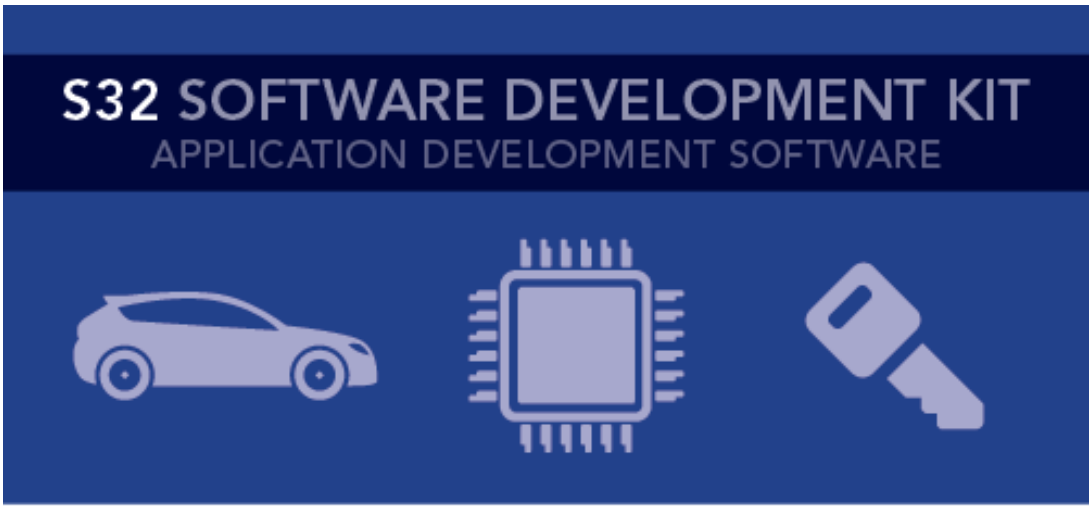

S32 SDK Release Notes

Version 4.0.2 RTM

Contents

1. Description

The S32 Software Development Kit (S32 SDK) is an extensive suite of peripheral abstraction layers, peripheral drivers, RTOS, stacks and middleware designed to simplify and accelerate application development on NXP S32K microcontrollers.

All software included in this release have RTM quality level in terms of features, testing and quality documentation, according to NXP software release criteria. RTM releases contain all planned features implemented and tested. RTM releases are candidates that can be used in production.

This SDK can be used standalone or it can be used with S32 Design Studio IDE (see Supported hardware and compatible software).

Refer to *Product license (License.txt)* for licensing information and *Software content register (S32_SDK_for_S32K1xx_RTM_4.0.2_SCR.txt)* for the Software contents of this product. The files can be found in the root of the installation directory.

For support and issue reporting use the following ways of contact:

- NXP Support to https://www.nxp.com/support/support:SUPPORTHOME
- NXP Community https://community.nxp.com/community/s32/s32k

2. New in this release

2.1 New features from S32K1xx RTM 4.0.2.

Migration tool

- Supported migrating component from Processor Expert to S32CT. It is a part of migration tool which helps user to migrate their project from SDK RTM 3.0.x (Configuration tool is Processor Expert) to SDK RTM 4.0.2 (Configuration tool is S32CT). This tool is integrated to S32 Design Studio 3.4 and releases by S32 Design Studio team.

Examples

- Upgraded S32DS version 3.3 to version 3.4.
- Supported new version of EVB board for S32K142, S32K144 and S32K148.

eDMA

- [S32CT] Supported multiple user configurations instead of one configuration only.

Linstack

- [S32CT] Supported LDF file selection with both relative and absolute path browsing.

Lpi2c

- Added LPI2C_DRV_SetMasterBusIdleTimeout function that should be called at initialization to recover the bus busy.

Header File

- Applied new header file according to new reference manual version: S32K1XXRM Rev. 13, 04/2020.

SBC_UJA116X

- [S32CT] Updated new device variants replacing old one (UJA116xA instead of UJA116x).

Silicon version

- Supported verifying on new part of S32K144W with 0P64A maskset.

2.2 List of fixed issues from S32K1xx RTM 4.0.1.

Component	Issue
Adc_pal	Fixed warning in generated code when DelayType of group is not NoDelay.
Clock_manager	Fixed ENET clock frequency get wrong clock source. ENET clock frequency will be got from DIV2 source instead of DIV1 source.
Clock_manager	Corrected constraint for SOSC source when selecting Medium range (4Mhz - 8MHz).
Clock_manager	Removed Trace clock on S32K11x derivatives due to it is only available on S32K14x platform.
Clock_manager	Added constraint that shows error when the system clock is inactive.
CSEc, Security_pal	Added support for multiple FTFx configurations.
Flash	Fixed issue of flash component: generate callback with NULL value.
FreeRTOS	Updated S32CT to support IAR compiler
Enet	Added the constraint for the validation of timer callback's name.

Example	Added the erasing all FLASH block sequence for csec_keyconfig example (S32K14XW).
I2s_pal	Fixed issue: generate user callback with the valid value.
I2s_pal	Added support for multiple user configurations.
Interrupt_manager	Fixed the linker definition that the INT_SYS_InstallHandler function should work normally and should not depend on the address of vector table in Flash.
Lpi2c	Corrected the order of processes in the interrupt handler of LPI2C slave to work with more than one interrupt flag at the same time.
Phy	Fixed issue: TJA110x did not switch to manage mode at initialization function.
Phy	Fixed issue about wake up and sleep settings that related to TJA1101/TJA1102 to be TC-10 compliant.
Phy	Added known limitation to doxygen file: When PHY TJA1102 device is used in the application, User must add 2 configurations with PHY TJA110x type. The configurations must be arranged in order TJA1102_P0 then TJA1102_P1. Because TJA1102_P1 is always initialized after TJA1102_P0.
Pins	Fixed issue that cannot configure Drive Strength for some pins on S32K146 and S32K148 derivatives.
Sai	Fixed issue: generate user callback with the valid value.
Sai	Fixed S32CT validation for SAI channels and the first bit index parameter.
Uart_pal	Corrected flexio_uart component ID in UartToolchainSettings to fix missing flexio_uart driver when using uart_pal over flexio_uart.

