Renesas Synergy™ Platform

Low-Power Modes and Mode Transitions on the DK-S124

Introduction

This document describes different Low Power Modes (LPMs) and mode transitions in the Renesas S124 Synergy™ MCU Group with the Synergy Software Package (SSP) and DK-S124 board kit. The attached application code allows you to set the S124 Synergy MCU into different power control modes, such as High-speed mode, Middle-speed mode, Low-speed mode; in addition to low power modes, such as Sleep mode and Software Standby mode.

Target Device

DK-S124 Kit v3.0

Required Resources

To build and run the application, you will need:

- A Renesas DK-S124 Synergy MCU v3.0 board
- A PC running Microsoft® Windows® 7 with the following software installed:
  - e² studio ISDE v7.3.0 or later
  - Synergy Software Package (SSP) v1.6.0 or later
  - IAR Embedded Workbench® for Synergy 8.23.3 or later
  - SSC v7.3.0 or later

Minimum PC Requirements

- Microsoft® Windows® 7
- Intel® Core™ family processor running at 2.0 GHz or higher (or equivalent processor)
- 8-GB memory
- 250-GB hard disk or SSD
- USB 2.0
- Internet connection

Contents

1. Overview

Reducing MCU power consumption has become a critical challenge for IoT applications, especially wearable devices. To provide different power controlling capabilities, the Arm® Cortex®-M0+ based S124 Synergy MCU Group has five operation power control modes with different operating frequencies and voltages, such as High-speed, Middle-speed, and Low-speed modes. In addition, it has three low power modes: Sleep, Software Standby, and Snooze. This document covers the basic features of those modes, as well as the transitions between modes triggered by a switch button and a potentiometer on the DK-S124 v3.0 board.

2. Power Modes of Synergy S124 MCU

The S124 MCU provides two sets of power modes; one set for power controlling and the other set for low power. These modes support different power or performance requirements. This section presents basic concepts and usage, especially with the SSP low power and the low-level HAL framework API. For details on the configuration of control registers, see the *S124 User's Manual*, and for APIs, see the *SSP User's Manual*.

2.1 Power control modes

When executing a program in the normal mode, the MCU power consumption is mainly affected by the clock and the peripheral module configurations. The S124 MCU allows you to adjust the System Clock (ICLK), the Peripheral Module Clocks (PCLKB, PCLKD), and the External Bus Clock (BCLK). It also allows you to stop peripheral modules by setting different control registers.

The S124 MCU has five predefined power control modes in the following table with different clock generation sources and frequency ranges.

Table 1. Available oscillators in each mode

	Oscillator						
Mode	**High-Speed On-Chip Osc**	**Middle-speed On-Chip Osc**	**Low-speed On-Chip Osc**	**Main Clock Osc**	**Sub-Clock Osc**	**IWDT-Dedicated On-Chip Osc**	**Power Consumption**
High-speed	Available	Available	Available	Available	Available	Available	High
Middle-speed	Available	Available	Available	Available	Available	Available	
Low-voltage	Available	Available	Available	Available	Available	Available	
Low-speed	Available	Available	Available	Available	Available	Available	
Subosc-speed	N/A	N/A	Available	N/A	Available	Available	Low

2.1.1 High-speed mode

The maximum operating frequency during flash read is 32 MHz for ICLK. The operating voltage range is 2.4 V to 5.5 V during flash read. However, for ICLK, the maximum operating frequency during flash read is 16 MHz when the operating voltage is 2.4 V or larger, and smaller than 2.7 V. During flash programming/erasure, the operating frequency range is 1 to 32 MHz and the operating voltage range is 2.7 V to 5.5 V. Figure 1 to Figure 5 summarizes various recommendations for the different modes available.

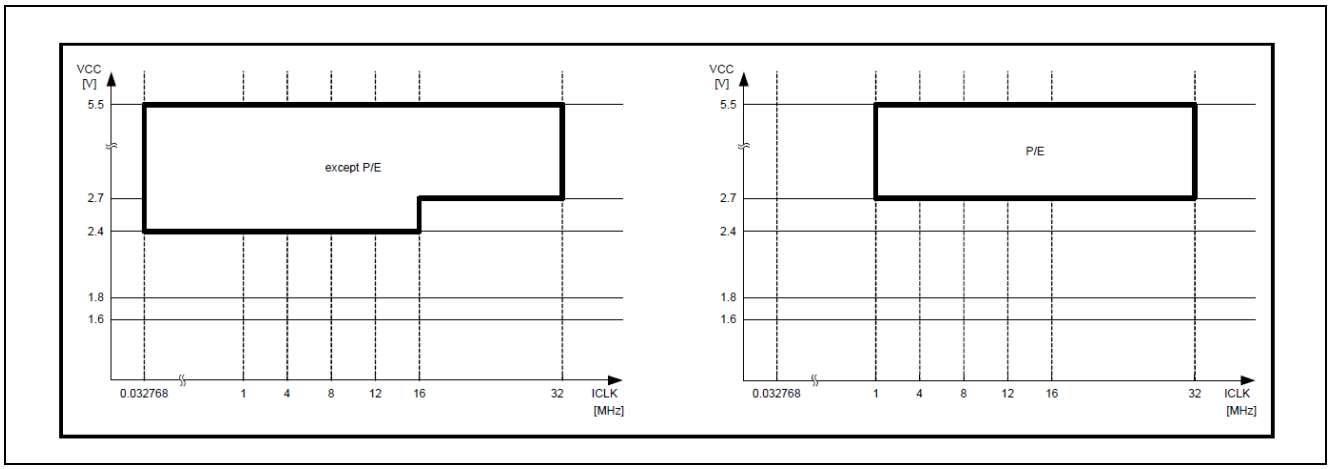


Figure 1. System clock range for High-speed mode

2.1.2 Middle-speed mode

The maximum operating frequency during flash read is 12 MHz for ICLK. The operating voltage range is 1.8 V to 5.5 V during flash read. However, for ICLK, the maximum operating frequency during flash read is 8 MHz when the operating voltage is 1.8 V or larger, and smaller than 2.4 V. The following figure shows the Middle-speed mode voltage and recommended frequencies.

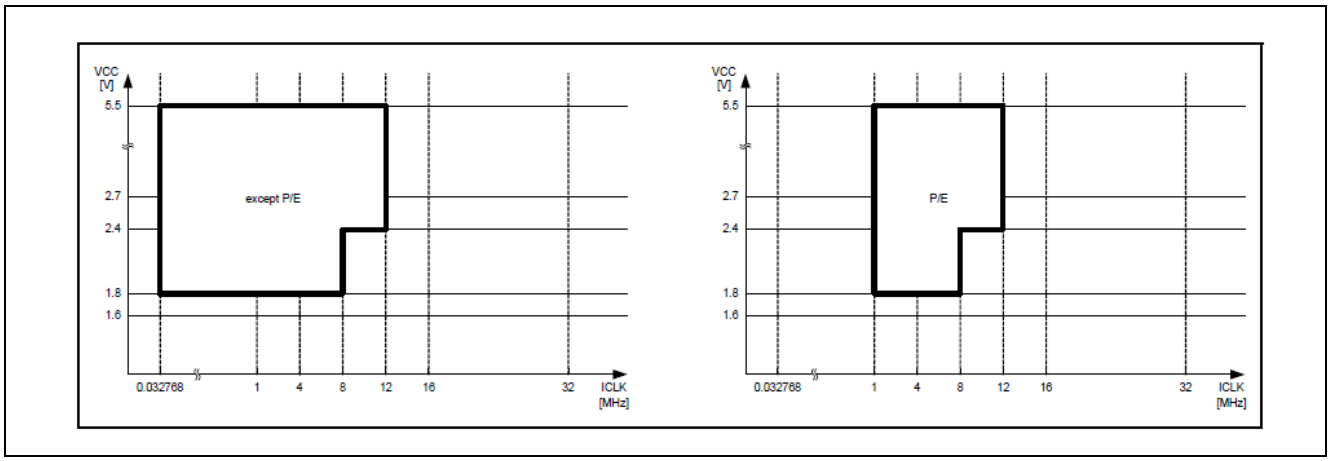


Figure 2. System clock range for Middle-speed mode

2.1.3 Low-voltage mode

After a reset is canceled, operation starts in low-voltage mode. The maximum operating frequency during flash read is 4 MHz for ICLK. The operating voltage range is 1.6 V to 5.5 V during flash read. During flash programming/erasure, the operating frequency range is 1 MHz to 4 MHz and the operating voltage range is 1.8 V to 5.5 V.

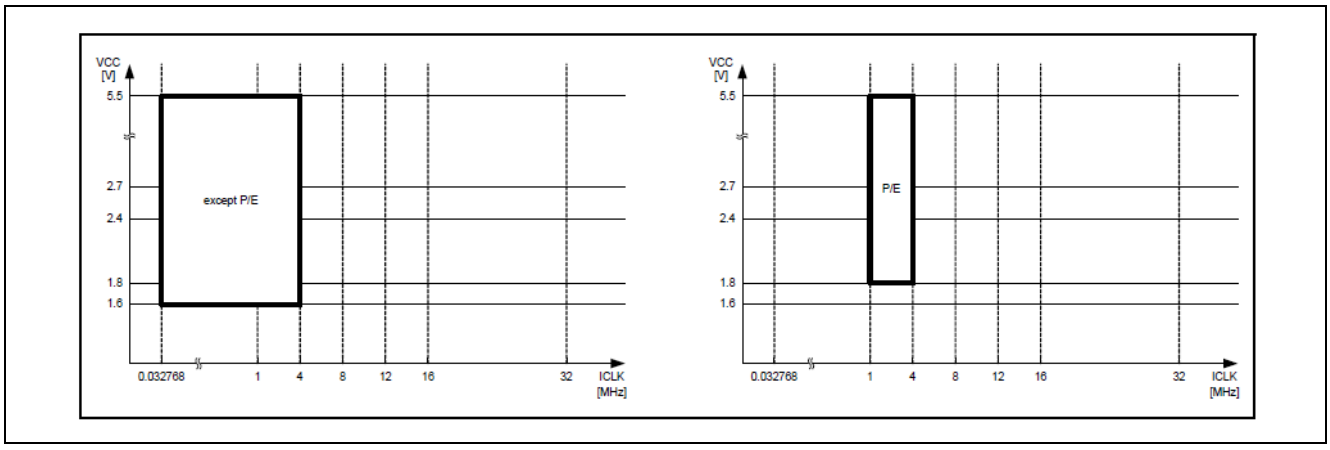


Figure 3. System clock range for low-voltage mode

2.1.4 Low-speed mode

The maximum operating frequency during flash read is 1 MHz for ICLK. The operating voltage range is 1.8 V to 5.5 V during flash read. P/E operations for flash memory are prohibited.

