AN14736

Power Management for MCX E24x

Rev. 1.0 — 22 July 2025 Application note

Document information

Information	Content
Keywords	AN14736, MCX E24x, power management, MCX E24x family
Abstract	This application note discusses how to use the power management system for the MCX E24x family. The document also outlines best practices for using each of the power modes available on the MCX E24x family.

1 Introduction

Currently, the power consumption of devices and the implications around designing for low power are common topics. The MCX E24x family includes internal power management feature. This feature can be used to control the power usage of the microcontrollers and assist reaching the targets of embedded designs. This application note discusses how to use the power management system for the MCX E24x family. The document also outlines best practices for using each of the power modes available on the MCX E24x family.

2 Overview of power modes

The typical power modes in legacy cores and other embedded systems are Run, Wait, and Stop. The Arm Cortex M4 and M0+ power modes are Run, Sleep, and Deep Sleep. The extended power modes and their relationship with typical and Arm Cortex M4 and M0+'s modes are depicted in Figure 1.

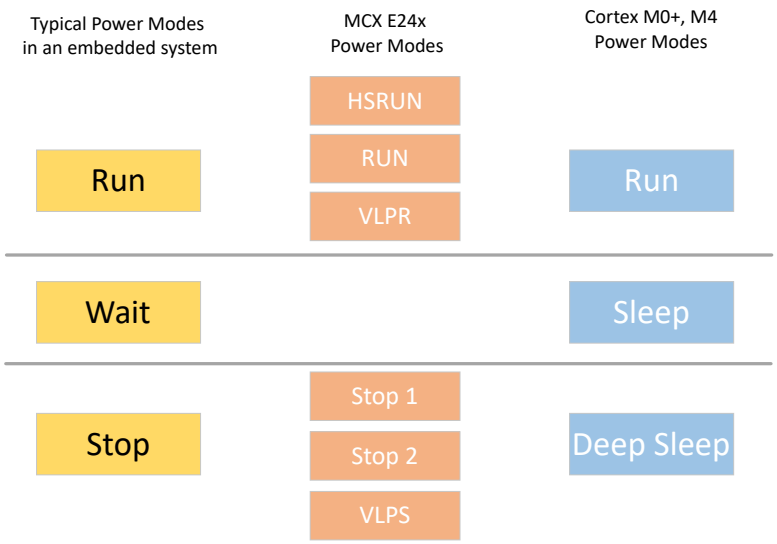

Figure 1. Power modes comparison

2.1 Arm Cortex-M4 and M0+ power modes implementation

The Arm Cortex-M4 and M0+ cores have three primary modes of operation: Run, Sleep, and Deep Sleep. The wait for interrupt (WFI) instruction invokes Sleep and Deep Sleep modes for the MCX E24x chips.

Figure 2 shows the Cortex-M4 and M0+ architecture for low-power implementation. The Sleep and Deep Sleep modes are architected in hardware to control the clock to the core and interrupt controller. When entering Sleep mode, the nested vectored interrupt controller (NVIC) logic remains active. Either the interrupts or a reset can wake the core from sleep. When entering Deep Sleep mode, an asynchronous wake-up interrupt controller (AWIC) is used to wake the MCU from a select group of sources. These sources are described in the *MCX E24x Series Reference Manual* (document MCXE24XRM). For more information, refer to the *Asynchronous Wake-Up Interrupt controller (AWIC) configuration* section within the *Core Overview* section.

Using the sleep-on-exit feature, the Arm Cortex cores have one more way to enter low-power modes. The system control block has a register called system control register (SCR) that contains several control bits related to sleep operation. The SLEEPDEEP bit selects whether Sleep or Deep Sleep mode is selected. Setting the SLEEPONEXIT bit to 1, enables the processor to immediately enter sleep modes when it completes the execution of all exception handlers. For more information, refer to Arm Cortex-M4 Devices Generic User Guide and Arm Cortex-M0+ Devices Generic User Guide.

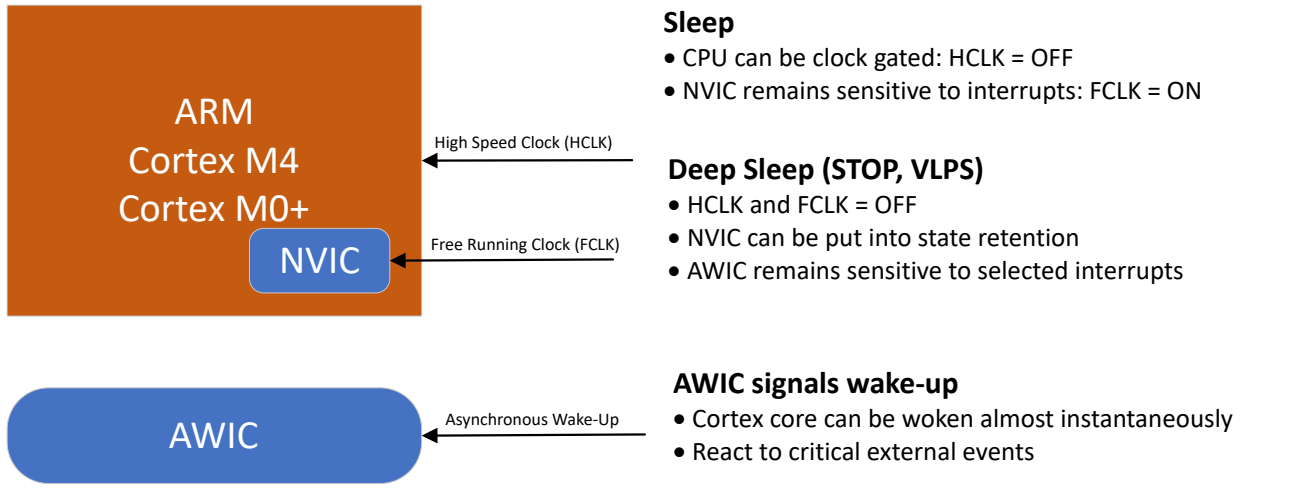

Figure 2. Arm Cortex M4 and M0+ architecture

3 Power modes description

MCX E24x provides multiple power options allowing users to optimize power consumption for the level of functionality needed.

