S32 Configuration Tool 1.0 R Release Notes

Contents

1 Release Description

S32 Configuration Tool product is composed of a suite of tools for configuring NXP processors and generating initialization code. This is the first mature version of S32 Configuration Tool with support for the following tools:

- **Pins** Tool
- **Clocks** Tool
- **Peripherals** Tool
- **Device Configuration Data** Tool
- **Image Vector Table** Tool

The package provides support for the following processors:

- **S32S247TV**
- **S32V23x**

Development of the tools is done in sync with the following documentation's versions:

- S32S247TV - Header files based on Reference Manual Rev 1 Draft O
- S32V23x - Header files based on Reference Manual Rev 3

1.1 Installation

S32 Configuration Tool is integrated into S32 Design Studio - Version: S32DS 3.2 and is compatible with:

- S32 SDK for S32V23x 1.0.0 RTM release
- S32 SDK for S32S2 0.8.0 EAR release

One can see S32 SDK and S32 Design Studio release announcements for more details about these products including their location.

S32 Configuration Tool was verified on:

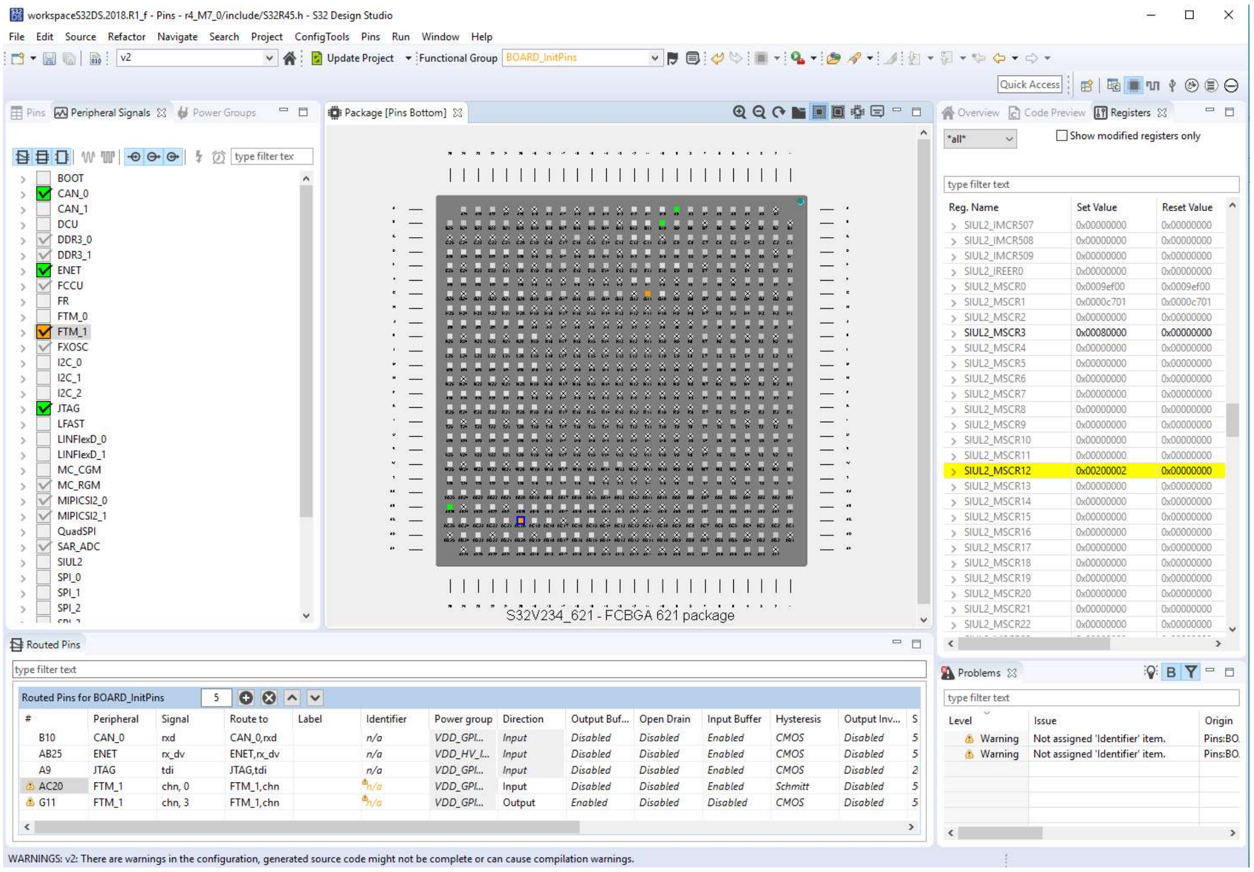
- Windows 7/10, 64bit
- Linux - Ubuntu 16/17, 64bit and CentOS 7, 64bit