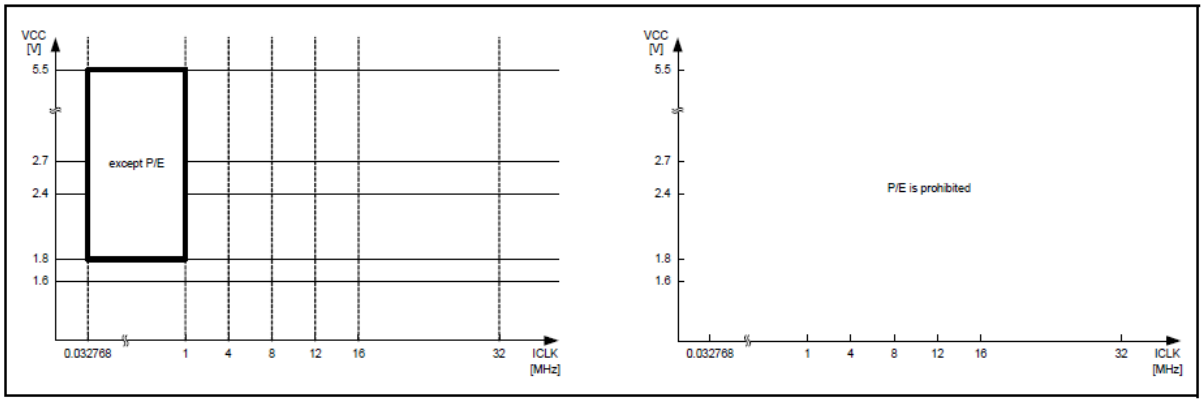


Figure 4. System clock range for the Low-speed mode

2.1.5 Subosc-speed mode

The maximum operating frequency during flash read is 37.6832 kHz for ICLK. The operating voltage range is 1.8 V to 5.5 V during flash read. P/E operations for flash memory are prohibited.

Using the oscillators other than the sub-clock oscillator or Low-speed on-chip oscillator is prohibited.

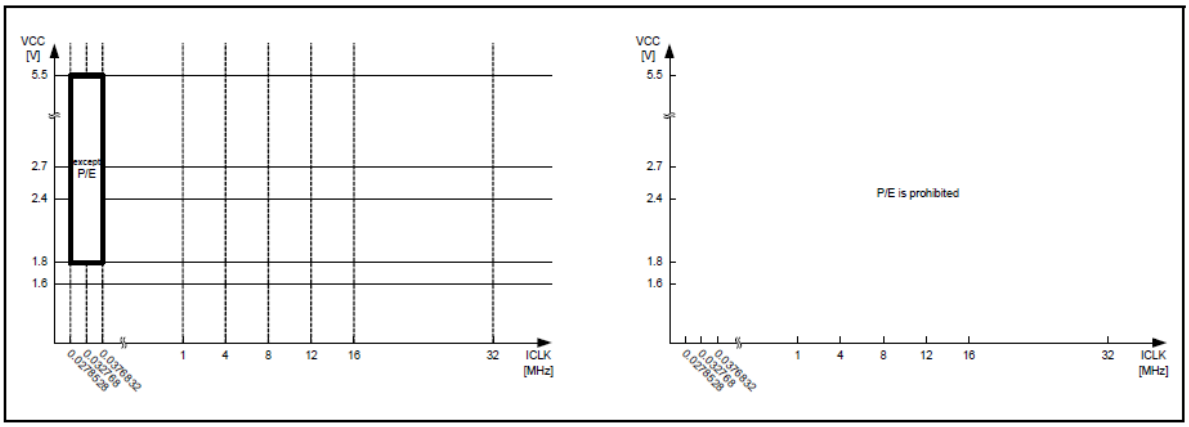


Figure 5. System clock range for the Subosc-speed mode

2.1.6 Transition between power control modes

To switch between power modes, you must follow the recommended procedure for setting registers. For example, switching from the Higher-speed mode to the Low-speed mode should be performed in the following steps.

Operation in High-speed mode

1. Change the oscillator to that used in Low-speed mode. Set the frequency of each clock lower than the maximum operating frequency in Low-speed mode.
2. Turn off the oscillators that are not required in Low-speed mode.
3. Confirm that the OPCCR.OPCMTSF flag is 0 (indicates transition completed).
4. Set the OPCCR.OPCM bit to 11b (Low-speed mode).
5. Confirm that the OPCCR.OPCMTSF flag is 0 (indicates transition completed).

Operation in Low-speed mode

2.2 Low power modes

To further reduce the power consumption, the MCU can be **stopped**, by entering one of the low-power modes (LPM). The S124 MCU device has three LPMs: Sleep, Software Standby and Snooze. The following table lists their clock sources and available peripheral modes. See *S124 User's Manual* for details.

Table 2. Operating conditions for each low-power mode

Item	Sleep mode	Software Standby mode	Snooze mode[1]
Transition condition	WFI instruction while SBYCR.SSBY = 0	WFI instruction while SBYCR.SSBY = 1	Snooze request in Software Standby mode. SNZCR.SNZE = 1
Canceling method	All interrupts. Any reset available in the mode.	Interrupts shown in Table 10.3 in the *S124 User's Manual*. Any reset available in the mode.	Interrupts shown in Table 10.3 in the *S124 User's Manual*. Any reset available in the mode.
State after cancellation by an interrupt	Program execution state (interrupt processing)	Program execution state (interrupt processing)	Program execution state (interrupt processing)
State after cancellation by a reset	Reset state	Reset state	Reset state
Main clock oscillator	Selectable	Stop	Selectable[2]
Sub-clock oscillator	Selectable	Selectable	Selectable
High-speed on-chip oscillator	Selectable	Stop	Selectable
Middle-speed on-chip oscillator	Selectable	Stop	Selectable
Low-speed on-chip oscillator	Selectable	Selectable	Selectable
IWDT-dedicated on-chip oscillator	Selectable[4]	Selectable[4,]	Selectable[4]
Oscillation stop detection function	Selectable	Operation prohibited	Operation prohibited
Clock/buzzer output function	Selectable	Selectable[3]	Selectable
CPU	Stop (Retained)	Stop (Retained)	Stop (Retained)
SRAM	Selectable	Stop (Retained)	Selectable
Flash memory	Operating	Stop (Retained)	Stop (Retained)
Data Transfer Controller (DTC)	Selectable	Stop (Retained)	Selectable
USB 2.0 Full-Speed Module (USBFS)	Selectable	Stop (Retained)[5]	Operation prohibited[5]
Watchdog Timer (WDT)	Selectable	Stop (Retained)	Stop (Retained)
Independent Watchdog Timer (IWDT)	Selectable[4]	Selectable[4]	Selectable[4]
Realtime clock (RTC)	Selectable	Selectable	Selectable
Asynchronous General Purpose Timer (AGTn, n = 0, 1)	Selectable	Selectable[6]	Selectable[6]
14-Bit A/D Converter (ADC14)	Selectable	Stop (Retained)	Selectable[10]
12-Bit D/A Converter (DAC12)	Selectable	Stop (Retained)	Selectable
Capacitive Touch Sensing Unit (CTSU)	Selectable	Stop (Retained)	Selectable
Data Operation Circuit (DOC)	Selectable	Stop (Retained)	Selectable
Serial Communications Interface (SCI0)	Selectable	Stop (Retained)	Selectable9
Serial Communications Interface (SCIn, n = 1, 9)	Selectable	Stop (Retained)	Operation prohibited
I^2C Bus Interface (IIC0)	Selectable	Selectable	Operation prohibited

Item	Sleep mode	Software Standby mode	Snooze mode[1]
I²C Bus Interface (IIC1)	Selectable	Stop (Retained)	Operation prohibited
Event Link Controller (ELC)	Selectable	Stop (Retained)	Selectable[7]
Low Power Analog Comparator (ACMPLP0)	Selectable	Selectable[8]	Selectable[8]
Low Power Analog Comparator (ACMPLP1)	Selectable	Selectable[8]	Selectable[8]
NMI, IRQn (n = 0 to 7) pin interrupt	Selectable	Selectable	Selectable
Key Interrupt Function (KINT)	Selectable	Selectable	Selectable
Low voltage detection (LVD)	Selectable	Selectable	Selectable
Power-on reset circuit	Operating	Operating	Operating
Other peripheral modules	Selectable	Stop (Retained)	Operation prohibited
I/O Ports	Operating	Retained	Operating

Note: Selectable means that operating or not operating can be selected by control registers. Stop (Retained) means that the contents of the internal registers are retained but the operations are suspended. Operation prohibited means that the function must be stopped before entering Software Standby mode. Otherwise, proper operation is not guaranteed in Snooze mode.

Notes:

1. All modules whose module-stop bits are 0 start as soon as PCLKs are supplied after entering Snooze mode. To avoid an increasing power consumption in Snooze mode, set the module-stop bit of modules that are not required in Snooze mode to 1 before entering Software Standby mode.
2. When using SCI0 in Snooze mode, the MOSCCR.MOSTP bit must be 1.
3. Stopped when the clock output source selects bits (CKOCR.CKOSEL[2:0]) are set to a value other than 010b (LOCO) and 100b (SOSC).
4. Operating or stopping is selected by setting the IWDT stop control bit (IWDTSTPCTL) in option function select register 0 (OFS0) in IWDT auto-start mode.
5. Detection of USBFS resumption is possible.
6. AGT0 operation is possible when 100b (LOCO) or 110b (SOSC) is selected by the AGT0.AGTMR1.TCK[2:0] bits. AGT1 operation is possible when 100b (LOCO), 110b (SOSC), or 101 (Underflow event signal from AGT0) is selected by the AGT1.AGTMR1.TCK[2:0] bits.
7. Event lists the restrictions described in section 10.9.13, ELC Event in Snooze Mode.
8. Only VCOUT function is permitted. The VCOUT pin operates when ACMPLP uses no digital filter. For details on digital filter, see section 33, Low Power Analog Comparator (ACMPLP).
9. Serial communication modes of SCI0 is only in asynchronous mode.
10. When using the 14-Bit A/D Converter in Snooze mode, the ADCMPCR.CMPAE or ADCMPCR.CMPBE bit must be 1.

2.2.1 Sleep mode

The CPU stops operating but the contents of its internal registers are retained. Other peripheral functions do not stop. The CPU can be woken by any interrupt, RES pin reset, a power-on reset, a voltage monitor reset, a SRAM parity error reset, or a reset caused by an IWDT or a WDT underflow.