Depending on the user application's requirements, various power modes are available that provide state retention, partial power down, or full power down on certain logic and/or memory. The input/output (I/O) states are maintained during all power modes. For more information about module functionality in different power mode, refer to Section 6 or *MCX E24x Series Reference Manual* (document MCXE24XRM). Table 1 compares the available power modes.

Table 1. MCX E24x power modes

MCX E24x mode	Description	Core mode	Normal recovery method
Normal Run	The default mode out of reset. The on-chip voltage regulator is on (the internal supply is fully regulated).	Run	-
High Speed Run (HSRUN)[1]	HSRUN mode allows the maximum performance of the chip. In this mode, the chip can operate at a higher frequency as compared to Normal Run mode but with restricted functionalities. Internal clocking requirements should be met.[1]	Run	-
Very Low-Power Run (VLPR)	The 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 memory access mode (1 MHz). • LVD is off. • SIRC provides a low-power 4 MHz source for the core, bus, and peripheral clocks.	Run	-
Very Low-Power Stop (VLPS) via WFI instruction	Places the chip in a static state with low voltage detect (LVD) operation off. This is the lowest-power mode in which pin interrupts are functional. • Some peripheral clocks are stopped.[2] • LPTMR, RTC, and CMP can be used. • NVIC is disabled.	Deep Sleep	Interrupt (for reset)

Table 1. MCX E24x power modes*...continued*

MCX E24x mode	Description	Core mode	Normal recovery method
	• AWIC is used to wake from interrupt. • Core is gated off. • All SRAM is operational (content is retained and I/O states are maintained).		
Stop 1 (via WFI instruction)	Places the chip in a static state. LVD protection is maintained. • NVIC is disabled. • AWIC is used to wake up from interrupt. • Some peripheral clocks are stopped.[2] • The core clocks, system clocks, and bus clock are all gated.	Deep Sleep	Interrupt (for reset)
Stop 2 (via WFI instruction)	Places the chip in a static state. LVD protection is maintained. • NVIC is disabled. • AWIC is used to wake up from interrupt. • Some peripheral clocks are stopped.[2] • Only the core and system clocks are gated, but the bus clock remains active. The bus controllers and bus targets clocked by the system clock enter Stop mode. However, the bus targets clocked by the bus clock remain in Run mode. The clock generators in the SCG and the PMC's on-chip regulator also remain in Run mode.[3]	Deep Sleep	Interrupt (for reset)

[1] Core and system clock must be 112 MHz or less. The bus clock must be programmed to 56 MHz or less and an integer divider of the core clock. The flash clock must be programmed to 28 MHz or less. The core clock to flash clock ratio is limited to a maximum value of 8.
[2] For more details, refer to Section 6.
[3] The following can initiate an exit from STOP: Reset, an asynchronous interrupt from a bus controller (valid also for STOP1). A synchronous interrupt from a bus target clocked by the bus clock (valid only for STOP2).

3.1 Power mode transitions

Figure 3 shows the power mode transitions. Any reset always brings the chip back to Normal Run mode. Depending on the needs of the user application, various Stop modes are available that allow the state retention and partial power down or full power down of certain logic and/or memory. The I/O states are held as well as registers retain their content in all modes of operation.

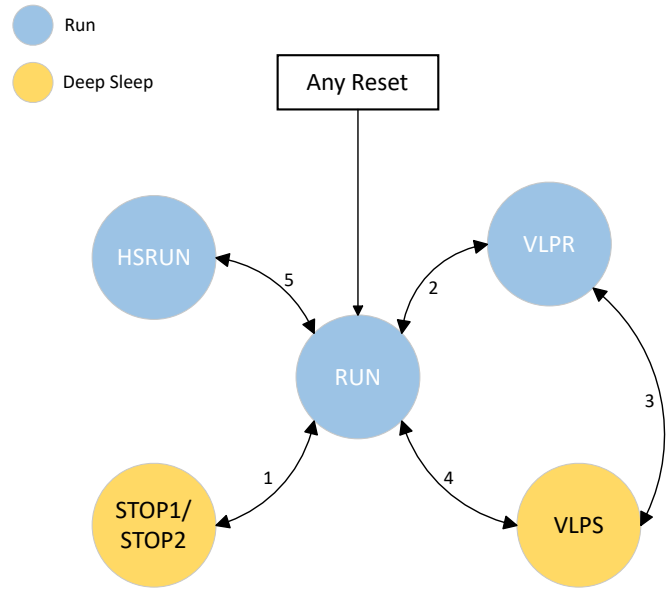

Figure 3. Power mode state transition diagram

Table 2 defines trigger for the various state transitions as shown in Figure 3.