2 What's New

2.1 Pins Tool

Pins Tool displays, inspects, modifies any aspect of pins configuration and muxing of the device. Pins routing can be done in multiple views:

- Pins view
- Peripheral Signals view
- Package view
- Routed Pins



Registers: *all*

Reg. Name	Set Value	Reset Value
SIUL2_IMCR507	0x00000000	0x00000000
SIUL2_IMCR508	0x00000000	0x00000000
SIUL2_IMCR509	0x00000000	0x00000000
SIUL2_IREER0	0x00000000	0x00000000
SIUL2_MSCR0	0x0009ef00	0x0009ef00
SIUL2_MSCR1	0x0000c701	0x0000c701
SIUL2_MSCR2	0x00000000	0x00000000
SIUL2_MSCR3	0x00080000	0x00000000
SIUL2_MSCR4	0x00000000	0x00000000
SIUL2_MSCR5	0x00000000	0x00000000
SIUL2_MSCR6	0x00000000	0x00000000
SIUL2_MSCR7	0x00000000	0x00000000
SIUL2_MSCR8	0x00000000	0x00000000
SIUL2_MSCR9	0x00000000	0x00000000
SIUL2_MSCR10	0x00000000	0x00000000
SIUL2_MSCR11	0x00000000	0x00000000
SIUL2_MSCR12	0x00200002	0x00000000
SIUL2_MSCR13	0x00000000	0x00000000
SIUL2_MSCR14	0x00000000	0x00000000
SIUL2_MSCR15	0x00000000	0x00000000
SIUL2_MSCR16	0x00000000	0x00000000
SIUL2_MSCR17	0x00000000	0x00000000
SIUL2_MSCR18	0x00000000	0x00000000
SIUL2_MSCR19	0x00000000	0x00000000
SIUL2_MSCR20	0x00000000	0x00000000
SIUL2_MSCR21	0x00000000	0x00000000
SIUL2_MSCR22	0x00000000	0x00000000

Routed Pins

Routed Pins for BOARD_InitPins 5

#	Peripheral	Signal	Route to	Label	Identifier	Power group	Direction	Output Buf...	Open Drain	Input Buffer	Hysteresis	Output Inv...	S
B10	CAN_0	rxd	CAN_0,rxd		n/a	VDD_GPI...	Input	Disabled	Disabled	Enabled	CMOS	Disabled	5
AB25	ENET	rx_dv	ENET,rx_dv		n/a	VDD_HV_I...	Input	Disabled	Disabled	Enabled	CMOS	Disabled	5
A9	JTAG	tdi	JTAG,tdi		n/a	VDD_GPI...	Input	Disabled	Disabled	Enabled	CMOS	Disabled	2
⚠ AC20	FTM_1	chn, 0	FTM_1,chn		⚠n/a	VDD_GPI...	Input	Disabled	Disabled	Enabled	Schmitt	Disabled	5
⚠ G11	FTM_1	chn, 3	FTM_1,chn		⚠n/a	VDD_GPI...	Output	Enabled	Disabled	Disabled	CMOS	Disabled	5

Problems

Level	Issue	Origin
⚠ Warning	Not assigned 'Identifier' item.	Pins:BO
⚠ Warning	Not assigned 'Identifier' item.	Pins:BO

WARNINGS: v2: There are warnings in the configuration, generated source code might not be complete or can cause compilation warnings.