2.2.2 Software standby mode (SBY)

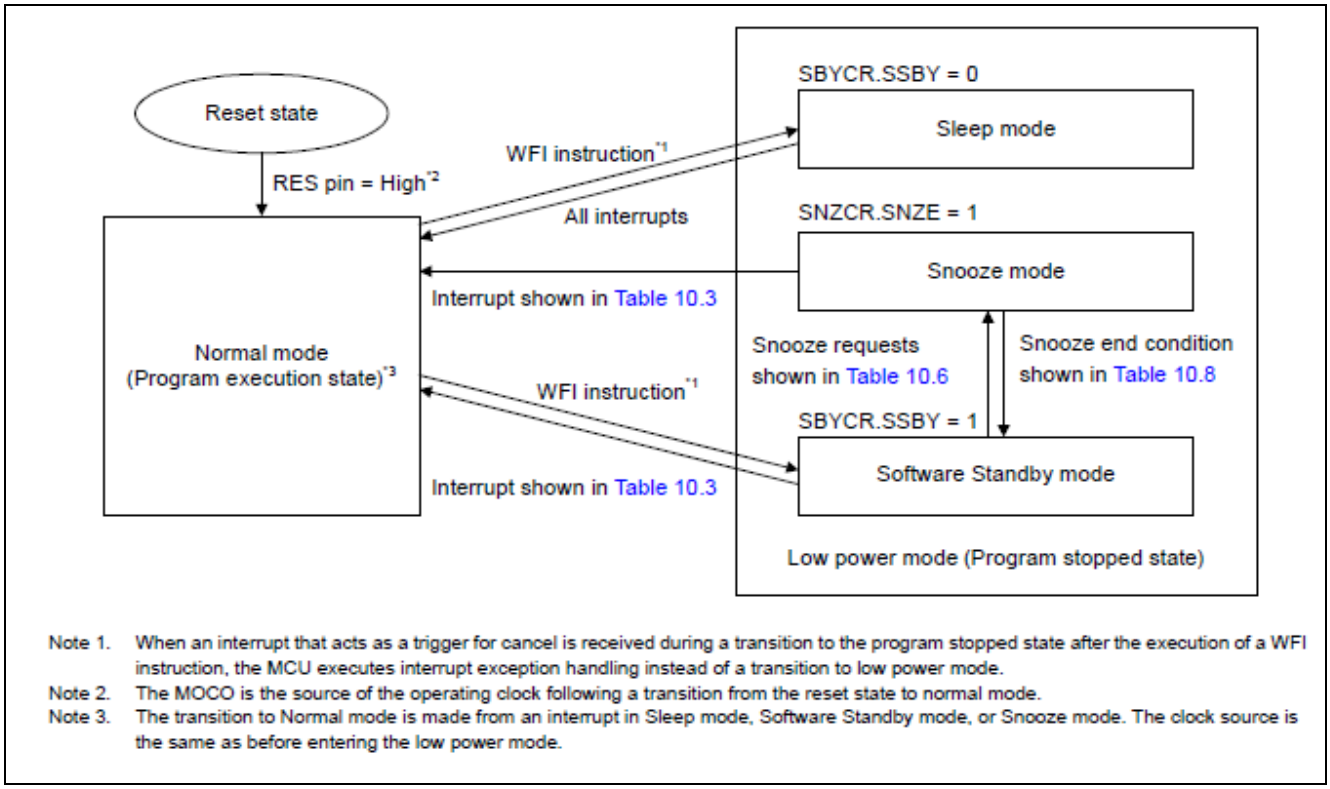
The CPU and most of the on-chip peripheral functions and oscillators stop. However, the contents of the CPU internal registers and SRAM data, the states of on-chip peripheral functions and the I/O ports are retained. Software Standby mode allows for a significant reduction in power consumption because most of the oscillators stop in this mode. Only those interrupts specified by Wake Up Interrupt Enable Register (WUPEN) can cancel the Software Standby mode.

2.2.3 Snooze mode (SNZ)

Snooze mode is similar to the Sleep mode, but some peripheral modules can operate without waking up the CPU. Snooze Mode can be entered through the Software Standby mode using some interrupt sources, called Snooze Requests, and woken up by those interrupts available in the Software Standby mode.

2.2.4 Transitions between the Low Power modes (LPMs)

The following figure lists available transitions between Normal mode and LPMs. The conditions or interrupt sources for triggering such a transition are specified in the *S124 User's Manual*.



Note 1. When an interrupt that acts as a trigger for cancel is received during a transition to the program stopped state after the execution of a WFI instruction, the MCU executes interrupt exception handling instead of a transition to low power mode.
Note 2. The MOCO is the source of the operating clock following a transition from the reset state to normal mode.
Note 3. The transition to Normal mode is made from an interrupt in Sleep mode, Software Standby mode, or Snooze mode. The clock source is the same as before entering the low power mode.

Figure 6. The transitions between the Normal mode and three Low Power modes

2.3 Power consumption for different power control modes and low power modes

See Section 41.2.9 in the *S124 User's Manual* for performance values, such as standby current, and so on. As a summary, sections 2.3.1 through 2.3.4 list standby current for some of the power control and low power mode combinations. The standby current information comes from Table 41.11 in the *S124 User's Manual*.

2.3.1 Standby current in the normal/sleep mode and high-speed mode

Operating and standby current (1) (1/2)

Conditions: VCC = AVCC0 = 1.6 to 5.5 V

Item					Symbol	Typ*9	Max	Unit	Test Conditions
Supply current*1	High-speed mode*2	Normal mode	All peripheral clock disabled, code executing from flash*5	ICLK = 32 MHz	I_{CC}	3.6	-	mA	*7
				ICLK = 16 MHz		2.4	-		
				ICLK = 8 MHz		1.7	-		
			All peripheral clock disabled, CoreMark code executing from flash*5	ICLK = 32 MHz		5.6	-		
				ICLK = 16 MHz		3.5	-		
				ICLK = 8 MHz		2.4	-		
			All peripheral clock enabled, code executing from flash*5	ICLK = 32 MHz		9.5	-		*8
				ICLK = 16 MHz		5.4	-		
				ICLK = 8 MHz		3.3	-		
			All peripheral clock enabled, code executing from flash*5	ICLK = 32 MHz		-	21.0		
		Sleep mode	All peripheral clock disabled*5	ICLK = 32 MHz		1.5	-		*7
				ICLK = 16 MHz		1.1	-		
				ICLK = 8 MHz		0.9	-		
			All peripheral clock enabled*5	ICLK = 32 MHz		7.2	-		*8
				ICLK = 16 MHz		4.0	-		
				ICLK = 8 MHz		2.4	-		
		Increase during BGO operation*6				2.5	-		-

2.3.2 Standby current in the normal/sleep mode and middle/low-speed modes

	Middle-speed mode*2	Normal mode	All peripheral clock disabled, code executing from flash*5	ICLK = 12 MHz	I_{CC}	1.7	-	mA	*7
				ICLK = 8 MHz		1.5	-		
			All peripheral clock disabled, CoreMark code executing from flash*5	ICLK = 12 MHz		2.7	-		
				ICLK = 8 MHz		1.9	-		
			All peripheral clock enabled, code executing from flash*5	ICLK = 12 MHz		3.9	-		*8
				ICLK = 8 MHz		3.0	-		
			All peripheral clock enabled, code executing from flash*5	ICLK = 12 MHz		-	8.0		
		Sleep mode	All peripheral clock disabled*5	ICLK = 12 MHz		0.8	-		*7
				ICLK = 8 MHz		0.8	-		
			All peripheral clock enabled*5	ICLK = 12 MHz		2.9	-		*8
				ICLK = 8 MHz		2.2	-		
		Increase during BGO operation*6				2.5	-		-
	Low-speed mode*3	Normal mode	All peripheral clock disabled, code executing from flash*5	ICLK = 1 MHz	I_{CC}	0.2	-	mA	*7
			All peripheral clock disabled, CoreMark code executing from flash*5	ICLK = 1 MHz		0.3	-		
			All peripheral clock enabled, code executing from flash*5	ICLK = 1 MHz		0.4	-		*8
			All peripheral clock enabled, code executing from flash*5	ICLK = 1 MHz		-	2.0		
		Sleep mode	All peripheral clock disabled*5	ICLK = 1 MHz		0.2	-		*7
			All peripheral clock enabled*5	ICLK = 1 MHz		0.3	-		*8

2.3.3 Standby current in the normal/sleep mode and low-voltage/subosc-speed modes

Supply current*1	Low-voltage mode*3	Normal mode	All peripheral clock disabled, code executing from flash*5	ICLK = 4 MHz	I_{CC}	1.4	-	mA	*7
			All peripheral clock disabled, CoreMark code executing from flash*5	ICLK = 4 MHz		1.4	-		
			All peripheral clock enabled, code executing from flash*5	ICLK = 4 MHz		2.1	-		*8
			All peripheral clock enabled, code executing from flash*5	ICLK = 4 MHz		-	4.0		
		Sleep mode	All peripheral clock disabled*5	ICLK = 4 MHz		0.9	-		*7
			All peripheral clock enabled*5	ICLK = 4 MHz		1.6	-		*8
	Subosc-speed mode*4	Normal mode	All peripheral clock disabled, code executing from flash*5	ICLK = 32.768 kHz	I_{CC}	5.9	-	µA	*7
			All peripheral clock enabled, code executing from flash*5	ICLK = 32.768 kHz		13.0	-		*8
			All peripheral clock enabled, code executing from flash*5	ICLK = 32.768 kHz		-	55.0		
		Sleep mode	All peripheral clock disabled*5	ICLK = 32.768 kHz		3.2	-		*7
			All peripheral clock enabled*5	ICLK = 32.768 kHz		10.0	-		*8

2.3.4 Standby current in the Normal/Software Standby mode

		Symbol	Typ*3	Max	Unit	Test conditions
Software Standby mode*2	T_a = 25°C	I_{CC}	0.4	1.5	µA	-
	T_a = 55°C		0.6	5.5		
	T_a = 85°C		1.2	10.0		
	T_a = 105°C		2.6	40.0		
Increment for RTC operation with low-speed on-chip oscillator*4			0.4	-		-
Increment for RTC operation with sub-clock oscillator*4			0.5	-		SOMCR.SODRV[1:0] are 11b (Low power mode 3)
			1.3	-		SOMCR.SODRV[1:0] are 00b (Normal mode)

Compared to the Sleep mode, the standby current of the Software Standby mode is even smaller.

3. Project: Making Low Power Mode Transitions with DK-S124

To illustrate transitions in power control and the low power modes, an application has been created with DK-S124 board v3.0. This section describes its hardware and software setup.

3.1 Hardware: Renesas DK-S124 Kit

This project is one of the applications developed on the DK-S124 v 3.0 Synergy MCU board (see Figure 7), which is a development kit for the Renesas Synergy™ S124 microcontroller in a LQFP64 package.

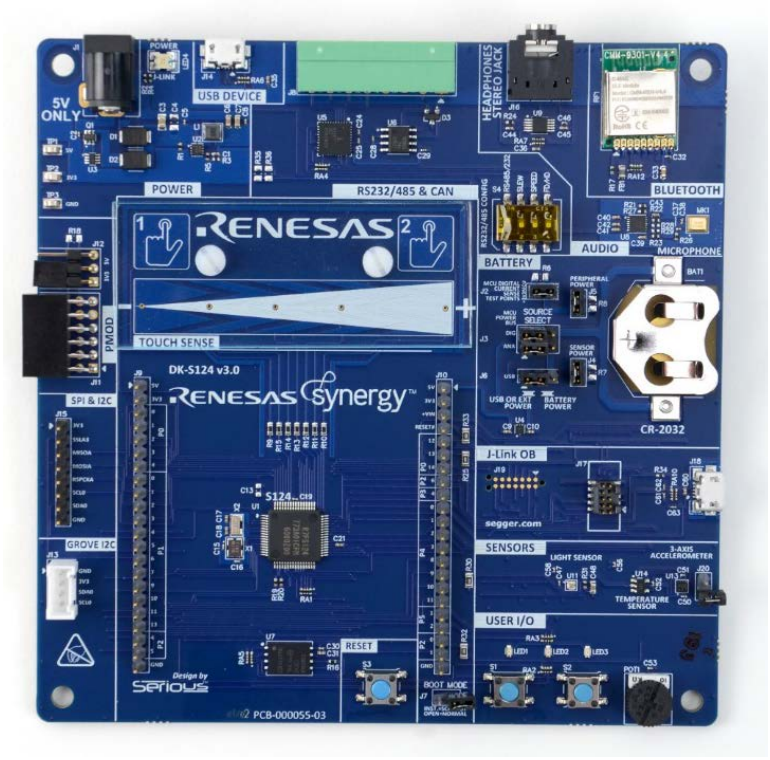


Figure 7. DK-S124 v3.0 Synergy MCU board

3.2 Project design: Algorithms

There are two sets of switchable power modes supported by S124 Synergy MCU. Different modes and their transitions can be abstracted into a finite state machine (FSM). Interaction between the two sets of power modes is considered a product of two FSMs. If you make power modes transitions at any time, the product such FSMs may have 15 of states, and more than 55 transitions to be implemented. To simplify implementation, only four power controlling modes are considered: High-speed mode, Middle-speed mode, Low-voltage mode, and Low-speed mode. The following figure shows the interaction of two FSMs implemented with a single thread. That application can be extended into a more complicated implementation with two threads; one for power controlling modes, and the other to switch Normal mode into different Low Power modes (LPMs).