Table 2. Power modes transition triggers

Transition	From	To	Main transition trigger command/condition
1	RUN	STOP	SMC_PMCTRL[RUNM] = 00, SMC_PMCTRL[STOPM] = 000[1] following a WFI instruction.[2]
	STOP	RUN	Interrupt[3] or reset
2	RUN	VLPR	Set SMC_PMPROT[AVLP] = 1, SMC_PMCTRL[RUNM] = 0b10[4]
	VLPR	RUN	Set SMC_PMCTRL[RUNM] = 0b00 or reset.
3	VLPR	VLPS	SMC_PMCTRL[STOPM] = 0b010, following a WFI instruction.[2]
	VLPS	VLPR	Interrupt. ***Note:*** *If VLPS is entered directly from RUN (Transition 4), the hardware forces exit back to RUN and does not allow a transition to VLPR.*
4	RUN	VLPS	SMC_PMPROT[AVLP] = 1, SMC_PMCTRL[STOPM] = 0b010, following a WFI instruction.[2]
	VLPS	RUN	Interrupt and VLPS mode are entered directly from RUN or reset.
5	RUN	HSRUN	Set SMC_PMPROT[AHSRUN] = 1, SMC_PMCTRL[RUNM] = 0b11.
	HSRUN	RUN	Set SMC_PMCTRL[RUNM] = 0b00 or reset.

[1] PSTOPO register must be configured to select the Stop mode variant (STOP1 or STOP2).
[2] Sleep-now (WFI instruction) or sleep-on-exit modes entered with SLEEPDEEP set, which is controlled in the system control register in Arm core.
[3] Module capable of providing an asynchronous interrupt to the device.
[4] Core, system, and bus clock must be 4 MHz (maximum) and flash clock must be set to 1 MHz (maximum). Also, all asynchronous clock sources are restricted to 4 MHz as configured SCG_SRICDIV.

4 Clock operation in low-power modes

There are several clock sources available in the MCU. To conserve power, most module clocks can be turned off by configuring the CGC field of the peripheral control register in the PCC module. These fields are cleared after any reset, which disables the peripheral clock of the corresponding module. Before initializing a module, the corresponding field in the PCC peripheral control register needs to be set to enable the module clock. Ensure to disable the module before turning off the clock.

Note: *Before disabling a module, its interrupt and DMA request should be disabled.*

4.1 System clock generator (SCG) clocks

The SCG module generates the clocks for the MCU. Figure 4 shows a block diagram of the SCG module.

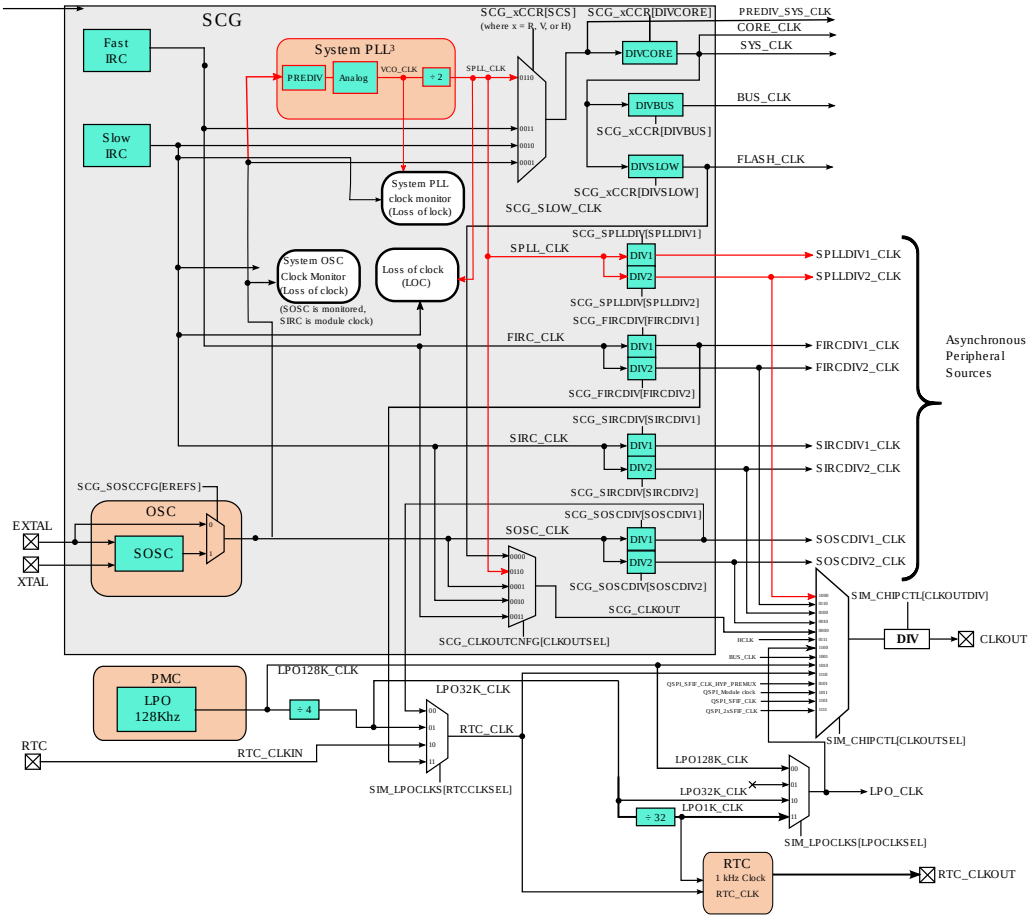

Figure 4. Clocking diagram

The clock generation circuitry provides several clock dividers and selectors allowing different modules to be clocked at a specific frequency for that module. The following Four main clock sources (besides low power oscillator (LPO) module) can be seen on SCG module:

- Fast internal reference clock (FIRC)
- Slow internal reference clock (SIRC)
- System oscillator (SOSC)
- System PLL (SPLL)

For Run modes (HSRUN, normal RUN, and VLPR), different sources can be used to provide clock signal to the core. Figure 5 shows all possible sources for core clock that can be used in different power modes.