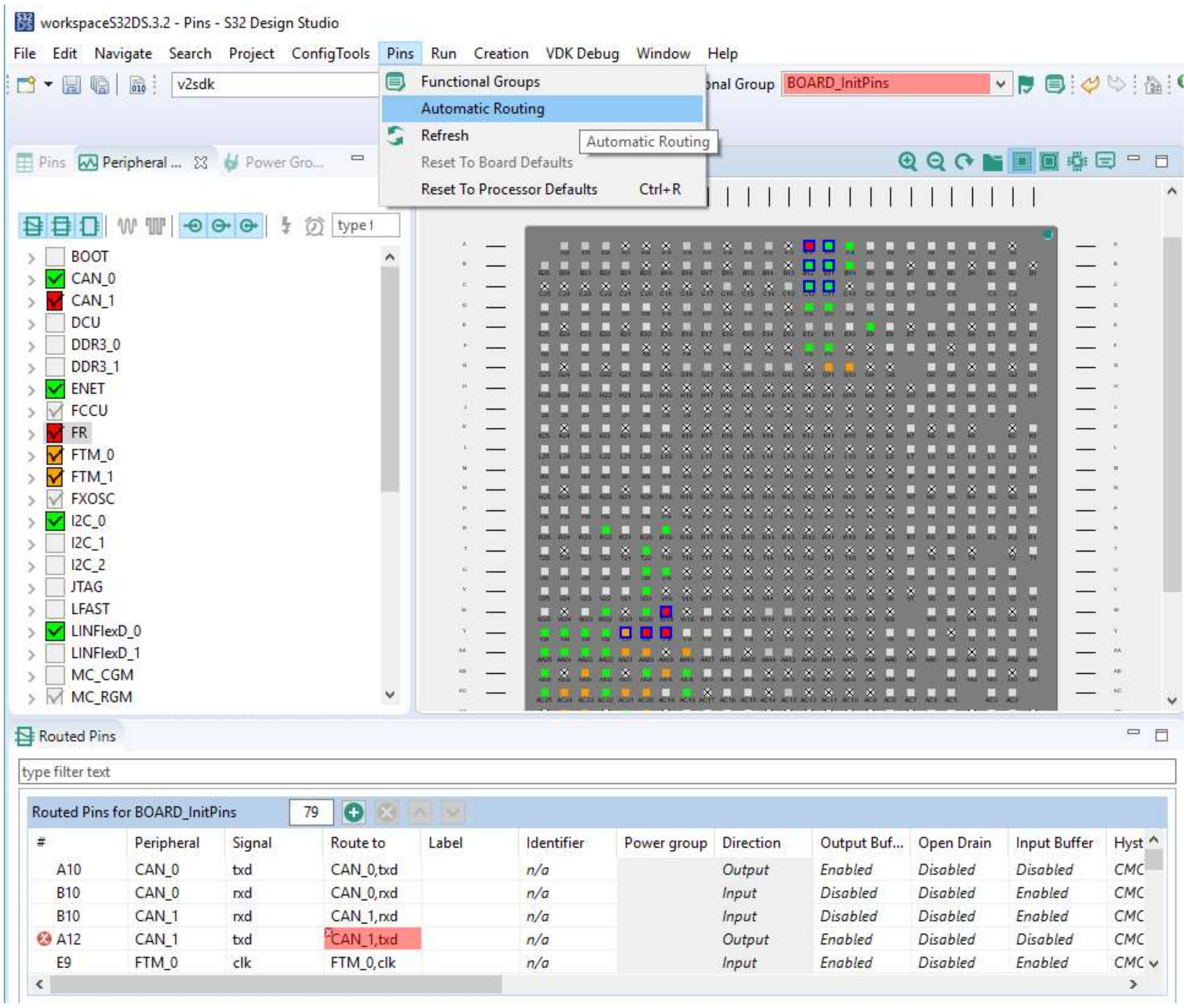
Figure 1: Pins Tool

Pins Tool is used for pin routing configuration, validation and code generation, including pin functional/electrical properties, run-time configurations, with the following main features:

- Muxing and pin configuration with consistency checking
- Graphical processor package view
- Multiple configuration blocks/functions
- Easy-to-use device configuration
- Selection of Pins and Peripherals

- Package with IP blocks
- Routed pins with electrical characteristics
- Registers with configured and reset values
- Source code for C/C++ applications
- Automatic pins routing

Automatic routing tries to resolve conflicts, it routes signals on different pins until a valid configuration is reached. The used algorithm is a mixed approach between running maximum matching algorithm in a signals – pins bipartite graph and processing remaining signals with backtracking.