3. Software Contents

3.1 Drivers

- ADC
- CMP
- CRC
- CSEc
- DMA
- EIM
- ENET
- ERM
- EWM
- FLASH
- FLASH_MX25L6433F
- FLEXCAN
- FLEXIO (I2C, SPI, I2S, UART profiles)
- FTM
- LIN
- LPI2C
- LPIT
- LPSPI
- LPTMR
- LPUART
- MCU (Clock Manager, Interrupt Manager, Power Manager)
- MPU
- PINS
- PDB
- PHY_TJA110x
- QSPI
- RTC
- SAI
- TRGMUX
- WDOG

3.2 PAL

- ADC
- CAN
- I2C
- I2S
- IC
- MPU
- OC
- PWM
- SECURITY

- SPI
- TIMING
- UART
- WDG

3.3 Middleware

- LIN stack – provides support for LIN 1.3, LIN 2.0, LIN 2.1, LIN 2.2 and J2602 communication protocols
- Support interleave mode for versions: LIN 1.3, LIN 2.0, LIN 2.1, LIN 2.2 and J2602
- TCP/IP stack – available for S32K148, for more details see TCP/IP stack release notes (in the SDK installation folder)
- SBC drivers – provides support for UJA116x System Basis Chips

Note: *For ISELED contact your Sales representative or FAE for more information.*

3.4 Libraries

- sCST RTM 1.0.1 – available for S32K11x
- sCST RTM 1.0.5 – available for S32K14x
- AMMCLib RTM 1.1.21 - available for S32K1xx

3.5 RTOS

- FreeRTOS version 10.2.1

4. Documentation

- Quick start guide available in *“doc”* folder.
- User and integration manual available at *“doc\Start_here.html”.*
- Driver user manuals available in *“doc”* folder.

5. Examples

Type	Name	Description
Driver examples	adc_hwtrigger	Uses PDB to trigger an ADC conversion with a configured delay and sends the result to host via LPUART.
	adc_pal	The application uses ADC PAL to trigger multiple executions of two groups of ADC conversions: first group configured for SW triggering and second group for HW triggering. For each execution of a group of conversions, an average conversion value is computed in SW, and the average value is printed on UART.
	adc_swtrigger	Uses software trigger to periodically trigger an ADC conversion and sends the result to host via LPUART.
	can_pal	Shows the usage of the CAN PAL module with Flexible Data Rate.
	cmp_dac	Configures the analog comparator to compare the input from the potentiometer with the internal DAC (configured to output half of the reference voltage) and shows the result using the LEDs found on the board.
	crc_checksum	The CRC is configured to generate the cyclic redundancy check value using 16 and 32 bits wide result.
	csec_keyconfig	The example demonstrates how to prepare the MCU before using CSEc (Key configuration, flash partitioning).
	edma_transfer	Demonstrates the following eDMA use cases: single block memory to memory transfer, a loop memory to memory transfer, memory to memory transfer using scatter/gather, LPUART transmission/reception using DMA requests.
	eim_injection	The purpose of this demo is to provide the user check able correction of ECC. Module EIM enable user addition error to RAM (low). And enable user can use module ERM to read address that user already error to region RAM. User seen RED_LED off when ERM read right address which EIM injected error.
	enet_loopback	Shows the usage of the ENET module with loopback mode using the S32 SDK API.
	erm_report	The purpose of this driver application is to show the user how to use the EWM from the S32K148 using the S32 SDK API. This Example only debug equal Flash This example use module EIM to addition error to RAM and use module ERM to read address and notify interrupt.
	ewm_interrupt	Shows the usage of the EWM driver.

	flash_partitioning	Writes, verifies and erases data on Flash.
	flexio_i2c	Demonstrates FlexIO I2C emulation. Use one instance of FlexIO and one instance of LPI2C to transfer data on the same board.
	flexio_i2s_master	Demonstrates FlexIO I2S emulation for master configurations. Use one instance of FlexIO to instantiate master drivers to transfer data on the same board.
	flexio_i2s_slave	Demonstrates FlexIO I2S emulation for slave configurations. Use one instance of FlexIO to instantiate slave drivers to transfer data on the same board.
	flexio_spi	Demonstrates FlexIO SPI emulation for both master and slave configurations. Use one instance of FlexIO to instantiate master and slave drivers to transfer data on the same board.
	flexio_uart	Demonstrates FlexIO UART emulation for both TX and RX configurations. Use one instance of FlexIO to instantiate UART transmitter and receiver drivers to transfer data from/to the host.
	ftm_combined_pwm	Uses FTM PWM functionality using two combined channels to light two LEDs on the board with opposite pulse width. The light's intensity is increased and decreased periodically.
	ftm_periodic_interrupt	Uses FTM Timer functionality to trigger an interrupt at a given period which toggles a LED.
	ftm_pwm	Uses FTM PWM functionality using a single channel to light a LED on the board. The light's intensity is increased and decreased periodically.
	ftm_signal_measurement	Using one FTM instance the example application generates a PWM signal with variable frequency which is measured by another FTM instance configured in signal measurement mode.
	i2c_pal	Shows the usage of I2C PAL driver in both master and slave configurations using FLEXIO and LPI2C
	ic_pal	Shows the usage of the IC_PAL
	i2s_pal_master	Demonstrates I2S_PAL emulation for master configurations. Use one instance of FlexIO or SAI to instantiate master drivers to transfer.
	i2s_pal_slave	Demonstrates I2S_PAL emulation for slave configurations. Use one instance of FlexIO or SAI to instantiate slave drivers to receive.
	lin_slave_baremetal	Shows the usage of LIN driver in slave mode.
	lpi2c_master	Shows the usage of the LPI2C driver in Master configuration
	lpi2c_slave	Shows the usage of the LPI2C driver in Slave configuration