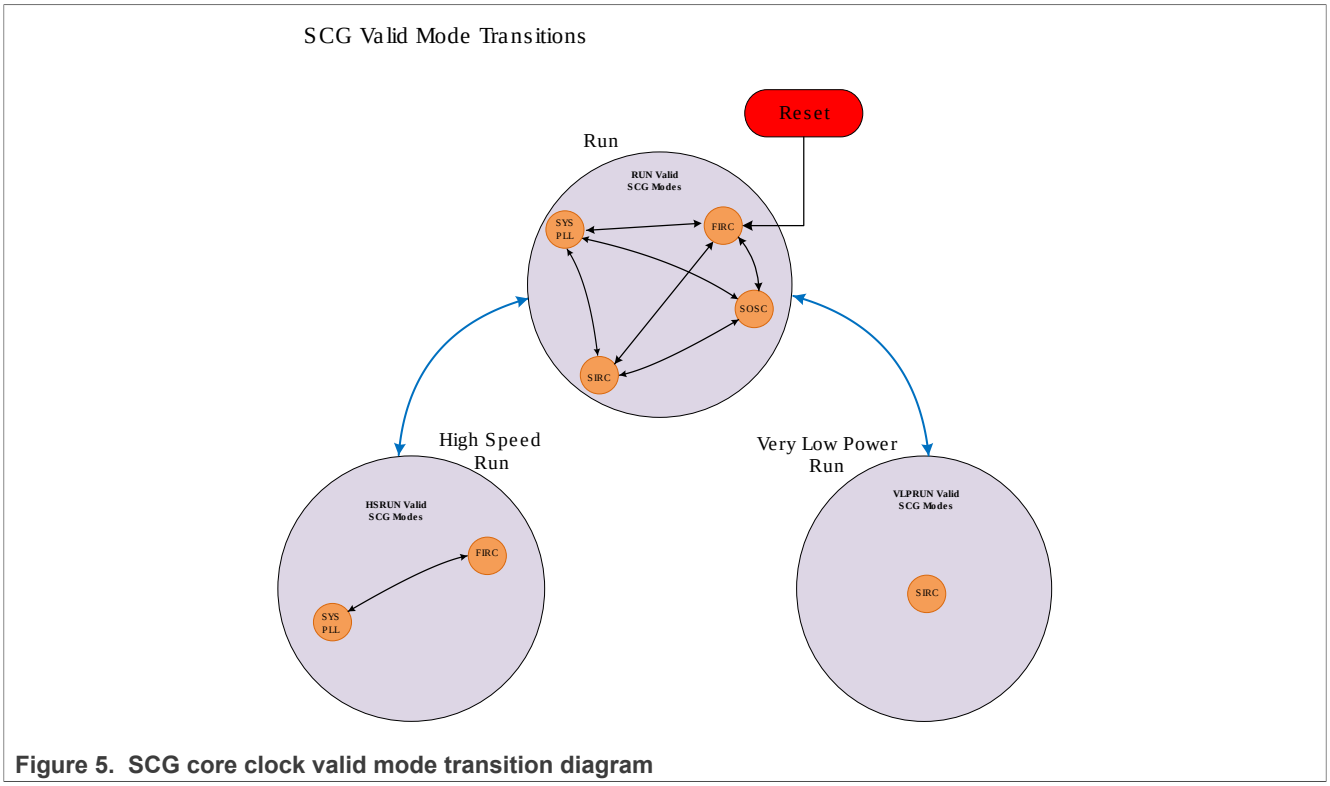

Figure 5. SCG core clock valid mode transition diagram

For other power modes, such as Stop and VLPS, as the core clock is gated off, no source is used.

When entering VLPR/VLPS mode, the software in RUN mode must disable the SPLL and FIRC before making any mode transition.

Before switching clock sources, ensure to meet the requirements listed in the *Internal clocking requirements* and *Module clocks* sections in *MCX E24x Series Reference Manual* (document MCXE24XRM).

4.2 Power management controller (PMC)

An internally generated low-power clock with a typical frequency of 128 kHz can be configured on the power management controller (PMC) module. This clock can be used as a source for modules that operate in low-power modes. Figure 4 shows the low power oscillator (LPO) source and its different derivations.

5 Power mode entry/exit

When entering or exiting low-power modes, the system must confirm an orderly sequence to manage transitions safety.

The SMC manages the system's entry into and exit from all power modes.

5.1 High-Speed Run (HSRUN) mode entry

In HSRUN mode, the on-chip voltage regulator remains in a run regulation state, but with a slightly elevated voltage output. In this state, the MCU can operate at a faster frequency compared to Normal RUN mode.

However, No Flash commands (FTFC) of any type, including CSEc commands, are available when the chip is in this mode.

While in this mode, the following restrictions must be adhered to:

- Modifications to clock gating control bits are prohibited.
- Flash programming/erasing is not allowed.

To enter HSRUN mode (with 112 MHz SPLL as clock source), perform the following steps:

1. Disable PLL. Configure FIRC as the RUN mode and HSRUN mode clock source by writing 0b0011 to SCG_RCCR[SCS] and SCG_HCCR[SCS].
2. Set PMPROT[AHSRUN] to allow HSRUN mode.
3. Write 0b11 to SMC_PMCTRL[RUNM] to enter HSRUN. Now, the system enters HSRUN mode with FIRC configured as the system clock source.
4. Reconfigure the SPLL for 112 MHz and enable it.
5. Switch to PLL as the clock source by configuring SCG_HCCR[SCS] as 0b0110.