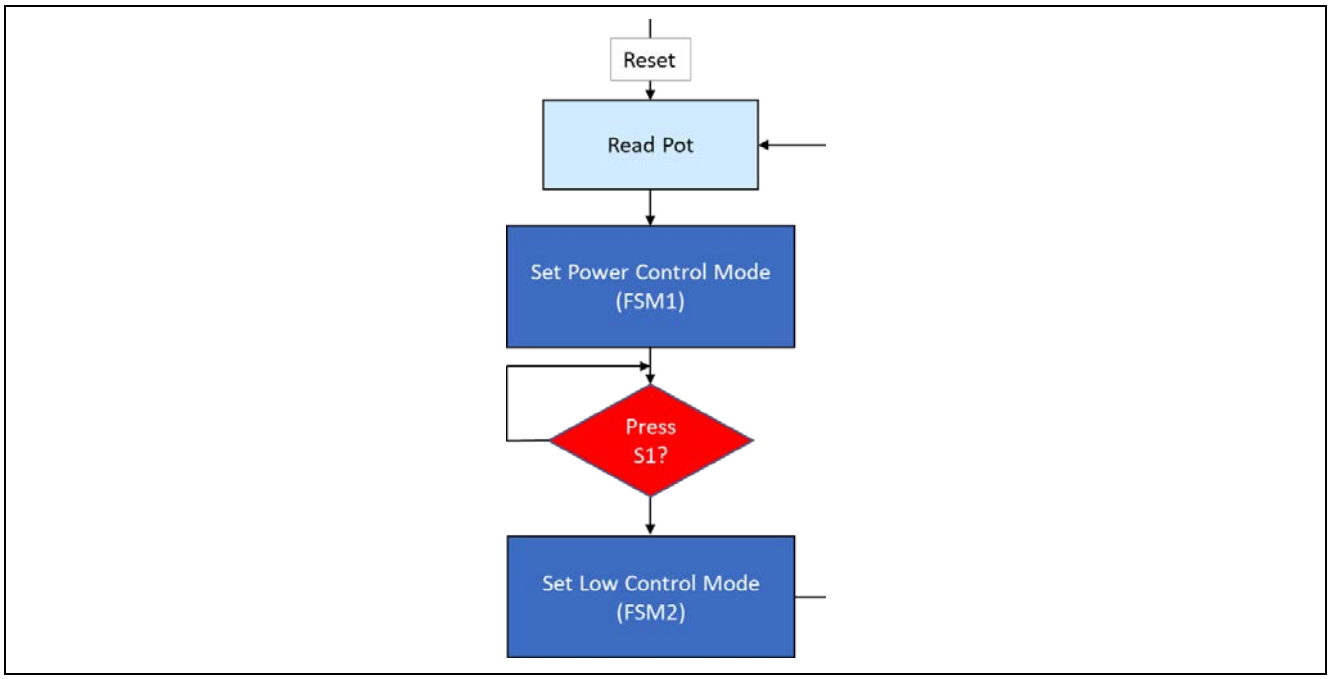


Figure 8. Single thread-based mode control algorithm

The following figure shows the FSM1 state transitions.

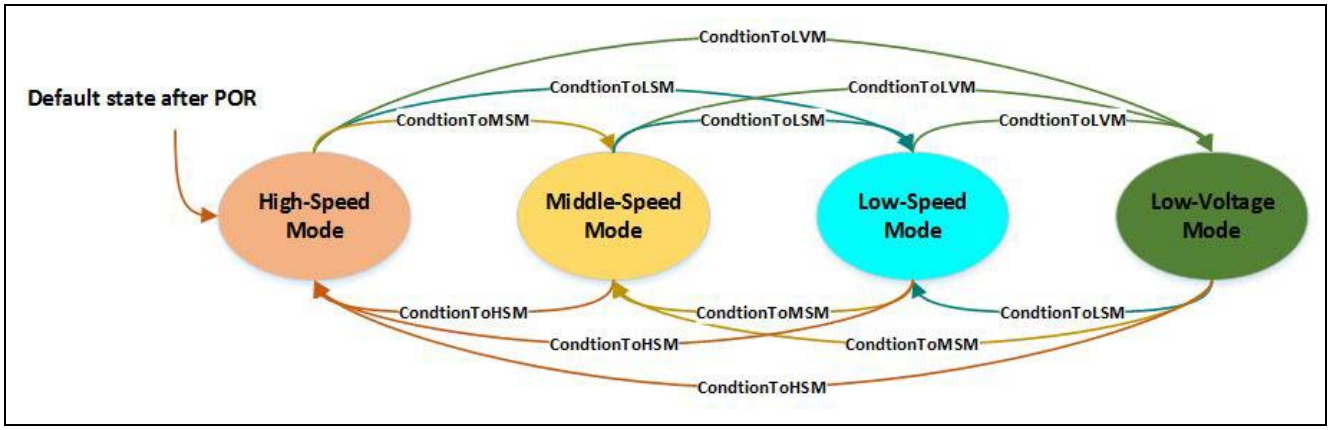


Figure 9. FSM1 state transition diagram

The following figure shows the FSM2 state transitions.

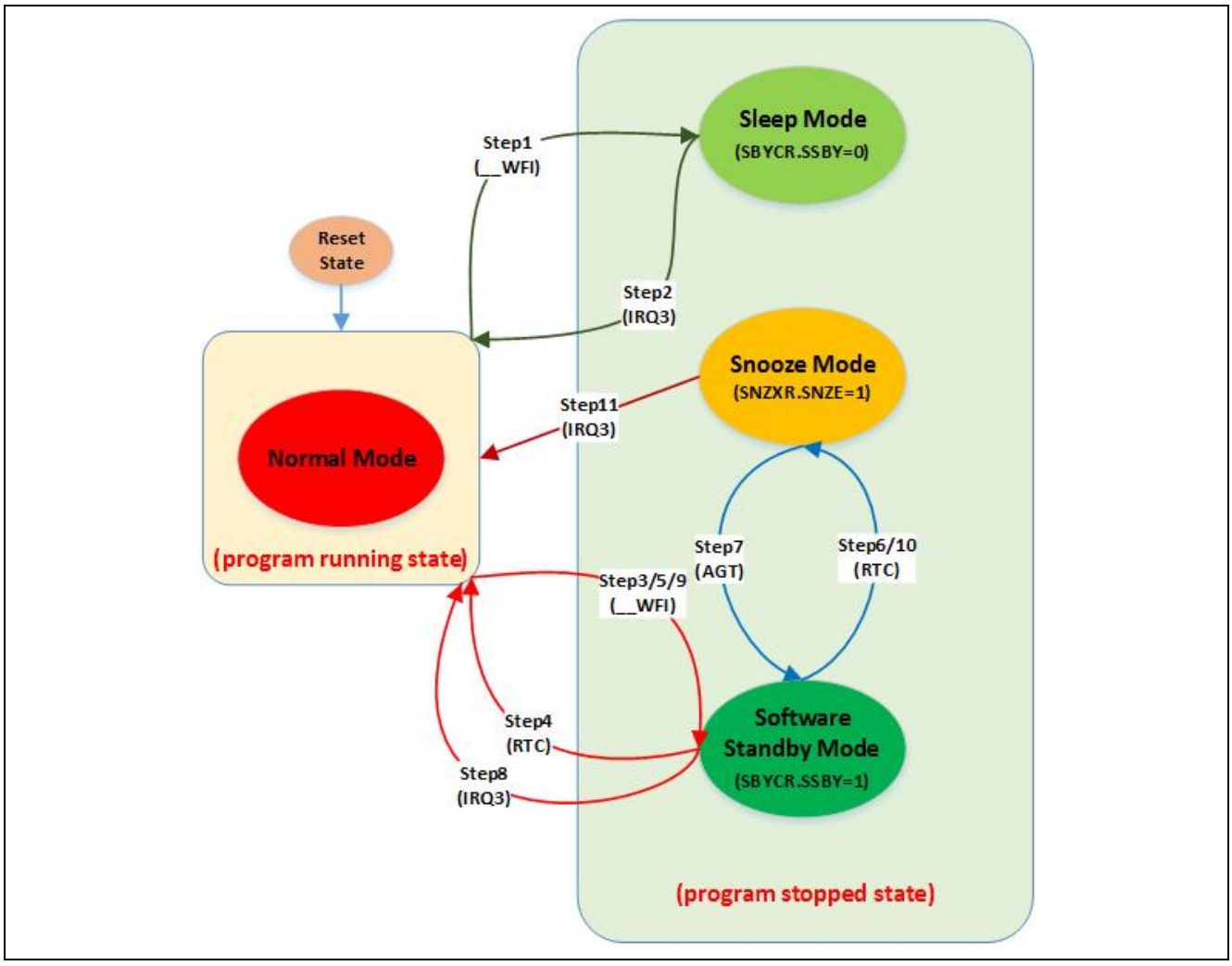


Figure 10. FSM2 state transition diagrams

3.3 Project design: User interface

The power mode transition should be controllable, and also visible to you. Therefore, two controllers are needed for triggering state changes of FSM1 and FSM2, in addition to three LEDs to indicate different modes.

Three push buttons, **S1**, **S2**, **S3**, and potentiometer **POT1** on the DK-S124 MCU board control the Low Power Mode (LPM) transition. Turn **POT1** to select different power control modes in the FSM1, such as the High-speed, Middle-speed, Low-speed and Low-voltage modes. Then, press **S1** to enter the FSM2 for transitioning to different LPMs, and **S2** to generate `PORT_IRQ3` to trigger mode transitions from the Sleep/SBY/SNZ to the Normal mode. Use **S3** to reset the device. Note that the jumper J20 has to be open to disconnect the accelerometer U13 from **S1**. The Micro USB connector J18 is connected to your PC host for debugging and programming through the J-Link® protocol.

The following figure shows a simple user interface design.