	lpit_periodic_interrupt	Shows how to initialize the LPIT to generate an interrupt every 1 s. It is the starting point for any application using LPIT.
	lpspi_dma	The application uses two on board instances of LPSPI, one in master configuration and the other one is slave to communicate data via the SPI bus using DMA.
	lpspi_transfer	Uses one instance of the LPSPI as slave to send ADC data to the master LPSPI instance which is on the same board. The master uses data received to feed a FlexTimer PWM.
	lptmr_periodic_interrupt	Exemplifies to the user how to initialize the LPTIMER so that it will generate an interrupt every 1 second. To make the interrupt visible a LED is toggled every time it occurs.
	lptmr_pulse_counter	Shows the LPTIMER pulse count functionality by generating an interrupt every 4 rising edges.
	lpuart_echo	Simple example of a basic echo using LPUART.
	mpu_memory_protection	Configures MPU to protect a memory area and demonstrates that read access is correctly restricted.
	mpu_pal_memory_protection	The purpose of this demo application is to show you how to configure and use the Memory Protection Unit PAL
	oc_pal	Shows the Periodic Event Generation functionality of the OC_PAL
	pdb_periodic_interrupt	Configures the Programmable Delay Block to generate an interrupt every 1 second. This example shows the user how to configure the PDB timer for interrupt generation. The PDB is configured to trigger ADC conversions in ADC_HwTrigger_Example.
	phy_tja1101	Shows the usage of the TJA1101 Ethernet PHY in conjunction with the S32K148 CPU using the S32 SDK API.
	power_mode_switch	Demonstrates the usage of Power Manager by allowing the user to switch to all power modes available.
	qspi_external_flash	The purpose of this demo is to present the usage of the flash_mx25l6433f (external serial flash) and QSPI drivers. The flash_mx25l6433f driver allows the application to use an external Macronix MX25L6433F serial flash device, using the QuadSPI interface for communication.
	rtc_alarm	Show the frequently used RTC use cases such as the generation of an interrupt every second and triggering an alarm.
	sai_transfer	Demonstrates the usage of the SAI module driver
	sbc_uja1169	Show the usage of the SBC UJA1169 driver with low power modes

	security_pal	This is an application created to show the generation of RND and CBC encryption/decryption of a string.
	spi_pal	The purpose of their own application is to show you how to use the LPSPI and FLEXIO Interfaces on the S32K144 using the S32 SDK API. The application uses one board instance of LPSPI in slave configuration and one board instance of FLEXIO in master configuration to communicate data via the SPI bus using interrupts.
	timing_pal	The purpose of their own application is to show you how to use the TIMING PAL over LPIT, LPTMR and FTM timers on the S32K144 using the S32 SDK API. The application uses one board instance of LPIT, LPTMR and FTM to periodically toggle 3 LEDs.
	trgmux_lpit	The purpose of this demo application is to show you how to use the Trigger MUX Control of the S32K14x MCU with this SDK.
	uart_pal_echo	The purpose of this demo is to show the user how UART PAL works over FLEXIO_UART or LPUART peripherals. The user can choose whether to use FLEXIO_UART or LPUART. The board sends a welcome message to the console with further instructions.
	wdg_pal_interrupt	The purpose of this driver application is to show the user how to use the WDG PAL from the S32K148 using the S32 SDK API. The examples use the SysTick timer from the ARM core to refresh the WDG PAL counter for 30 times. After this the WDG PAL counter will expire and the CPU will be reset.
	wdog_interrupt	Shows the basic usage scenario and configuration for the Watchdog.
Demos	adc_low_power	This demo shows the user how to reduce CPU overhead and power usage by triggering ADC conversions with the LPIT via TRGMUX. The CPU is set in the STOP mode via the Power Manager API, with the wakeup condition being the validity of the ADC conversion result, the latter being a value greater than half of the ADC reference voltage achieved by using the hardware compare functionality. If the condition is met, the value in the form of a graph is sent using LPUART and DMA to further reduce the CPU usage.
	Anfc	Shows the integration between Automotive NFC stack and S32SDK

	csec_boot_protection	Basic application that shows the boot protection functionality of the CSEc module.
	flexcan_encrypted	Uses two boards to demonstrate FlexCAN functionality with Flexible Data Rate on. LEDs on a board are toggled depending on the buttons actioned on the other board. Also demonstrates the use of SBC driver to configure the CAN transceiver from EVB board. The application is configured using CSEc to encrypt the data on enabled security parts.
	Freemaster	This demo uses the FreeMASTER Run-Time Debugging Tool to visualize ADC conversions and allows the user to monitor the ADC sampling rate for different ADC configurations (ADC sampling time and resolution can be controlled through FreeMASTER Variable Watch). The application uses FreeMASTER SCI driver for communication.
	Freertos	This demo application demonstrates the usage of the SDK with the included FreeRTOS. Uses a software timer to trigger a led and waits for a button interrupt to occur.
	hello_world	This is a simple application created to show the basic configuration with S32DS
	hello_world_iar	This is a simple application created to show the basic configuration with IAR Embedded Workbench
	hello_world_mkf	This is a simple application created to show the basic configuration with makefile for the supported compilers
	lin_master	This demo application shows the usage of LIN stack in master mode.
	lin_slave	This demo application shows the usage of LIN stack in slave mode.
	Lwip	Shows the usage of lwIP stack.
	sCST	Demo application created to demonstrate sCST integration with S32 SDK

6. Supported hardware and compatible software

6.1 CPUs

- S32K116_32 revision 1.0, maskset 0N96V
- S32K116_48 revision 1.0, maskset 0N96V
- S32K118_48 revision 1.0, maskset 0N97V
- S32K118_64 revision 1.0, maskset 0N97V
- S32K142_48 revision 1.0, maskset 0N33V
- S32K142_64 revision 1.0, maskset 0N33V
- S32K142_100 revision 1.0, maskset 0N33V
- S32K144_48 revision 2.1, maskset 0N57U
- S32K144_64 revision 2.1, maskset 0N57U
- S32K144_100_LQFP revision 2.1, maskset 0N57U
- S32K144_100_BGA revision 2.1, maskset 0N57U
- S32K146_64 revision 1.0, maskset 0N73V
- S32K146_100_LQFP revision 1.0, maskset 0N73V
- S32K146_100_BGA revision 1.0, maskset 0N73V
- S32K146_144 revision 1.0, maskset 0N73V
- S32K148_100_LQFP revision 1.0, maskset 0N20V
- S32K148_100_BGA revision 1.0, maskset 0N20V
- S32K148_144 revision 1.0, maskset 0N20V
- S32K148_176 revision 1.0, maskset 0N20V
- S32K144W_64 revision 1.0, maskset 0P64A
- S32K144W_48 revision 1.0, maskset 0P64A
- S32K142W_64 revision 1.0, maskset 0P64A
- S32K142W_48 revision 1.0, maskset 0P64A

The following processor reference manual has been used to add support:

- S32K1XXRM Rev. 13.0, 04/2020.