To enter HSRUN mode (with 80 MHz as SPLL clock source), perform the following steps:

1. Configure SPLL at 80 MHz in RUN mode and use this clock source in HSRUN mode by writing 0b0110 to SCG_RCCR[SCS] and SCG_HCCR[SCS].
2. Configure PMPROT[AHSRUN] to allow HSRUN mode.
3. Write 0b11 to SMC_PMCTRL[RUNM] to enter HSRUN.

Before increasing clock frequencies, the PMSTAT register should be polled to determine when the system has completed entry into HSRUN mode. The below code snippet shows a basic RUN to HSRUN transition function.

```
void RUN_to_HSRUN (void)
{
    /* Disable SPLL and use FIRC as MCU's source clock */
    scg_disable_spll_enable_firc();
    /* Allow high speed run mode */
    SMC->PMPROT |= SMC_PMPROT_AHSRUN_MASK;
    /* Check if current mode is RUN mode */
    if(SMC->PMSTAT == 0x01)
    {
        /* Move to HSRUN Mode*/
        SMC->PMCTRL = SMC_PMCTRL_RUNM(0b11);
        /* Wait for Transition*/
        while(SMC->PMSTAT != 0x80);
        /* Configure SPLL for 112/80 MHz */
        scg_configure_spll();
    }
}
```

```
static void scg_disable_spll_enable_firc (void)
{
    /* SCG_RCCR register only accepts 32-bit writes */
    /* Use FIRC as clock for CPU and set valid dividers */
    SCG->RCCR = (uint32_t) (SCG_RCCR_SCS(0b11) | /* FIRC */
    /* Core clock is 48MHz / 1 */
    SCG_RCCR_DIVCORE(0) |
    /* Bus clock is Core clock / 1 */
    SCG_RCCR_DIVBUS(0) |
    /* Flash clock is Core clock / 2 */
    SCG_RCCR_DIVSLOW(1));
    /* SCG_HCCR register only accepts 32-bit writes */
    /* Use FIRC as clock for CPU and set valid dividers */
```

```
    SCG->HCCR = (uint32_t) (SCG_HCCR_SCS(0b11) | /* FIRC */
    /* Core clock is 48MHz / 1 */
    SCG_HCCR_DIVCORE(0) |
    /* Bus clock is Core clock / 1 */
    SCG_HCCR_DIVBUS(0) |
    /* Flash clock is Core clock / 2 */
    SCG_HCCR_DIVSLOW(1));
    /* Disable PLL and clock monitors */
    SCG->SPLLCSR &= ~(SCG_SPLLCSR_SPLLEN_MASK |
    SCG_SPLLCSR_SPLLCM_MASK);
}
```

Note: *Flash programming/erasing is not allowed. No FTFC commands of any type, including CSE commands (for CSEc parts), are available when the chip is in this mode.*

Note: *Modifications to clock gating control bits are prohibited.*

5.2 HSRUN mode exit

Transition from HSRUN to RUN mode can be made by either a reset event or setting SMC_PMCTRL to 0b00. In HSRUN mode, the core clock can be set to a maximum of 112 MHz, while in RUN mode, it is limited to 80 MHz. Therefore, it is necessary to lower the core frequency before switching back to RUN mode. The below code snippet shows a basic HSRUN to RUN transition function.

```
void HSRUN_to_RUN (void)
{
    /* Adjust SCG settings to meet maximum frequencies value at Run mode */
    scg_configure_freq_for_RUN();
    /* Check if current mode is HSRUN mode */
    if(SMC->PMSTAT == 0x80)
    {
        /* Move to RUN Mode*/
        SMC->PMCTRL = SMC_PMCTRL_RUNM(0b00);
        /* Wait for Transition*/
        while(SMC->PMSTAT != 0x01);
    }
}
```

As HSRUN mode allows the MCU to run at maximum clock speed, ensure to adjust frequencies in the SCG module to meet clock requirements for RUN mode. For details on these requirements, refer to the *Internal clocking requirements* section in the *MCX E24x Series Reference Manual* (document MCXE24XRM).

5.3 Very Low-Power Run (VLPR) mode entry

In VLPR mode, the on-chip voltage regulator is put into a Stop mode regulation state. In this state, the regulator is designed to supply enough current to the MCU over a reduced frequency. To further reduce power in this mode, disable the clocks to unused modules using their corresponding clock gating control bits in the PCC's registers.

Before entering this mode, the following conditions must be met:

- All clock monitors in the SCG must be disabled.
- Adjust the clock frequencies according to their maximum allowed values: core, system, and bus clock must be 4 MHz (maximum) and the flash clock must be set to 1 MHz (maximum). Also, all asynchronous clock sources are restricted to 4 MHz. System PLL, system oscillator, and FIRC must be disabled by software prior mode entry.
- Mode protection must be set to allow VLP modes, that is, SMC_PMPROT[AVLP] to 1.

- SMC_PMCTRL[RUNM] must be set to 0b10 to enter VLPR.