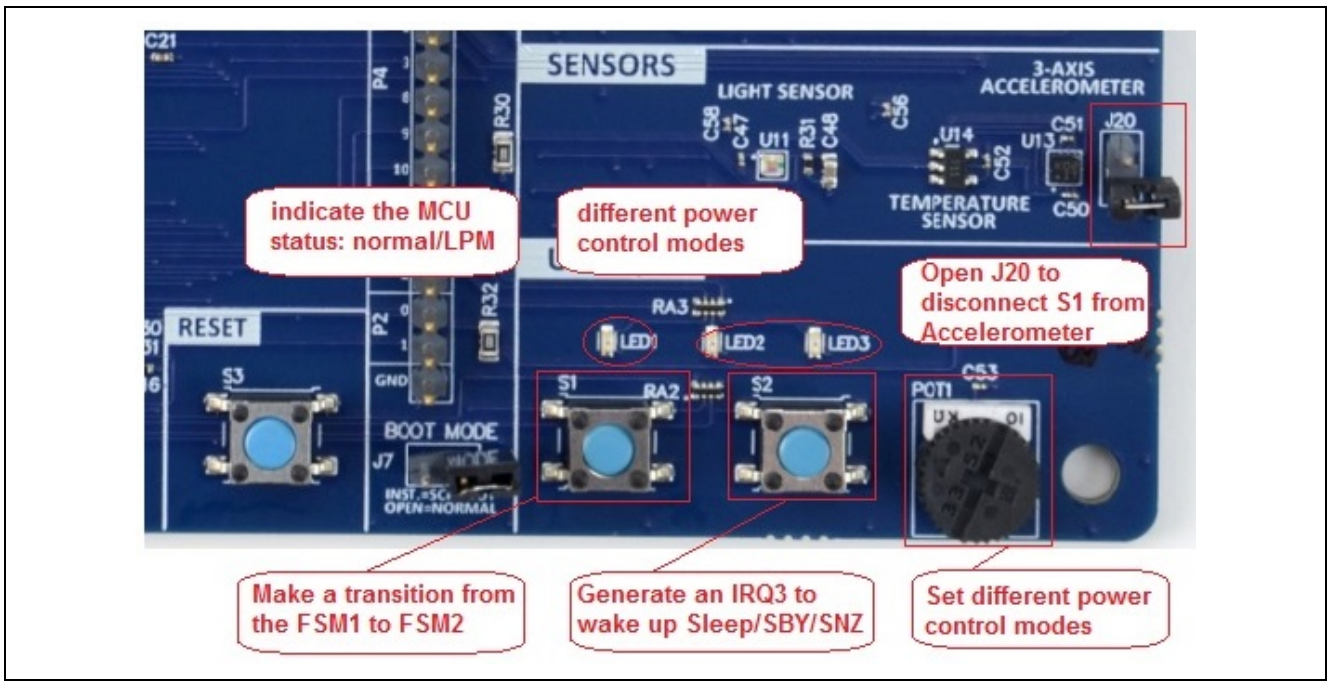


Figure 11. User interface for DK-S124 v3.0

3.3.1 Potentiometer to select power controlling modes in FSM1

After the potentiometer is turned on the board, its positions or values can be read through the ADC module in the S124 device, and different conditions can be set for selecting power controlling modes. The following table shows a possible configuration.

Table 3. FSM1 transition controlled by the potentiometer values

Conditions	ADC values	LED2, LED3
`ConditionToHSM`	(14000, 17000)	11
`ConditionToMSM`	(3000, 14000]	10
`ConditionToLVM`	(800,3000]	01
`ConditionToLSM`	(0, 800]	00

The values can be adjusted after calibrating the potentiometer.

3.3.2 LED2 and LED3 indicating states of FSM1

Even though you can use a multifunction meter on the current measurement resistor (R4) to observe the difference for each mode, a coding of LED2 and LED3 indicate different power controlling modes (see Table 3).

3.3.3 Button S1 to initiate low power mode transition in FSM2

The transitions between an LPM and the Normal mode is triggered by interrupts generated by the RTC timer, AGT1 timer, and IRQ3, which are in the available interrupt source list for requesting or ending the LPM.

The Sleep mode can be cancelled by any interrupt, so an IRQ3 is used here. The transition from the Normal mode into one of the LPMs is triggered by pressing **S1**, after the pin P2_6 is disconnected from the accelerometer with J20.

3.3.4 LED1 for showing CPU status: normal or sleep

LED1 is used to show whether the CPU is in Normal or LPM. If LED1 is on, it indicates that the CPU is in Normal mode, and if LED1 is off, it indicates that the CPU is in an LPM.

4. Project Implementation: e² studio and SSP

To switch between different power controlling modes and low power modes using bare-metal programming, you must set all controls, register by register, and bit by bit in a specified order. It is a tedious and error-prone procedure. Renesas Synergy provides an ISDE tool, e² studio, and a software library/API to overcome those barriers and to accelerate project development. This section gives a simple summary of this tool and library.

4.1 Synergy e² studio

The Synergy™ e² studio, Integrated Solution Development Environment (ISDE) is a complete development and debug environment based on the popular Eclipse CDT project. It allows engineers to integrate a wide range of compilers for exploring different optimizations on running time and memory space.

The latest version of e² studio is available for download at www.renesas.com/synergy/software. You need to create an account or sign in to your account, then download the latest revision of the e² studio (ISDE). After following the installation instructions, you select the **Renesas Synergy™ Device Family** in the e² studio **Setup** dialog, then a GNU Arm compiler, such as GCC Arm Embedded 4.9.3.20150529.

4.2 Synergy Software Package (SSP)

The Renesas Synergy Software Platform (SSP), is a complete and qualified platform for developing embedded and IoT applications. It provides engineers with a platform that already has basic system elements implemented, configured, and tested. Engineers can eliminate the time normally needed to implement and integrate baseline functionality and move almost immediately to product design, potentially reducing time to market by months.

The latest version of SSP is available for download at www.renesas.com/synergy/software. Follow the installation instructions to install SSP into the e² studio.

SSP provides a power profile framework for you to put the MCU into one of several available low power configurations. The application also may make API calls that place it into a low power Sleep mode from which an external interrupt, or periodic RTC interrupt may awaken it.

For example, to make a switch from the High-speed mode into the Low-speed mode, you use the following code:

```
if (g_lpm0.p_api->operatingPowerModeSet(LPM_OPERATING_POWER_LOW_SPEED_MODE,
                                                    LPM_SUBOSC_OTHER)){
          while(1);
       }
```

4.3 SSP configuration panels in e² studio

Assuming that a project **`LPM_TRANS`** has been created, clicking the file `configuration.xml` under the **Synergy Configuration** tab shows a window for the SSP setup (see the following figure). The SSP version and board information are displayed on the BSP panel.

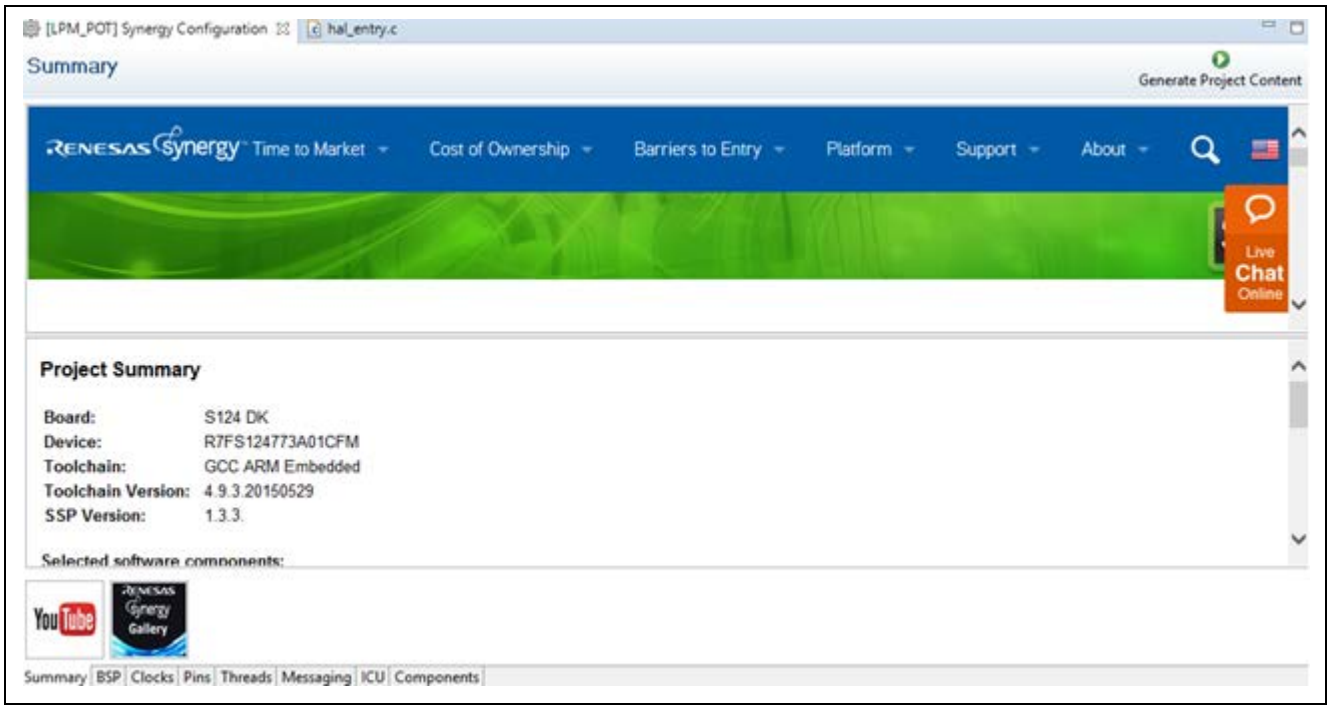


Figure 12. SP configuration panel on the e²studio

4.4 Default Clock Frequency Setup

The default clock frequencies after power-on is set up using the SSP panel, where users have five different internal clock sources available:

- Main clock oscillator (XTAL 12 MHz)
- Sub clock oscillator (SUBVLK 32,768 Hz)
- High-speed on-chip oscillator (HOCO 48 MHz)
- Middle-speed on-chip oscillator (MOCO 8 MHz)
- Low-speed on-chip oscillator (LOCO 32,768 Hz).

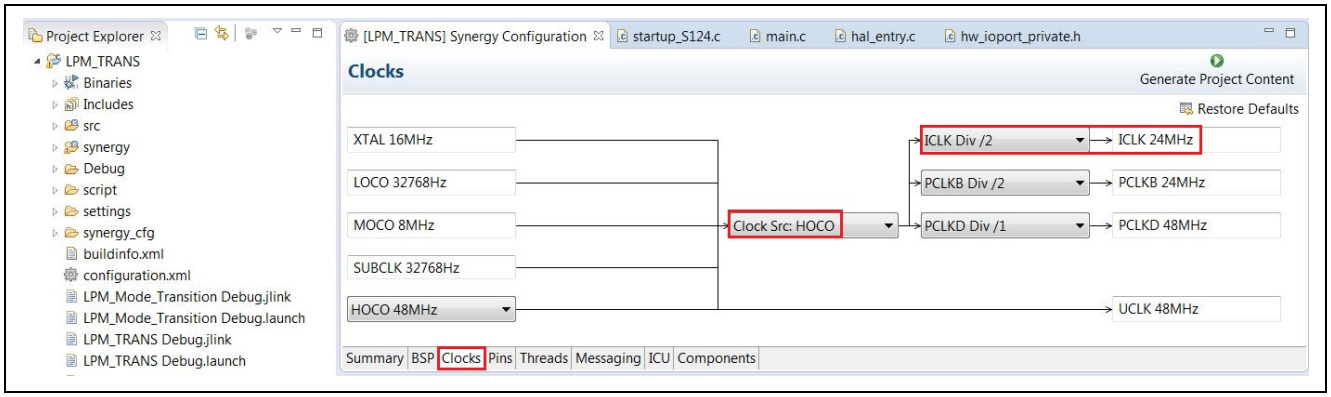


Figure 13. CGC setup on the SSP clock panel

The generated clocks are:

- System clock (ICLK), that is the operating clock for the CPU, DTC, Flash, FlashIF, and SRAM.
- Peripheral module clock (PCLKB), that is the operating clocks for peripheral modules, CAC, ELC, I/O Ports, POEG, RTC, and so on.
- Peripheral module clock (PCLKD), that is the operating clocks for GPT and ADC14 modules.