The following errata documents were taken into consideration:

- S32K142 0N33V errata: S32K142_0N33V Rev.20/APR/2020
- S32K144 0N57 errata: S32K144_0N57 Rev.20/APR/2020
- S32K146 0N73V errata: S32K146_0N73V Rev.20/APR/2020
- S32K148 0N20V errata: S32K148_0N20V Rev.20/APR/2020
- S32K116 0N96V errata: S32K116_0N96V Rev.20/APR/2020
- S32K118 0N97V errata: S32K118_0N97V Rev.20/APR/2020
- S32K144W 0P64A errata: S32K144W_0P64A Rev.14 AUG 2020

6.2 Boards

- S32K-MB with mini module S32K144-100LQFP REV X1/X2
- S32K-MB with mini module S32K14xCVD-Q144 REV X2/X3
- S32K-MB with mini module S32K14xCVD-Q100 REV A
- S32K-MB with mini module S32K1xxCVD-Q048 REV X1
- S32K-MB with mini module S32K1xxCVD-Q064 REV X2

- S32K116EVB-Q048 PCB 30003 RevX2 SCH RevB
- S32K118EVB-Q064 PCB 29945 RevX2 SCH RevB
- S32K142EVB-Q100 29701 PCB RevX1 SCH RevD
- S32K144EVB-Q100 29248 PCB RevA SCH RevC
- XS32K146EVB-Q144 PCB 29844 RevX1 SCH RevB
- S32K148EVB-Q176 29644 PCB RevX1 SCH RevB
- XS32K14WEVB-Q064 PCB 46873 RevA SCH RevB

6.3 Compiler and IDE versions

- GreenHills compiler v. 2017.1.4
- IAR compiler v. 8.11.2
- GCC compiler for ARM v. 6.3.1 20170509
- Wind River Diab Compiler v5.9.6.2
- ARM Compiler 6.6.1 Long Term Maintenance
- S32 Design Studio 3.4

7. Known issues and limitations

7.1 Standalone installation

- The installer will automatically append the new SDK path to the S32SDK_PATH variable. Please make sure that only the desired value is kept, if the variable is used by previous projects.
- Custom installation type is not fully supported, keep “All Packages” selection in Choose Components page.
- Code generated by S32 Configuration Tool may contain MISRA violations.

7.2 Hardware support

- Support for S32K14xW was validated on S32K144W with 0P64A maskset.

7.3 Migration tool

Component	Devices	Known issues and limitations	Workaround solution
General	S32K11x, S32K14x	The project with SDK RTM 3.0.x works properly in term of functionality but its processorExpert file might inherit from SDK RTM 2.0.x and was not updated with SDK RTM 3.0.x. It leads to user cannot migrate their project.	User needs to update the ProcessorExpert.pe by opening the project with DS 2018.R1 or DS 2.2 and click to generate code button. Then import again project to DS 3.4 and perform migrating. If the project still can’t be migrated then it might cause from clock manager component, so user needs to remove this component from his project then add and configure it again. After that execute migration procedures.
General	S32K11x, S32K14x	The “Include” folder which was added by the user will be renamed to “include.bak” and excluded from build path after executing migration tool. It leads to migrated project cannot compile.	After migrating project do steps as following. **Step 01:** Renamed folder “include.bak” **to** “include”. **Step 02:** Right click to “include” folder -> Build Path -> Add to. Then select the provided value that matches to your project (Debug_Ram, Debug_Flash…).
General	S32K11x, S32K14x	SDK RTM 3.0.x releases allowed to declare variables in header files and call those variables in certain sources file but it is not allowed with SDK RTM 4.0.x releases. It makes a migrated project cannot compile	User should declare their variables in sources files and extern in headers file, respectively. Eg: For edma_transfer example of SDK after migrating there are two steps as following should be

		with error message: "multiple definition".	done: **Step 01:** Add two declarations to edmaTransfer.c source file: *edma_scatter_gather_list_t srcList[SG_TCD_COUNT];* *edma_scatter_gather_list_t dstList[SG_TCD_COUNT];* **Step 02:** Extern for these variables in edmaTransfer.h header file: *extern edma_scatter_gather_list_t srcList[SG_TCD_COUNT];* *extern edma_scatter_gather_list_t dstList[SG_TCD_COUNT];*
General	S32K11x, S32K14x	The migrated project might encounter the errors or warnings in “Problems” tab even though it was informed that project compiles successfully in “Console” tab. It happened with all SDK RTM 3.0.x except SDK RTM 3.0.3	User can right click into Project -> Index -> Freshen All Files.
General	S32K11x, S32K14x	During executing migration procedures, the default linker file from latest SDK version will be added automatically to migrated project. So, if the project uses custom linker file, then migrated project might not compile or run properly.	User needs to manually update their linker files after executing migration tool.
General	S32K11x, S32K14x	ProcessorExpert supports generating multiple configurations for every SDK’s component meanwhile S32CT supports only one. It happens with following SDK components: Adc_pal, Can_pal, Enet, Quadspi, Flexcan. When the user executes migration procedures, only the first configuration will be migrated.	User needs manually update the left configurations in his application.

