type First Subsequ Add	
CopyOfK200X128_Ta Flash Programme V V Remove	

Figure 28.

7.2 Steps to run the demo

Once you have successfully loaded the application to the device and stopped the debugger, you can proceed with the following steps:

- 1. Open the Terminal application from P&E.
- 2. Set the port to Port: USB COM, Baud: 115200, Parity: None, Bits: 8 Click on "Open Serial Port".

	Port: USB COM 💌	P&E Homepage		Duplex: Ful 💽	
ν	Baud : 115200 💌	Open Serial Port	Doss Port	Download File to ComPart	Freescale
	Pority: None 💌	Opensenarruit	LIUSSITU	Chor Delay (ms) : 🗾 🦻	Alliance Member
micro	Bits: 🛛 💌			EOL Delay (ms) : 15	
erminal Window:	Clear Window				

Figure 29.

3. Make sure the port is successfully opened.



Terminal Window a	opplication v2.05				
	Part : USB COM 🖃	PEE Homepage		Duplex : 🔽 💌	~
$ \nu $	Boud : 115200 💌	Deen Serial Port	Close Part	Download File to ComPort	🚀 freescale
	Parity : None 💌		L	Char Delay (ms) : 5	Alliance Member
micro	Bits: 8 💌 Open. •	—		EOL Delay (ms) : 15	
erminal Window:	Clear Window				

Figure 30.

- 4. Now click on the reset button of your board.
- 5. If you look to the Terminal Window you should have all available options displayed.

🖇 Terminal Window applic	ation v2.05		
	ıd : 115200 💌	Open Serial Port	Close Port
Terminal Window:	Clear Window		
LLWU PE3 = 0x00, LLWU ME = 0x00, LLWU F2 = 0x00, LLWU F1LT1 - 0x00, LLWU RST = 0x02 * Low Power Mo * Low Power Mo *	iter	¥. * *	
Copyright 2007, F&E Microcompute	er Systems, All Rights Reserved	http://www.pemicra.com	
<	III		>

Figure 31.

6. Follow the instructions on the menu.





Example:

To enter Stop Mode:

• Type letter B in the terminal window.

The following message will be displayed:

🛠 Terminal Window application v2.05					
P	ort : USB CDM 🗾 🛛 🔤	Homepage			
Bal Par		Open Serial Port	Closs Port		
micro 🛛	Rits: 🛛 🔽 Open.			EO	
Terminal Window:	Clear Window				
LLVU NE = 0×00, LLVU F2 = 0×00, LLVU F1LT1 = 0×00, LLVU RST = 0×02	LLVU F1 = 0x00 LLVU F3 = 0x00 LLVU FILT2 = 0x00				
* Lov Pover Mode Test *					
Select the mode to en 0 to enter ULLS0 no F 1 to enter ULLS0 2 to enter ULLS1 3 to enter ULLS3 5 to enter ULLS3 5 to enter ULPS 7 to enter ULPR 8 exit ULPR 9 to enter ULPV A to enter ULPV B to enter STOP bPress any key to ent	POR				
Copylight 2007, P&E Microcomput	er Systems, All Rights Reserved.	http://www.pemicro.com	m		
<	Ш)	>	

Figure 32.

• Press any key.

The following message is displayed:



🚿 Terminal Window application v2.05						
P	ort : USB COM 🔽 🦳 🤒	E Homepage				
	ud : 115200 🔽	Open Serial Port	Close Port			
MICRO E	ity : None 🔽 Bits: 8 🔽 Open.			E C		
		1		20		
Terminal Window:	Clear Window					
LLWU ME = 0x00, LLWU F2 = 0x00, LLWU FILT1 = 0x00, LLWU RST = 0x02	LLWU F1 = 0×00 LLWU F3 = 0×00 LLWU FILT2 = 0×00	÷				
* Low Power Mode Test *						
Select the mode to er Ø to enter VLLSØ no I 1 to enter VLLSØ 2 to enter VLLS1 3 to enter VLLS2 4 to enter VLLS3 5 to enter VLPS 6 to enter VLPS 7 to enter VLPR 8 exit VLPR 9 to enter VLPW A to enter VLPW A to enter STOP bPress any key to ent Press SW2 or SW3 to	er Stop	*				
Copyright 2007, P&E Microcomputer Systems, All Rights Reserved. http://www.penicro.com						
<				>		

Figure 33.

This means we are currently in stop mode.

• To measure power consumption (depending on your board) you should take off the Jumper that corresponds to MCU power connections.

This is J25 on the TWR-K20D50M Rev. C

NOTE

If you are unsure of the corresponding jumper please refer to your tower board user manual.

• Now measure the device consumption between these terminals:

You must be measuring approximately 320µA.

• If you want to exit this low power mode and enter another one press SW2 or SW3:

Using Low Power modes on Kinetis family, Rev. 0

Conclusion

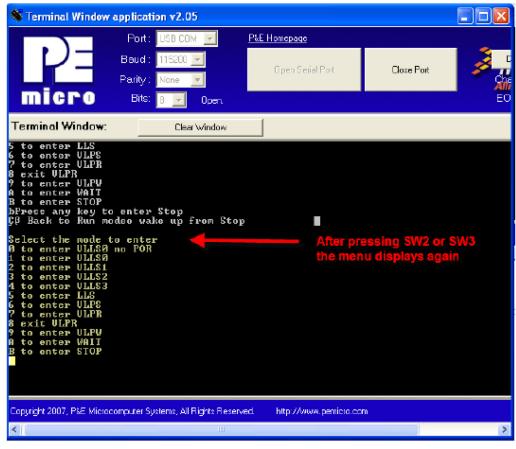


Figure 34.

Now you can follow the steps again to enter a new power mode.

NOTE

You can exit any power mode via reset.

8 Conclusion

When designing for low power you must take into consideration both hardware and software, as both variables are equally important to design a successful power efficient application.

Remember the basics, set pins to a known state, higher core frequency and higher temperature both translate to higher power consumption, disable clocks to unused modules.

9 References

The chapters in this application note contain a summary of the most important details of each topic.

For more details on any of the topics of this document you can refer to the following:

Information about power consumption:

Using Low Power modes on Kinetis family, Rev. 0



References

• Refer to the device datasheet

Search for "Power consumption operating behaviors"

Information and details about power modes:

• Refer to the device Reference Manual

Search for chapter "Power Management"