Note: System clock (ICLK), peripheral module clock (PCLKB and PCLKD), and FlashIF clock (ICLK) must be set according to the following table (from Table 8.2 in the *S124 User's Manual*).

Table 8.2 Clock generation circuit specifications (internal clock) (1/2)

Item	Clock Source	Clock Supply	Specification
System clock (ICLK)	MOSC/SOSC/HOCO/ MOCO/LOCO	CPU, DTC, FLASH, SRAM, FlashIF	Up to 32 MHz Division ratio: 1/2/4/8/16/32/64 1 MHz to 32 MHz (P/E)
Peripheral module clock B (PCLKB)	MOSC/SOSC/HOCO/ MOCO/LOCO	Peripheral module (CAC, ELC, I/O Ports, POEG, RTC, WDT, IWDT, SCI, IIC, CAN, SPI, CRC, GPT, ADC14, DAC12, TSN, DOC, AES, TRNG, KINT, AGT, USBFS, ACMPLP, and CTSU)	Up to 32 MHz Division ratio:1/2/4/8/16/32/64
Peripheral module clock D (PCLKD)	MOSC/SOSC/HOCO/ MOCO/LOCO	Peripheral module (GPT count clock, ADC14 conversion clock)	Up to 64 MHz Division ratio: 1/2/4/8/16/32/64
USB clock (UCLK)	HOCO	USBFS	48 MHz
CAN clock (CANMCLK)	MOSC	CAN	1 MHz to 20 MHz
AGT clock (AGTCLK) (AGTSCLK/AGTLCLK)	SOSC/LOCO/PCLKB	AGT	Up to 32 MHz

4.5 Reading the Potentiometer with the SSP ADC Driver

As specified in the schematics for the DK-S124 Synergy MCU, the potentiometer POT1 is sampled through analog channel AN007 on the pin P0_12. So, the analog input function on pin P0_12 must be enabled with the e^2 studio pin configurator. Figure 14 shows the circuit diagram for POT1, while Figure 15 shows the P0_12 pin configuration.

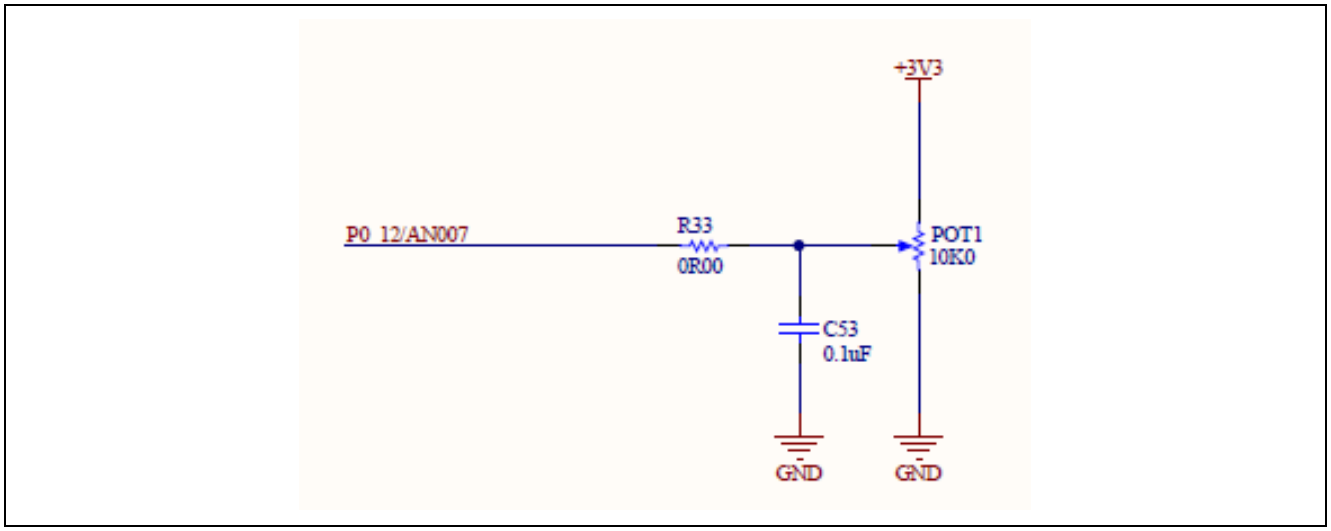


Figure 14. DK-S124 potentiometer circuit schematic

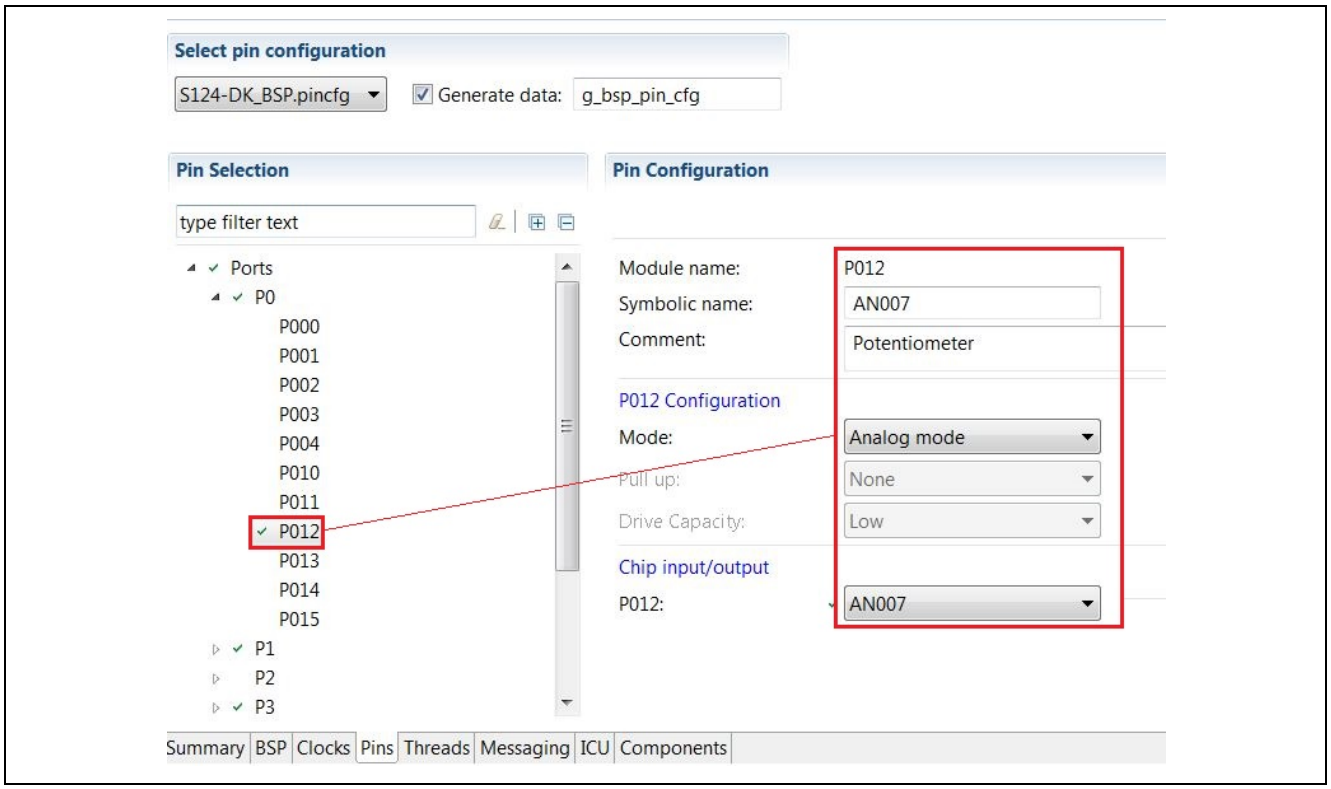


Figure 15. P0_12 Pin Configuration

An instance of the SSP ADC driver is called from the **Threads** panel of the SSP configuration, shown in the following figure.