SBC_FS45, SBC_UJA113x	S32K11x, S32K14x	Migration procedures does not support SBC_FS45 and SBC_UJA113x components because S32CT for these ones are not available in latest SDK version.	Not available.
tcpip	S32K148	Migrated project cannot be compiled successfully. Because tcpip S32CT component does not generate folder structure similar to tcpip PEx component (no more demo/test.c and platform.c/h files)	User needs to manually copy application source files (e.g. test.c, platform.c/h for clock and pin initialization) from the PEx project to the new migrated project and update the new migrated main.c file to reflect new changes.
phy_tja1101	S32K148	PHY driver was changed from SDK RTM 3.0.x releases to SDK RTM 4.0.x releases as following leads to migrated project cannot compile. - Changed API name. - Updated sleep, wakeup behavior. - Managed mode support. - Extended configuration support.	User needs to update his application following 4 points as mentioned. For phy_tja1101 example of SDK , user need to manually updated as following: **Step 01**: User needs to update his application by replacing PHY_RMR() to PHY_RMW(). **Step 02**: The example is about showing local wakeup via WAKE_IN_OUT pin on TJA device. User should change the wakeup configuration by manually updating these lines: 1. "PHY_RMR(0, PHY_CONFIG1, PHY_CONFIG1_FWDREM, PHY_CONFIG1_FWDREM)" **to** "PHY_RMR(0, PHY_CONFIG1, 0x0400U, 0x0400U)" 2. "if (extCtrl == 0xD000U)" **to** "if ((extCtrl & 0x7800) == 0x5000U)"
qspi	S32K148	ProcessorExpert did not support generating QuadSPI state structure (qspi_state_t) into configuration files meanwhile S32CT supported it. So, if user declares this variable in his application, It might make the migrated project cannot compile due to duplicated declaration.	User needs to remove QuadSPI state structure was declared in his application and re-compile the migrated project. Eg: For qspi_external_flash example of SDK user needs to remove declaration qspi_state_t qspiState; in main.c source file.

rtc	S32K11x, S32K14x	ProcessorExpert does not support to generate extern APIs of callBack functions into header file meanwhile S33CT support it. So migrated project cannot compile due to no-alignment of callBack's APIs between User's code and APIs which generated by S32CT.	User has to update prototype of callback functions in his application to align with what was generated by S32CT. For example, with rtc_alarm example of SDK user has to update as following in main.c source file: - void secondsISR(void) **to** void secondsISR(void * callbackParam) - void alarmISR(void) **to** void alarmISR(void * callbackParam)
freertos	S32K11x, S32K14x	User setting field does not support comment type: “/* */”. It leads to migrated project cannot compile.	User needs to manually remove this type of comment from project which was created with ProcessorExpert before migrating.
pins	S32K11x, S32K14x	There was an issue for ProcessorExpert that allows to configure more than 1 pin to be able to route to 01 signal meanwhile it does not happen in S32CT. It leads to migrated project might not run properly.	User should correct the pins after migration on S32CT or before migration on ProcessorExpert.
SBC_UJA116x	S32K142, S32K144	Driver code was changed from SDK RTM 3.0.x releases to SDK RTM 4.0.x as following leads to migrated project cannot compile. -Renamed driver from SBC_UJA1169 to SBC_UJA116x - Function SBC_Init was divided two functions: SBC_InitDriver and SBC_InitDevice	User needs to update these changes in his application.
i2s_pal, flexio_i2s	S32K11x, S32K14x	Migration procedures will encounter an error "FLEXIO used in multiple component" if project was created with ProcessorExpert and added multiple i2s_pal/flexio_i2s instances. The reason of this issue is ProcessorExpert supports adding multiple i2s_pal instances but S32CT just supports only one.	User can split his project to 2 separate project each one can use one i2s_pal/ flexio_i2s instance only. User can refer example from SDK RTM 4.0.x release, we separate 02 project: i2s_pal_master and i2s_pal_slave.

sai	S32K148	There was an issue for ProcessorExpert that the names of “state” structure variables (sai_state_t) will be duplicated if multiple configurations are added in user’s project. It leads to migrated project get warning about redefine when compiling and it might not run properly.	User needs to manually correct the names of state structure variables which are using in his application.
ftm_pwm	S32K116	Fault channel configuration cannot be migrated correctly for FTM instance 0 because: -An issue from ProcessorExpert that allows user to configure 04 faults. It was corrected to support only one fault in S32CT. -Migrated project will take first channel configuration from old project, but the correct channel configuration is channel 02. It leads to migrated project does not run properly.	User should update fault channel configuration manually.
csec	S32K118, S32K14x	Migrated project does not run properly because there were some changes might relate to S32 Design Studio version impact to the content of executable file. It happens will all SDK RTM 3.0.x releases except SDK RTM 3.0.3.	CSEc driver uses the content of executable file to validate the secure boot process. The BOOT_MAC value will be changed whenever the executable content changes, and the difference could be observed by disassembling executable file. User should update boot size (number of bit) value after migrating (the address corresponding to changed content * 8). For example, with csec_boot_protection example of SDK, the changed address could be observed at 0x234D, choose 0x3000 to contain the changed content => u32BootSize = 0x3000 * 8 = 0x18000U (number of bit). Therefore, the value of u32BootSize (in main.c file) should be update to 0x18000U instead of 0x3000U.

7.4 Drivers

CPU

- When using DIAB toolchain on S32K11x and the interrupt handlers are overwritten with INT_SYS_InstallHandler, the core will not return from interrupt handlers that are not calling other functions or writing a global variable. Workaround: Make sure that all interrupt handlers are performing at least one function call or are writing a global variable.
- ALIGNED macro is not supported for aligning function when using IAR Compiler.

CLOCK

- CLOCK_SYS_GetFreq function returns obsolete core clock frequency right after VLPR to HSRUN power mode transition because SCS bit field from SCG_CSR register is not immediately updated (workaround: function to be called twice, second call returns correct value).
- The default clock configuration always sets the SPLLDIVx_CLK, SIRCDIVx_CLK, FIRCDIVx_CLK, SOSCDIVx_CLK divider by 1 in RUN mode. This does not follow the recommendation in table 27-1 Clock description, RM rev 12.1 because the clock migration from projects created for S32K1xx with RTM 3.0.x to the S32K1xx with RTM 4.0.0 or higher release may have error. User can configure divider in each power mode if needed.

EDMA

- When using Single-block transfer or Multi-block transfer, NBYTES (Number of bytes to be transferred in each service request of the channel) shall be always configurable on 30 bits instead of 32 bits.

EIM

- An attempt to invert more than 2 bits in check bit mask or data mask might result in undefined behavior. To avoid this situation, you should invert a maximum of two bits.

FlexIO, SAI

- FlexIO drivers and the SAI driver cannot be simultaneously used in DMA mode due to overlapping DMA requests.

FlexIO_I2C

- No STOP condition is generated when aborting a transfer due to NACK reception.
- No clock stretching when the application does not provide data fast enough, so Tx underflows and Rx overflows are possible.
- The driver does not support multi-master mode. It does not detect arbitration loss condition.
- Due to device limitations, it is not always possible to tell the difference between NACK reception and receiver overflow.
- FlexIO_I2C may not create the N-ACK after receiving last data byte. This case only occurs when another interrupt takes over flexIO_I2C interrupt.