The below code snippet shows a basic RUN to VLPR transition function.

```
void RUN_to_VLPR (void)
{
    /* Disable clock monitors on SCG module */
    disable_clock_monitors();
    /* Adjust SCG settings to meet maximum frequencies value
    Disable SPLL, System Oscillator and FIRC */
    scg_configure_freq_for_VLPR();
    /* Allow very low power run mode */
    SMC->PMPROT |= SMC_PMPROT_AVLP_MASK;
    /* Check if current mode is RUN mode */
    if(SMC->PMSTAT == 0x01)
    {
        /* Reduce MCU power consumption in Very Low Power modes*/
        PMC->REGSC |= PMC_REGSC_BIASEN_MASK;
        /* Move to VLPR Mode*/
        SMC->PMCTRL = SMC_PMCTRL_RUNM(0b10);
        /* Wait for Transition*/
        while(SMC->PMSTAT != 0x04);
    }
}
```

Note: *Flash programming/erasing is not allowed. No FTFC commands of any type, including CSE commands (for CSEc parts), are available when the chip is in this mode.*

Note: *Do not increase the clock frequency while in VLPR mode, because the regulator is slow in responding and cannot manage fast load transitions. In addition, do not modify the clock source in the SCG module or any clock divider registers. Module clock enabled in the PCC can be set, but not cleared.*

Note: *To reduce MCU power consumption in low-power modes, PMC_REGSC[BIASEN] bit should be set. This bit enables source and well biasing for the core logic in low power modes. This bit must be set to 1 when using Very Low Power (VLP) modes.*

5.4 VLPR mode exit

To re-enter Normal RUN mode, clear SMC_PMCTRL[RUNM] bits. If a higher execution frequency is desired, poll PMSTAT until it is set to RUN, and then configure the SCG module as desired. Also, a reset event causes the MCU to come back to RUN mode.

The below code snippet shows a basic VLPR to RUN transition function.

```
void VLPR_to_RUN (void)
{
    /* Check if current mode is VLPR mode */
    if(SMC->PMSTAT == 0x04)
    {
        /* Move to RUN Mode*/
        SMC->PMCTRL = SMC_PMCTRL_RUNM(0b00);
        /* Wait for Transition*/
        while(SMC->PMSTAT != 0x01);
    }
}
```

5.5 Stop and VLPS mode entry sequence

When entering Stop/VLPS mode, clocks are shut off in an orderly sequence to safely place the chip in the targeted low-power state. The core executing a WFI instruction initiates all low-power entry sequences. When entering low-power modes, the chip performs the sequence described below:

1. The CPU clock is gated off immediately.
2. Requests are made to all non-CPU bus controllers (DMA and ENET if available) to enter Stop mode.
3. After all controllers have acknowledged they are ready to enter Stop mode, requests are made to all bus targets to enter Stop mode.
4. After all targets have acknowledged they are ready to enter Stop mode, the system and bus clocks are gated off depending on the target mode–VLPS/STOP1/STOP2. In STOP2 mode, bus clocks are not gated.
5. Additionally, for VLPS mode, clock generators are disabled in the SCG unless configured to be enabled.
6. The on-chip regulator in the PMC and internal power switches are configured to meet the power consumption goals for the targeted low-power mode. This step is valid only for VLPS and not for Stop mode. In Stop mode, the PMC is in run regulation.

Note: *If SIRC is not enabled in VLPS modes, setting PMC_REGSC[CLKBIASDIS] bit can reduce the power consumption. While using this bit, it must be ensured that the respective clock modules are disabled in VLPS mode, else, severe malfunction of clock modules can happen.*

5.5.1 Stop1/2 mode entry

STOP1/2 mode is entered via the sleep-now or sleep-on-exit with the SLEEPDEEP bit set in the system control register in Arm core.

The SCG module can be configured to leave the reference clocks running. For Stop modes, all SCG clocks are available (LPO, PLL, SIRC, FIRC, and OSC). For more details, refer to Section 6.

SMC_STOPCTRL[PSTOPO] bits selects whether the MCU is sent to STOP1 (0b01) or STOP2 (0b10) mode.

The below code snippet shows a basic RUN to STOP1/STOP2 transition function.

```
void RUN_to_STOP (void)
{
    /* Enable SLEEPDEEP bit in the Core
    * (Allow deep sleep modes) */
    SCB ->SCR |= SCB_SCR_SLEEPDEEP_Msk;
    /* Select Stop Mode */
    SMC->PMCTRL = SMC_PMCTRL_STOPM(0b00);
    /* Select which STOP mode (Stop1 or Stop2)
    * is desired (Stop1 - 0b01, Stop2 - 0b10) */
    SMC->STOPCTRL = SMC_STOPCTRL_PSTOPO(0b01);
    /* Check if current mode is RUN mode */
    if(SMC->PMSTAT == 0x01)
    {
        /* Go to deep sleep mode */
        asm("WFI");
    }
}
```

5.5.2 VLPS mode entry

There are two ways in which VLPS mode can be entered:

- Entry into Stop mode occurs when the SLEEPDEEP bit is set in the SCR, and the MCU executes either sleep-now or sleep-on-exit instructions. This transition happens while the MCU is in VLPR mode, and the PMCTRL[STOPM] field is set to either 0b010 or 0b000.
- Entry into Stop mode occurs when the SLEEPDEEP bit is set in the SCR of the Arm core, and the MCU executes either the sleep-now or sleep-on-exit instruction. This transition happens when the MCU is in Normal RUN mode, and the PMCTRL[STOPM] field is set to 0b010. When VLPS is entered directly from RUN mode, the hardware disables exit to VLPR, and the system always exit back to RUN.