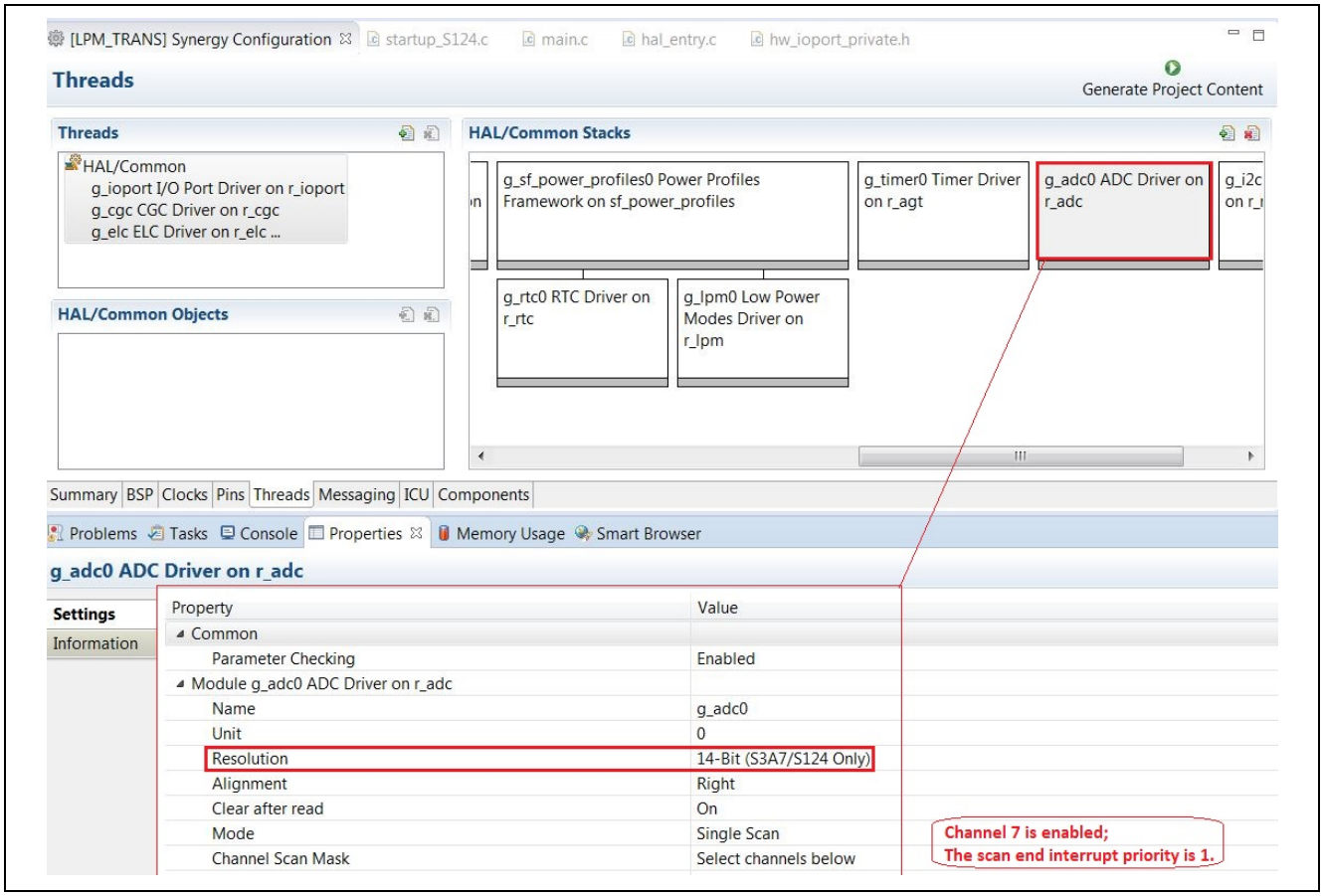


Figure 16. Set up the ADC API on the Threads panel

The Single Scan mode is selected and generates a priority 1 interrupt at the end of scanning. The procedure for reading **POT1** can be simplified as shown in the following code:

```
//read the Potentiometer
static void vReadPOT(void) {
    if (g_adc0.p_api->open( g_adc0.p_ctrl, g_adc0.p_cfg )) {
        while(1);
    }
    if (g_adc0.p_api->scanCfg( g_adc0.p_ctrl, g_adc0.p_channel_cfg )) {
        while(1);
    }
    if (g_adc0.p_api->scanStart( g_adc0.p_ctrl )) {
        while(1);
    }
    if (g_adc0.p_api->read( g_adc0.p_ctrl, ADC_REG_CHANNEL_7, &u16ADCValue)){
        while(1);
    }
    if (g_adc0.p_api->scanStop( g_adc0.p_ctrl )) {
        while(1);
    }
    if (g_adc0.p_api->close( g_adc0.p_ctrl )) {
        while(1);
    }
}
```

The u16ADCValue is then used for selecting a power control mode in the subroutine, `vChangePCM(void)`.

4.6 Waking up from Software Standby Mode with the SSP RTC Driver

One of the LPM transition paths in the FSM2 is step 3, going from Normal to the Software Standby mode after an IRQ3 is generated by pressing **S2**. Then, step 4 transitions the MCU back to Normal mode after an interrupt is generated from the RTC periodic timer. This transition has been used for running the ultra-low-power measurement standard, EEMBC ULPBench on the DK-S124 MCU. The following figure shows the EEMBC ULP running for 12 cycles or seconds for low power modes and normal mode.

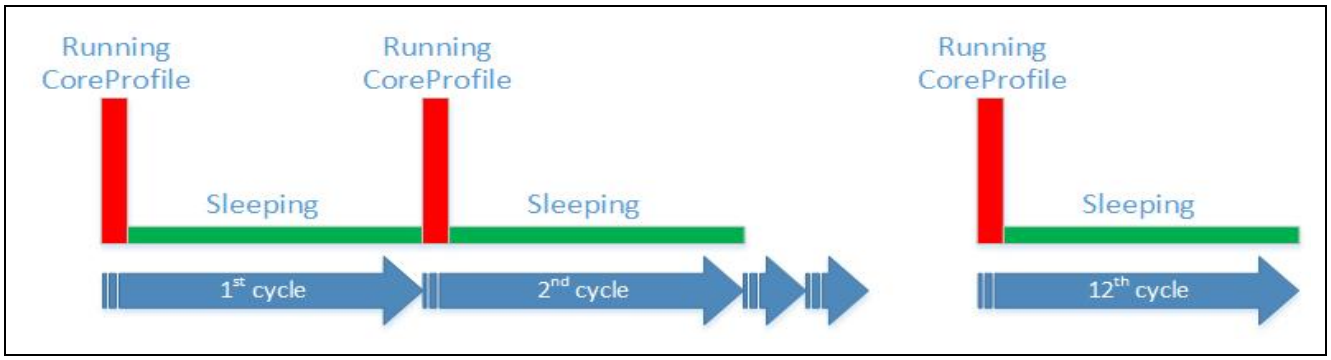


Figure 17. Running/sleeping cycles of the EEMBC ULPB

The following figure shows conditions or interrupt sources from the Software Standby mode are specified in the Wake up Interrupt Enable Register (WUPEN). For details, see section 12.2.8 in the *S124 User's Manual*.

12.2.8 Wake Up Interrupt Enable Register (WUPEN)

Address(es): ICU.WUPEN 4000 61A0h

	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	IIC0WUPEN	AGT1CBWUPEN	AGT1CAWUPEN	AGT1UDWUPEN	USBFSWUPEN	—	RTCPRDWUPEN	RTCALMWUPEN	ACMPLP0WUPEN	—	—	—	LVD2WUPEN	LVD1WUPEN	KEYWUPEN	IWDTWUPEN
Value after reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	IRQWUPEN[7:0]							
Value after reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Bit name	Description	R/W
b7 to b0	IRQWUPEN[7:0]	IRQ Interrupt Software Standby Returns Enable	0: Software standby returns by IRQ interrupt disabled 1: Software standby returns by IRQ interrupt enabled.	R/W
b16	IWDTWUPEN	IWDT Interrupt Software Standby Returns Enable	0: Software standby returns by IWDT interrupt disabled 1: Software standby returns by IWDT interrupt enabled.	R/W
b17	KEYWUPEN	Key Interrupt Software Standby Returns Enable	0: Software standby returns by KEY interrupt disabled 1: Software standby returns by KEY interrupt enabled.	R/W
b18	LVD1WUPEN	LVD1 Interrupt Software Standby Returns Enable	0: Software standby returns by LVD1 interrupt disabled 1: Software standby returns by LVD1 interrupt enabled.	R/W
b19	LVD2WUPEN	LVD2 Interrupt Software Standby Returns Enable	0: Software standby returns by LVD2 interrupt disabled 1: Software standby returns by LVD2 interrupt enabled.	R/W
b22 to b20	—	Reserved	This bit is read as 0. The write value should be 0.	R/W
b23	ACMPLP0WUPEN	ACMPLP0 Interrupt Software Standby Returns Enable	0: Software standby returns by ACMPLP0 interrupt disabled 1: Software standby returns by ACMPLP0 interrupt enabled.	R/W
b24	RTCALMWUPEN	RTC Alarm Interrupt Software Standby Returns Enable	0: Software standby returns by RTC alarm interrupt disabled 1: Software standby returns by RTC alarm interrupt enabled.	R/W
b25	RTCPRDWUPEN	RTC Period Interrupt Software Standby Returns Enable	0: Software standby returns by RTC period interrupt disabled 1: Software standby returns by RTC period interrupt enabled.	R/W
b26	—	Reserved	This bit is read as 0. The write value should be 0.	R/W
b27	USBFSWUPEN	USBFS Interrupt Software Standby Returns Enable	0: Software standby returns by USBFS interrupt disabled 1: Software standby returns by USBFS interrupt enabled.	R/W
b28	AGT1UDWUPEN	AGT1 Underflow Interrupt Software Standby Returns Enable	0: Software standby returns by AGT1 underflow interrupt disabled 1: Software standby returns by AGT1 underflow interrupt enabled.	R/W
b29	AGT1CAWUPEN	AGT1 Compare Match A Interrupt Software Standby Returns Enable	0: Software standby returns by AGT1 compare match A interrupt disabled 1: Software standby returns by AGT1 compare match A interrupt enabled.	R/W
b30	AGT1CBWUPEN	AGT1 Compare Match B Interrupt Software Standby Returns Enable	0: Software standby returns by AGT1 compare match B interrupt disabled 1: Software standby returns by AGT1 compare match B interrupt enabled.	R/W
b31	IIC0WUPEN	IIC0 Address Match	0: Software standby returns by IIC0 address match	R/W

Figure 18. Wake up Interrupt Enable Register (WUPEN)

The Synergy SSP provides APIs for using RTC and LPM libraries in the Power Profile framework and is configured with the **Threads** panel of the SSP Configuration in e² studio.

The following figure shows the setup of the RTC API on the **Threads** panel.

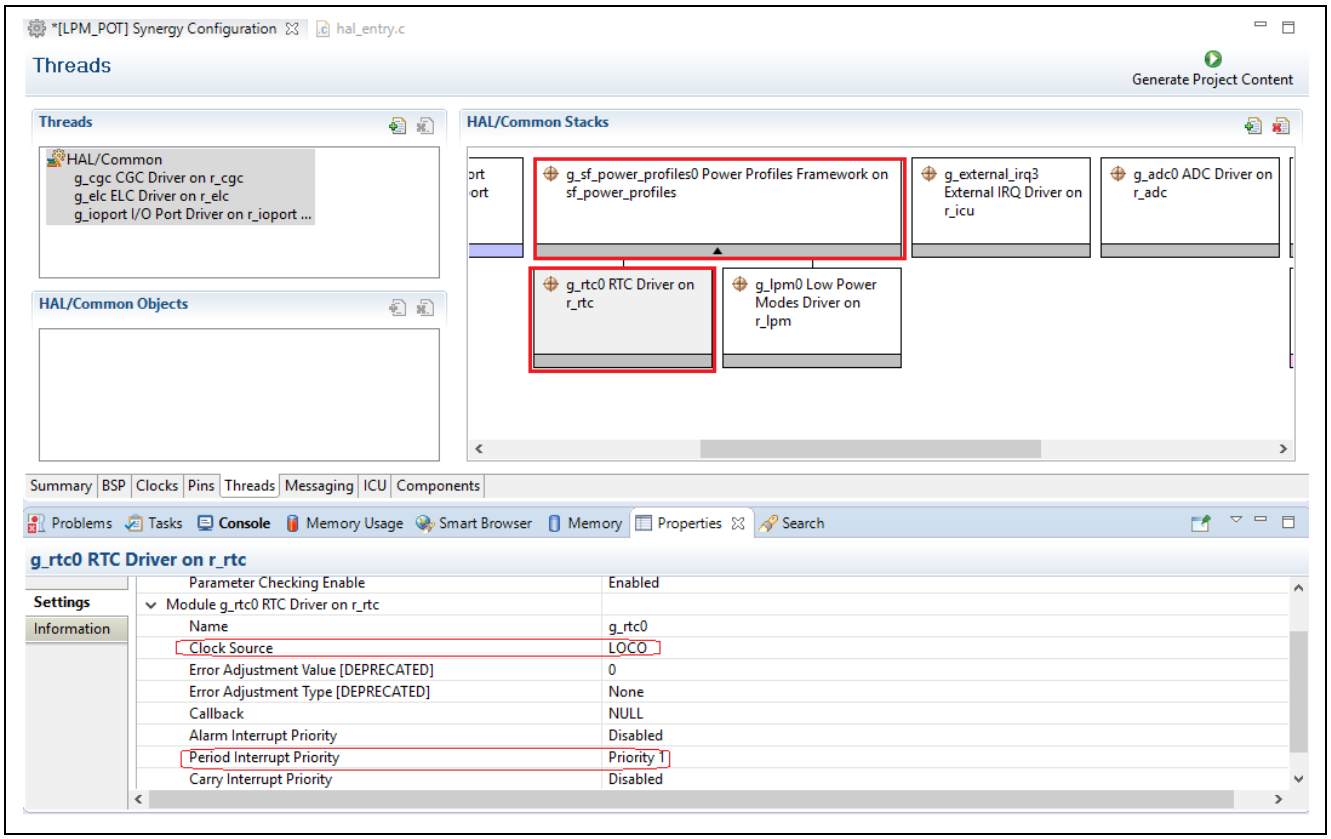


Figure 19. Setting up the RTC API on the Thread panel

An RTC-based timer wakes up the MCU every second. The counting clock source could be from sub-oscillator SOSC (32,768 Hz) or LOCO (32,768 Hz). The LOCO is more energy efficient, so is selected in the example. The following figure shows the setup of the LPM API on the **Threads** panel.