Note: FLEXIO I2C issues described above are caused by Hardware limitations.

FlexIO_SPI

- The driver does not support back-to-back transmission mode for CPHA = 1

FTM

- Each FTM instance can only be used in either FTM_MC or FTM_OC or FTM_QD mode. However, only FTM_PWM and FTM_IC mode can be used together, at the same time on one instance.
  - For example, this configuration is **not** possible: FTM0 configured with 4 channels for OC mode and 4 channels for QD mode.
  - FTM_PWM, FTM_IC: this configuration is possible: FTM0 configured with 4 channels for PWM mode and 4 channels for IC mode.
- S32CT: Adding FTM_PWM and FTM_IC configurations to use the same instance is not possible on S32CT, even if the driver support this.
- Complementary channel is not enabled in all configurations for independent channels. The workaround is to use complementary channel only for combined channels.

LIN

- LIN driver may not meet the requirements for nesting level due to an issue in tools.

LPSPI

- When a SPI transfer in slave mode over DMA is initialized with an invalid address for the TX buffer, the driver can never finish the transfer.

I2C_PAL, LPI2C

- When (LPI2C|I2C)_MasterAbortTransfer is called after a transfer operation was started and the address was not sent, the bus may hang. Workaround is to avoid calling the function shortly after a transfer was initiated.

LPI2C

- LPI2C_DRV_MasterAbortTransferData function can't abort a master receive transfer because the module sees the whole receive as a single operation and will not stop it even if the FIFO is reset.
- Minimum allowed value for HIGH period of the SCL is sometimes violated in standard mode.

RTC

- When using LPO clock as input, the user may need to use LPO trimming to obtain the 32kHz frequency needed by RTC module.

SBC

- All SBC components share the same SPI configuration. For each SBC component the SPI controller and configuration index shall be manually selected.

Timing_PAL

- For TIMING_PAL over FTM, the TIMING_GetElapsed and TIMING_GetRemaining functions return invalid value if the FTM interrupt occurs between 'Get current counter value' and 'Get channel start value' within these functions.
- The TIMING_GetElapsed and TIMING_GetRemaining do not detect elapsed or remaining time if the period is expired in case timer channel type is one-shot.

MPU

- On S32K14xx, if there is a PFlash access protection error caused by CM4, both slave port 0 & slave port 2 will report the same error.

SAI

- Tx Data fifo is not cleared in some case after user calls SAI_DRV_AbortSending function because of residue data remaining in fifo.

LIN stack

- S32 configuration tool only support LDF file with absolute path. When moving workspace/project to other PC (portability), customers need to re-select the correct LDF file path in their machine manually.

7.5 Examples

- Running the FLASH driver example from the flash will secure the device. To unsecure the MCU a mass erase of the flash needs to be done.
- Hello World example S32K146, S32K116, S32K118, S32K142W and S32K144W cannot be supported on IAR IDE.
- After partitioning Flash for CSEc operation, using the JLink Flash configuration of any other project will not work anymore.
  Workaround:
  - Run csec_keyconfig example with ERASE_ALL_KEYS 0, using PEmicro debug configuration
  - Run csec_keyconfig example with ERASE_ALL_KEYS 1, using PEmicro debug configuration

- Example projects for IAR Embedded Workbench use simulator as default debugger. The user has to manually select and configure the debug probe prior to downloading to the target.
- csec_keyconfig, csec_boot_protection, security_pal will no longer support j-link debugger configuration.

7.6 Libraries

- AMMCLIB: Missing the 16 and 32 bit fixed point pre-processor macros in the header files on version 1.1.21. The 16 and 32 bit fixed point always are enabled in the UI and it does not impact the code size, since the library is configured to allow the linker to discard unused code.

8. Compiler options

The example projects are using the first level of optimizations (low optimizations).

For exceptions from the following compiler settings, additional information can be found in the SDK documentation, Build Tools section.

8.1 IAR Compiler/Linker/Assembler Options

Table 8.1 IAR Compiler Options

Option	Description
-Ol	Low optimizations
-e	Allow IAR extensions
--cpu=Cortex-M4 / --cpu Cortex-M0+	Selects target processor: Arm Cortex M4 / Arm Cortex M0+
--thumb	Selects generating code that executes in Thumb state.
--fpu VFPv4_sp / --fpu none	Use floating point instructions / Use software floating point
--debug	Include debug information
-D<cpu_define>	Define a preprocessor symbol for MCU
-warnings_are_errors	Treat code warnings as errors

Table 8.2 IAR Assembler Options

Option	Description
--cpu Cortex-M4 / --cpu Cortex-M0+	Selects target processor: Arm Cortex M4 / Arm Cortex M0+
--thumb	Selects generating code that executes in Thumb state.
--fpu VFPv4_sp / --fpu none	Use floating point instructions / Use software floating point
-DSTART_FROM_FLASH	Mandatory when flash target is used

Table 8.3 IAR Linker Options

Option	Description
--cpu Cortex-M4 / --cpu Cortex-M0+	Selects target processor: Arm Cortex M4 / Arm Cortex M0+
--thumb	Selects generating code that executes in Thumb state.
--fpu VFPv4_sp / --fpu none	Use floating point instructions / Use software floating point
--map <map_file>	Produce a linker memory map file
--entry Reset_Handler	Make the symbol Reset_Handler be treated as a root symbol and the start label of the

	application
--config <linker_file.icf>	Use the specified linker file

8.2 GCC Compiler/Linker/Assembler Options

Table 8.3 GCC Compiler Options

Option	Description
-mcpu=cortex-m4 / -mcpu=cortex-m0plus	Selects target processor: Arm Cortex M4 / Arm Cortex M0+
-mthumb	Selects generating code that executes in Thumb state.
-O1	Optimize
-funsigned-char	Let the type char be unsigned, like unsigned char
-funsigned-bitfields	Bit-fields are signed by default
-fshort-enums	Allocate to an enum type only as many bytes as it needs for the declared range of possible values.
-ffunction-sections	Place each function into its own section in the output file
-fdata-sections	Place data item into its own section in the output file
-fno-jump-tables	Do not use jump tables for switch statements
-std=c99	Use C99 standard
-g	Generate debug information
-D<cpu_define>	Define a preprocessor symbol for MCU
-mfloat-abi=hard / -mfloat-abi=soft	Use FPU instructions / Use software FP
-mfpu=fpv4-sp-d16	Specify the FPU variant (only for S32K14x)
-Wall	Produce warnings about questionable constructs
-Wextra	Produce extra warnings that -Wall
-Wstrict-prototypes	Warn if a function is declared or defined without specifying the argument types.
-pedantic	Issue all the warnings demanded by strict ISO C
-Wunused	Produce warnings for unused variables
-Werror	Treat warnings as errors
-Wsign-compare	Produce warnings when comparing signed type with unsigned type