Transition to VLPS is almost the same as for Stop1/2 modes except allowing very low-power modes in SMC_PMPROT register. Ensure to disable system PLL, system oscillator, and FIRC before VLPS mode entry. The below code snippet shows a basic RUN to VLPS transition function.

```
void RUN_to_VLPS (void)
{
    /* Adjust SCG settings to meet maximum frequencies value */
    scg_disable_pll_and_firc();
    /* Enable SLEEPDEEP bit in the Core
    * (Allow deep sleep modes) */
    SCB ->SCR |= SCB_SCR_SLEEPDEEP_Msk;
    /* Allow very low power run mode */
    SMC->PMPROT |= SMC_PMPROT_AVLP_MASK;
    /* Select VLPS Mode */
    SMC->PMCTRL = SMC_PMCTRL_STOPM(0b10);
    PMC->REGSC |= PMC_REGSC_BIASEN_MASK;
    /* Check if current mode is RUN mode */
    if(SMC->PMSTAT == 0x01)
    {
        /* Go to deep sleep mode */
        asm("WFI");
    }
}
```

Note: *To reduce MCU power consumption in low-power modes, PMC_REGSC[BIASEN] bit should be set. This bit enables source and well biasing for the core logic in low-power modes. This bit must be set to 1 when using very low-power modes.*

5.6 STOP and VLPS mode exit sequence

Either a reset or an interrupt event initiates exit from a Low-Power Stop mode. The following sequence then executes to restore the system to a Run mode (RUN or VLPR):

1. The on-chip regulator in the PMC, clock generators, and internal power switches are restored. This step is valid only for VLPS and not for Stop mode because in this mode, the PMC is in run regulation.
2. System and bus clocks are enabled to all controllers and targets.
3. The CPU clock is enabled and the CPU begins servicing the reset or interrupt that initiated the exit from the Low-Power Stop mode.

6 Modules in power modules

For details on all peripherals and their availability in different low-power modes, refer to the *Module operation in available low-power modes* section in *MCX E24x Series Reference Manual* (document MCXE24XRM).

7 Hardware and software considerations

This section describes recommendations that can be followed to reduce MCU power consumption.

7.1 Hardware considerations

All unused pins in an application (especially analog functionality) should follow some recommendations to eliminate possible current consumption increase:

- The DAC output pin should be floating.
- The ADC pins should be grounded.

7.2 Software considerations

To conserve power, most module clocks can be turned off by configuring the CGC field of the peripheral control register in the PCC module. These fields are cleared after any reset, which disables the peripheral clock of the corresponding module. Ensure to disable the module before turning off the clock.

Several peripherals support Peripheral Doze mode. In this mode, a register field can be used to disable the peripheral during a low-power mode.

7.3 Tips for making low-power measurements on the bench

This section provides tips for making low-power measurements on the bench.

7.3.1 External

The suggestions below address the most common issues encountered when trying to duplicate the data sheet current specs.

- **When using a digital multimeter (DMM), use Manual Range mode**: Using a DMM with an auto-ranging function enabled can cause LVD and POR resets. This is most common when the user is exiting from one of the low-power modes like LLS or VLPS back to Run. While the MCU is in low-power mode, the DMM switches to a micro-amp or nano-amp range to measure current accurately. However, a sudden inrush of current can force the DMM to change its range again, potentially affecting the measurement. The range change does not happen fast enough and the MCU starves and pulls the VDD level below the LVD or power on reset (POR) limits.
- **Disconnect the debugger and power cycle the MCU**: With the JTAG debugger attached, the MCU can have the debugger module in the MCU active, clocking, and consuming power. The external debugger hardware can also load the I/O of the JTAG port when attached. Therefore, the low-power measurements are higher than expected.
- **Isolate the MCU VDDs**: To measure the current drawn of the MCU, then remove the other IC and component networks that are sourced by the voltage supply sourcing the MCU. For example, some EVBs have a potentiometer connected between MCU_VDD and the ground. A 5 K potentiometer across a 3.6 V supply pulls 720 µA. This is huge when considering that the MCU consumes around dozens µA in lowest power modes.
- **Match the impedance of the inputs**: If the impedances of high-speed signals (fast edge transitions) are not well matched, the signals can "ring" and exceed the VDD supply of the device. This results in the signal providing current to the device through the input protection diodes. This is true for high-speed input clocks. This issue can result in negative IDD measurements while in the lowest power modes.
- **Match the voltage levels**: Some MCU input pins are 5 V tolerant. However, when the MCU enters low-power modes, any input voltage higher than MCU_VDD can affect the current measurement through MCU_VDD. The higher input pin back powers the MCU through the input pin, resulting in negative IDD reading in low-power modes.
- **Reduce pin loading of the MCU**: When the MCU sources current through the output pins, the power is being sourced through MCU_VDD. This is most evident when the user outputs high frequency signals to an output pin as they might with the external memory interfaces, such as clock and address/data pins.

7.3.2 Internal

Below is a list of the most common issues that can prevent the user from getting to the lowest data sheet current specs.

- **Watchdog is not disabled causing resets**: Disable or service the watchdog.
- **The clock monitor is not disabled, which may cause resets**: Disable all clock monitors.
- **A crystal oscillator is enabled in Low-Power mode**: The RTC oscillator typically consumes <500 nA of current.
- **The CLKOUT signal is being output to a pin**: Any pin that is constantly changing state draws power.
- **The requested low-power mode is not allowed with a corresponding bit in the PMPROT register**: For example, if AVLP is not set in the power mode protection (PMPROT) and the WFI instruction is executed, the user does not enter Stop mode.
- **The clock gate for a module is not enabled before it is read or written**: This causes a reset before the MCU tries to enter the Low-Power mode.
- **The clock gate for a module that must acknowledge the mode controller mode entry request is turned off prior to Low-Power mode entry**: This results in a Stop mode acknowledge reset.
- **Failure to un-comment out the call to the stop or sleep function after debugging is complete**: This keeps the user in the higher run current mode.
- **The frequency of wake-up events is too high**: It means that the MCU spends more time in Run or VLPR mode than in a Low-Power mode. The transition time from Low-Power mode to Run mode is quick. If the MCU only spends 9 ms in run and 1 ms in a Low-Power mode, the average current of the system is considerably higher than if the MCU was running only 1 ms every 1 second.
- **The MCU is running at a higher frequency than is needed to accomplish the work**: Throttle the clock with the SCG dividers or reduce the clock. Obviously, the higher the MCU frequency the higher the IDD of the MCU. Reduce the clock and lower the run or wait current. However, there is some trade-off. If the current of Run mode can be tolerated, then getting work done as quickly as possible and going right back to Low-Power mode is more advantageous than running a slower clock in Run mode.
- **An input pin is floating without an internal or external pull device**: This can result in 50-80 µA of current per pin. This includes the JTAG or SWD pins. Disable the JTAG pins on PORTA or properly terminate the inputs.