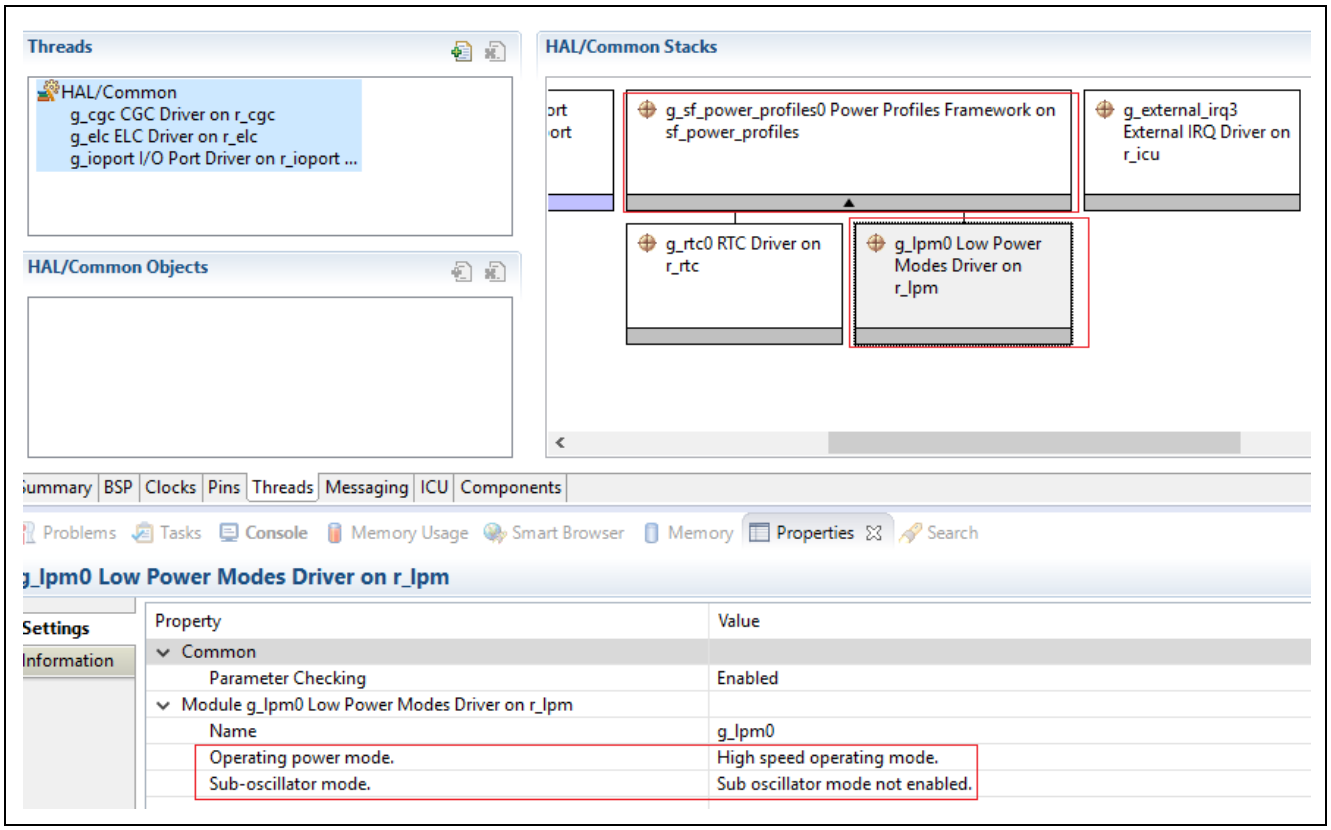


Figure 20. Setting up the LPM API on the Thread panel

The High-speed mode is selected as a default state before entering one of the LPMs; it can be changed in the program with a simple function call, such as switching into the Middle-Speed mode in the subroutine, `vChangePCM(void)`.

```
      if (g_lpm0.p_api-
>operatingPowerModeSet(LPM_OPERATING_POWER_MIDDLE_SPEED_MODE,
                                                LPM_SUBOSC_OTHER)){
            while(1);
      }
```

To implement such a transition in the programming, use the following code to enter and return from the Software Standby mode.

```
//make a transition: Normal->Standby->Normal
void vTransition_NomSbyNom(void) {
    if (g_lpm0.p_api->lowPowerCfg(LPM_LOW_POWER_MODE_STANDBY,
                                  LPM_OUTPUT_PORT_ENABLE_HIGH_IMPEDANCE,
                                  LPM_POWER_SUPPLY_DEEPCUT3,
                                  LPM_IO_PORT_NO_CHANGE)) {
       while (1);
    }

    //Get WUPEN reg value
    if (g_lpm0.p_api->wupenGet(&u32WupenRegVal)) {
       while(1);
    }

    //Set RTC interrupt as condition for waking up SBY
    if ((g_lpm0.p_api->wupenSet(u32WupenRegVal | WUPEN_RTC)) ) {
       while (1);
    }

    vInitRTC();
```

```
    //Start the RTC
    vStartRTC();

    //LED1 Off
    g_ioport_on_ioport.pinWrite(LED1_RED, LED_OFF);
    #ifdef DEBUG
    printf("\n in software standby mode (step 3)\n");
    #endif

    //WFI
    if ((g_lpm0.p_api->lowPowerModeEnter()) ){
       while (1);
    }
    //wake up after RTC timer out (2sec)
    vToggleLED1(20);
    g_ioport_on_ioport.pinWrite(LED1_RED, LED_ON);

    //Stop the RTC
    vStopRTC();

    //reset the WUPEN
    if (g_lpm0.p_api->wupenGet(&u32WupenRegVal)) {
       while(1);
    }

    if (g_lpm0.p_api->wupenSet(u32WupenRegVal &
(uint32_t)~(WUPEN_RTC|WUPEN_RTC)) ) {
       while (1);
    }

    #ifdef DEBUG
    printf("\n rtc wakeup to normal mode (step 4)\n");
    #endif
}
```

5. Importing the Project into e² studio

See the *Renesas Synergy Project Import Guide* included in this package for instructions on importing the project into e² studio and building/running the project.

6. Testing and Observation

Since this project is implemented with a single thread and with the same priority level for interrupts generated from the controllers, testing and operating are relatively simple. You can go through the power mode transitions in two ways:

1. Adjusting the potentiometer POT1 to set a Power Control mode - High-speed mode, Middle-speed mode, Low-voltage mode, and Low-speed mode as shown in the FSM1 diagram, then pressing the button **S1** to transition the MCU to an LPM as shown in the state transition diagram of the FSM2.
2. Modifying the `void hal_entry(void)` by uncommenting/commenting the targeted transition/states to a specific mode:

```
 //for testing LPM states
             //lpm_transition = NORMAL_SLEEP_NORMAL;
             //lpm_transition = NORMAL_SBY_NORMAL;
             //lpm_transition = NORMAL_SBY_SNZ_SBY_NORMAL;
 //lpm_transition = NORMAL_SBY_SNZ_NORMAL;
```

With these testing approaches, all of LPM transitions can be performed. However, some combinations of requesting/ending Snooze modes remain under further investigation.

Table 4. Observation of mode transitions

	Nm -> Slp -> Nm	Nm -> SBY -> Nm	Nm -> SBY -> SNZ -> SBY -> Nm	Nm -> SNZ -> Slp -> Nm
High Speed	Ok	Ok	Ok	Ok
Middle Speed	Ok	Ok	Ok	Ok
Low Voltage	Ok	Ok	Ok	Ok
Low Speed	Ok	Ok	Ok	Ok

7. References

Renesas Electronics Corporation. (2016). *S124 User's Manual Microcontrollers Renesas Synergy Platform.*

Renesas Electronics Corporation. (2018). *Importing a Renesas Synergy Project* (r11an0023eu0121-synergy-ssp-import-guide.pdf).

Website and Support

Visit the following vanity URLs to learn about key elements of the Synergy Platform, download components and related documentation, and get support.

Synergy Software	www.renesas.com/synergy/software
Synergy Software Package	www.renesas.com/synergy/ssp
Software add-ons	www.renesas.com/synergy/addons
Software glossary	www.renesas.com/synergy/softwareglossary
Development tools	www.renesas.com/synergy/tools
Synergy Hardware	www.renesas.com/synergy/hardware
Microcontrollers	www.renesas.com/synergy/mcus
MCU glossary	www.renesas.com/synergy/mcuglossary
Parametric search	www.renesas.com/synergy/parametric
Kits	www.renesas.com/synergy/kits
Synergy Solutions Gallery	www.renesas.com/synergy/solutionsgallery
Partner projects	www.renesas.com/synergy/partnerprojects
Application projects	www.renesas.com/synergy/applicationprojects
Self-service support resources:	
Documentation	www.renesas.com/synergy/docs
Knowledgebase	www.renesas.com/synergy/knowledgebase
Forums	www.renesas.com/synergy/forum
Training	www.renesas.com/synergy/training
Videos	www.renesas.com/synergy/videos
Chat and web ticket	www.renesas.com/synergy/resourcelibrary

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	Jul.08.16	—	Initial version
1.10	Oct.18.16	—	Updated for SSP 1.2.0-b1, and DK-S124 (v3.0)
1.11	Nov.30.16	—	Added support for IAR EW
1.20	Feb.23.17	—	Updated for SSP 1.2.0
1.21	Aug.04.17	—	Updated to SSP 1.3.0
1.22	Sep.27.17	1	Required resources of SSP version changed
1.23	Jan.16.18	—	Updated to SSP v1.3.3
1.24	Mar.01.18	—	Updated for SSP v1.4.0
1.25	Mar.21.19	—	Updated for SSP v1.6.0

Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving patents, copyrights, or other intellectual property rights of third parties, by or arising from the use of Renesas Electronics products or technical information described in this document, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.
3. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.
5. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The intended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below.
   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; industrial robots; etc.
   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.

   Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems; surgical implantations; etc.), or may cause serious property damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user’s manual or other Renesas Electronics document.
6. When using Renesas Electronics products, refer to the latest product information (data sheets, user’s manuals, application notes, “General Notes for Handling and Using Semiconductor Devices” in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.
7. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, such as safety design for hardware and software, including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult and impractical, you are responsible for evaluating the safety of the final products or systems manufactured by you.
8. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
9. Renesas Electronics products and technologies shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.
10. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.
11. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

(Note1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.

(Note2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

(Rev.4.0-1 November 2017)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan
www.renesas.com

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.

Contact information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit:
www.renesas.com/contact/.

© 2019 Renesas Electronics Corporation. All rights reserved.