Table 8.4 GCC Assembler Options

Option	Description
-mcpu=cortex-m4 / -mcpu=cortex-m0plus	Selects target processor: Arm Cortex M4 / Arm Cortex M0+
-mthumb	Selects generating code that executes in Thumb state.

-mfloat-abi=hard / -mfloat-abi=soft	Use FPU instructions / Use software FP
-mfpu=fpv4-sp-d16	Specify the FPU variant (only for S32K14x)
-Wall	Produce warnings about questionable constructs
-Wextra	Produce extra warnings that -Wall
-Wstrict-prototypes	Warn if a function is declared or defined without specifying the argument types.
-pedantic	Issue all the warnings demanded by strict ISO C
-Werror	Treat warnings as errors
-x assembler-with-cpp	Preprocess assembly files
-DSTART_FROM_FLASH	Mandatory when flash target is used

Table 8.6 GCC Linker Options

Option	Description
-mcpu=cortex-m4 / -mcpu=cortex-m0plus	Selects target processor: Arm Cortex M4 / Arm Cortex M0+
-mthumb	Selects generating code that executes in Thumb state.
--entry=Reset_Handler	Make the symbol Reset_Handler be treated as a root symbol and the start label of the application
-T<linker_file.ld>	Use the specified linker file
-mfloat-abi=hard / -mfloat-abi=soft	Use FPU instructions / Use software FP
-mfpu=fpv4-sp-d16	Specify the FPU variant (only for S32K14x)
-Xlinker –gc-sections	Remove unused sections
-WI, -Map=<map_file>	Produce a map file
-lgcc	Link libgcc
-lc	Link C library
-lm	Link Math library

8.3 GHS Compiler/Linker/Assembler Options

Table 8.5 GHS Compiler Options

Option	Description
-cpu=cortexm0plus / -cpu=cortexm4	Selects target processor: Arm Cortex M4 / Arm Cortex M0+
-thumb	Selects generating code that executes in Thumb state.
-fhard / -fsoft	Use FPU instructions / Use software FP
-fpu=vfpv4_d16	Specify FPU type (only for S32K14x)
-c99	Use C99 standard
--gnu_asm	Enables GNU extended asm syntax support
-Ogeneral	Optimize
-gdwarf-2	Generate DWARF 2.0 debug information

-G	Generate debug information
-D<cpu_define>	Define a preprocessor symbol for MCU
--quit_after_warnings	Treat warnings as errors
-Wimplicit-int	Produce warnings if functions are assumed to return int
-Wshadow	Produce warnings if variables are shadowed
-Wtrigraphs	Produce warnings if trigraphs are detected
-Wundef	Produce a warning if undefined identifiers are used in #if preprocessor statements

Table 8.8 GHS Assembler Options

Option	Description
-cpu=cortexm0plus / -cpu=cortexm4	Selects target processor: Arm Cortex M4 / Arm Cortex M0+
-fhard / -fsoft	Use FPU instructions / Use software FP
-fpu=vfpv4_d16	Specify FPU type (only for S32K14x)
-preprocess_assembly_files	Preprocess assembly files
-DSTART_FROM_FLASH	Mandatory when flash target is used

Table 8.9 GHS Linker Options

Option	Description
-cpu=cortexm0plus / -cpu=cortexm4	Selects target processor: Arm Cortex M4 / Arm Cortex M0+
-thumb	Selects generating code that executes in Thumb state.
-entry=Reset_Handler	Make the symbol Reset_Handler be treated as a root symbol and the start label of the application
-T<linker_file.ld>	Use the specified linker file
-map=<map_file>	Produce a map file
-larch	Link architecture specific library

8.4 DIAB Compiler/Linker/Assembler Options

Table 8.10 DIAB Compiler Options

Option	Description
-tARMCORTEXM4LV / -tARMCORTEXM0PLS	Selects target processor: Arm Cortex M4 / Arm Cortex M0+
-mthumb	Selects generating code that executes in Thumb state.
-Xdialect-c99	Use C99 standard
-D<cpu_define>	Define a preprocessor symbol for MCU
-g	Add debug information to the executable

-O	Optimize
-Xstop-on-warning	Treat warnings as errors
-ei5388,5387,1824	Ignore some specific warnings

Table 8.11 DIAB Assembler Options

Option	Description
-tARMCORTEXM4LV / -tARMCORTEXM0PLS	Selects target processor: Arm Cortex M4 / Arm Cortex M0+
-mthumb	Selects generating code that executes in Thumb state.
-Xpreprocess-assembly	Preprocess assembly files
-DSTART_FROM_FLASH	Mandatory when flash target is used

Table 8.12 DIAB Linker Options

Option	Description
-tARMCORTEXM4LV / -tARMCORTEXM0PLS	Selects target processor: Arm Cortex M4 / Arm Cortex M0+
-Xremove-unused-sections	Removes unused code sections
-lc	Link the standard C library to the project in order to support elementary operations that are used by the drivers
-lm	Link the standard math library to the project in order to support elementary math operations that are used by the drivers
<linker_file.dld>	Use the specified linker file
-e Reset_Handler	Make the symbol Reset_Handler be treated as a root symbol and the start label of the application
-m6 > <map_file>	Produce a linker map
-Xpreprocess-lecl	Perform pre-processing on linker scripts

8.5 ARMC Compiler/Linker/Assembler Options

Table 8.13 ARMC Compiler Options

Option	Description
--target=arm-arm-none-eabi	Select arm-none-eabi as target architecture
--cpu=cortex-m4 / --cpu=cortex-m0plus	Selects target processor: Arm Cortex M4 / Arm Cortex M0+
-mthumb	Selects generating code that executes in Thumb state.
-O1	Optimize
-fshort-enums	Allocate to an enum type only as many bytes as it needs for the declared range of possible values.