7.3.3 FRDM-MCXE247

If the user want to use the FRDM-MCXE247 to reproduce data sheet low-power current data, follow the below suggestions:

- To use onboard MCU-link (for UART or debug), short C93 (Ethernet PYH reset pin) and remove R72 (Ethernet PYH CLK in).
- To measure low power current with an external debugger, remove R2.

7.4 Current consumption measurements

Current consumption depends on several factors, such as which modules are enabled, the frequency driving the core and peripherals, and environmental conditions like temperature.

To account for these variables, the MCU data sheet includes test cases for different power modes. For accurate current consumption measurements, refer to the *MCX E24x Data Sheet* (document MCXE24X).

8 Abbreviations and acronyms

Table 3 lists the abbreviation and acronyms used in this document.

Table 3. Abbreviations and acronyms

Abbreviation	Description
NVIC	Nested vectored interrupt controller
AWIC	Asynchronous wake-up interrupt controller
SCR	System control register
HSRUN	High Speed Run
VLPR	Very Low-Power Run
VLPS	Very Low-Power Stop
LVD	Low voltage detect
WFI	Wait for interrupt
FIRC	Fast internal reference clock
SIRC	Slow internal reference clock
SOSC	System oscillator
PLL	Phase-locked loop
SPLL	System PLL
PMC	Power management controller
LPO	Low power oscillator
DMM	Digital multimeter

9 Note about the source code in the document

Example code shown in this document has the following copyright and BSD-3-Clause license:

Copyright 2025 NXP Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials must be provided with the distribution.
3. Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

10 Revision history

Table 4 summarizes revisions to this document.

Table 4. Revision history

Document ID	Release date	Description
AN14736 v.1.0	22 July 2025	Initial public release

Legal information

Definitions

Draft — A draft status on a document indicates that 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 in a draft version of a document and shall have no liability for the consequences of use of such information.

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. NXP Semiconductors takes no responsibility for the content in this document if provided by an information source outside of NXP Semiconductors.

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 and its suppliers accept 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.

Terms and conditions of commercial sale — NXP Semiconductors products are sold subject to the general terms and conditions of commercial sale, as published at https://www.nxp.com/profile/terms, unless otherwise agreed in a valid written individual agreement. In case an individual agreement is concluded only the terms and conditions of the respective agreement shall apply. NXP Semiconductors hereby expressly objects to applying the customer's general terms and conditions with regard to the purchase of NXP Semiconductors products by customer.

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.

Suitability for use in non-automotive qualified products — Unless this document expressly states that this specific NXP Semiconductors product is automotive qualified, the product is not suitable for automotive use. It is neither qualified nor tested in accordance with automotive testing or application requirements. NXP Semiconductors accepts no liability for inclusion and/or use of non-automotive qualified products in automotive equipment or applications.

In the event that customer uses the product for design-in and use in automotive applications to automotive specifications and standards, customer (a) shall use the product without NXP Semiconductors' warranty of the product for such automotive applications, use and specifications, and (b) whenever customer uses the product for automotive applications beyond NXP Semiconductors' specifications such use shall be solely at customer's own risk, and (c) customer fully indemnifies NXP Semiconductors for any liability, damages or failed product claims resulting from customer design and use of the product for automotive applications beyond NXP Semiconductors' standard warranty and NXP Semiconductors' product specifications.

HTML publications — An HTML version, if available, of this document is provided as a courtesy. Definitive information is contained in the applicable document in PDF format. If there is a discrepancy between the HTML document and the PDF document, the PDF document has priority.

Translations — A non-English (translated) version of a document, including the legal information in that document, is for reference only. The English version shall prevail in case of any discrepancy between the translated and English versions.

Security — Customer understands that all NXP products may be subject to unidentified vulnerabilities or may support established security standards or specifications with known limitations. Customer is responsible for the design and operation of its applications and products throughout their lifecycles to reduce the effect of these vulnerabilities on customer's applications and products. Customer's responsibility also extends to other open and/or proprietary technologies supported by NXP products for use in customer's applications. NXP accepts no liability for any vulnerability. Customer should regularly check security updates from NXP and follow up appropriately.

Customer shall select products with security features that best meet rules, regulations, and standards of the intended application and make the ultimate design decisions regarding its products and is solely responsible for compliance with all legal, regulatory, and security related requirements concerning its products, regardless of any information or support that may be provided by NXP.

NXP has a Product Security Incident Response Team (PSIRT) (reachable at PSIRT@nxp.com) that manages the investigation, reporting, and solution release to security vulnerabilities of NXP products.

NXP B.V. — NXP B.V. is not an operating company and it does not distribute or sell products.

Trademarks

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

NXP — wordmark and logo are trademarks of NXP B.V.

Contents