-fdata-sections	Place data item into its own section in the output file
-std=c99	Use C99 standard
-g	Generate debug information
-D<cpu_define>	Define a preprocessor symbol for MCU
-mfloat-abi=hard / -mfloat-abi=soft	Use FPU instructions / Use software FP
-pedantic	Issue all the warnings demanded by strict ISO C
-Weverything	Produce warnings for unused variables
-Werror	Treat warnings as errors
-Wno-switch-enum	Do not issue warnings for enum values that are not explicitly treated in switch statements
-Wno-cast-align	Do not issue warnings for cast statements that increase the required alignment
-Wno-cast-qual	Do not issue warnings for cast statements that are discarding const qualifier.
-Wno-covered-switch-default	Do not issue warnings for “default” switch case being present when all enum values are covered in a switch
-Wno-reserved-id-macro	Do not issue warnings when macros starting with double underscore (e.g. __IO) are present in the code.
-Wno-padded	Do not issue warnings when padding is added.

Table 8.14 ARMC Assembler Options

Option	Description
--target=arm-arm-none-eabi	Select arm-none-eabi as target architecture
--cpu=cortex-m4 / --cpu=cortex-m0plus	Selects target processor: Arm Cortex M4 / Arm Cortex M0+
-mfloat-abi=hard / -mfloat-abi=soft	Use FPU instructions / Use software FP
--cpreproc	Instructs the assembler to call armcc to preprocess the input file before assembling it
--cpreproc_opts	Enables the assembler to pass options to the compiler when using the C preprocessor
-DSTART_FROM_FLASH	Mandatory define when flash target is used

Table 8.15 ARMC Linker Options

Option	Description
--target=arm-arm-none-eabi	Select arm-none-eabi as target architecture
--cpu=cortex-m4 / --cpu=cortex-m0plus	Selects target processor: Arm Cortex M4 / Arm Cortex M0+
--entry Reset_Handler	Make the symbol Reset_Handler be treated as a root symbol and the start label of the application
--scatter “<scatter_file>”	Use the specified scatter file
--datacompressor off	Turn off compression for data sections
--map	Produce a map file

--list=<map_file>	Assign a file for the map
--symbols	Save the symbol information in the map file
--predefine="-mcpu=cortex-m0plus" or --predefine="-mcpu=cortex-m4"	Selects target processor for preprocessor

Note: The symbol <linker_file> must be replaced with the corresponding path and linker file name per device, memory model and target compiler.

E.g. *C:\NXP\S32_SDK\platform\devices\S32K144\linker\gcc\S32K144_64_flash.ld* - for S32K144, 64 KB of SRAM and Flash target on GCC.

Symbol <map_file> shall be replaced with the desired map file name.

Symbol <cpu_define> shall be replaced with CPU_S32K144HFT0VLLT for S32K144, CPU_S32K148 for S32K148, CPU_S32K142 for S32K142, CPU_S32K146 for S32K146, CPU_S32K144W for S32K144W and CPU_S32K142W for S32K142W

9. Acronyms

Acronym	Description
EAR	Early Access Release
JRE	Java Runtime Environment
EVB	Evaluation board
PAL	Peripheral Abstraction Layer
RTOS	Real Time Operating System
PEx	Processor Expert Configurator
PD	Peripheral Driver
RTM	Ready to Manufacture
S32DS	S32 Design Studio IDE
SDK	Software Development Kit
SOC	System-on-Chip
sCST	Structural Core Self Test
S32CT	S32 Configuration Tool

10. Version Tracking

Date (dd-Mmm-YYYY)	Version	Comments	Author
30-Oct-2015	1.0	First version for EAR 0.8.0	Vlad Baragan-Stroe
18-Dec-2015	1.1	Added patch 1	Vlad Baragan-Stroe
01-Apr-2016	2.0	Added drivers, new in release section, updated examples, known limitations for EAR 0.8.1	Vlad Baragan-Stroe
27-Oct-2016	3.0	Updated new in this release section, known limitations and examples description for EAR 0.8.2 release. Added “Compiler options” section. Updated header, footer and front page with new logos	Rares Vasile
21-Dec-2016	4.0	Updated Release Notes for 0.9.0 BETA release	Rares Vasile
23-Mar-2017	5.0	Updated Release Notes for 1.0.0 RTM release	Rares Vasile
04-May-2017	6.0	Updated Release Notes for 0.8.3 EAR release	Rares Vasile
10-May-2017	6.1	Updated Release Notes for 0.8.3 EAR release - Added drivers, new in release section, updated examples, known limitations for EAR 0.8.3	Cezar Dobromir
27-Jun-2017	7.0	Updated for EAR 0.8.4 release	Rares Vasile
31-Aug-2017	8.0	Updated for EAR 0.8.5 release	Rares Vasile
27-Nov-2017	9.0	Updated for EAR 0.8.6 release	Rares Vasile
3-May-2018	10.0	Updated for BETA 1.9.0 release	Rares Vasile
26-Jun-2018	11.0	Updated for RTM 2.0.0 release	Rares Vasile
21-Aug-2018	12.0	Updated for BETA 2.9.0 release	Rares Vasile
21-Nov-2018	13.0	Updated for BETA 2.9.2 release	Vlad Lionte
21-Feb-2019	14.0	Updated for RTM 3.0.0 release	Vlad Lionte
28-Mar-2019	14.1	Updated for RTM 3.0.1 service release	Vlad Lionte
11-Oct-2019	14.2	Updated for RTM 3.0.2 service release	Ovidiu-Marius Alexandru
5-May-2020	15.0	Updated for RTM 3.0.3 service release	Cuong Nguyen Van
12-June-2020	16.0	Updated for RTM 4.0.0 release	Cuong Nguyen Van

20-Oct-2020	17.0	Updated for RTM 4.0.1 release	Cuong Nguyen Van
11-June-2021	18.0	Updated for RTM 4.0.2 release	Tung Dang Thanh