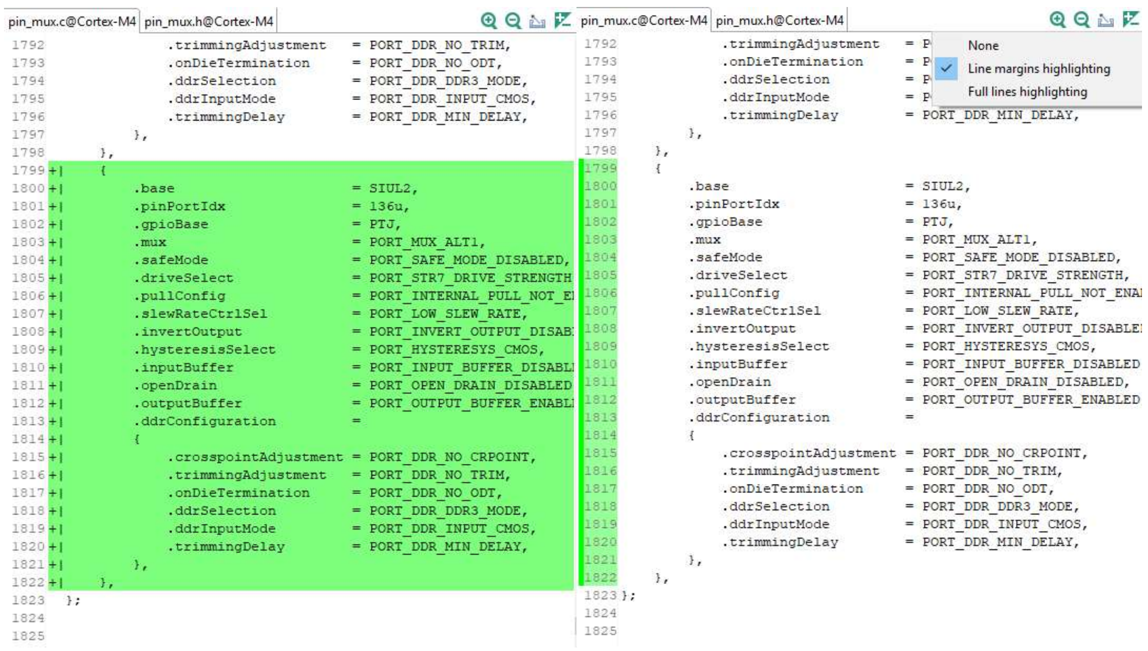


#	Peripheral	Signal	Route to	Label	Identifier	Power group	Direction	Output Buf...	Open Drain	Input Buffer	Hyst
A10	CAN_0	txd	CAN_0,txd		n/a		Output	Enabled	Disabled	Disabled	CMC
B10	CAN_0	rxd	CAN_0,rxd		n/a		Input	Disabled	Disabled	Enabled	CMC
B10	CAN_1	rxd	CAN_1,rxd		n/a		Input	Disabled	Disabled	Enabled	CMC
A12	CAN_1	txd	CAN_1,txd		n/a		Output	Enabled	Disabled	Disabled	CMC
E9	FTM_0	clk	FTM_0,clk		n/a		Input	Enabled	Disabled	Enabled	CMC

Figure 2: Pins Tool - Automatic routing

Enabled code changes highlighting for all tools. Two styles are currently supported:

- Margin highlighting
- Full line highlighting



None
[✓ Line margins highlighting]
Full lines highlighting

```
                .trimmingAdjustment    = PORT_DDR_NO_TRIM,
                .onDieTermination      = PORT_DDR_NO_ODT,
                .ddrSelection          = PORT_DDR_DDR3_MODE,
                .ddrInputMode          = PORT_DDR_INPUT_CMOS,
                .trimmingDelay         = PORT_DDR_MIN_DELAY,
            },
        },
        {
            .base                      = SIUL2,
            .pinPortIdx                = 136u,
            .gpioBase                  = PTJ,
            .mux                       = PORT_MUX_ALT1,
            .safeMode                  = PORT_SAFE_MODE_DISABLED,
            .driveSelect               = PORT_STR7_DRIVE_STRENGTH
            .pullConfig                = PORT_INTERNAL_PULL_NOT_EI
            .slewRateCtrlSel           = PORT_LOW_SLEW_RATE,
            .invertOutput              = PORT_INVERT_OUTPUT_DISAB
            .hysteresisSelect          = PORT_HYSTERESYS_CMOS,
            .inputBuffer               = PORT_INPUT_BUFFER_DISABL
            .openDrain                 = PORT_OPEN_DRAIN_DISABLED
            .outputBuffer              = PORT_OUTPUT_BUFFER_ENABL
            .ddrConfiguration          =
            {
                .crosspointAdjustment = PORT_DDR_NO_CRPOINT,
                .trimmingAdjustment    = PORT_DDR_NO_TRIM,
                .onDieTermination      = PORT_DDR_NO_ODT,
                .ddrSelection          = PORT_DDR_DDR3_MODE,
                .ddrInputMode          = PORT_DDR_INPUT_CMOS,
                .trimmingDelay         = PORT_DDR_MIN_DELAY,
            },
        },
    };
```

```
                .trimmingAdjustment    = P
                .onDieTermination      = P
                .ddrSelection          = P
                .ddrInputMode          = P
                .trimmingDelay         = PORT_DDR_MIN_DELAY,
            },
        },
        {
            .base                      = SIUL2,
            .pinPortIdx                = 136u,
            .gpioBase                  = PTJ,
            .mux                       = PORT_MUX_ALT1,
            .safeMode                  = PORT_SAFE_MODE_DISABLED,
            .driveSelect               = PORT_STR7_DRIVE_STRENGTH,
            .pullConfig                = PORT_INTERNAL_PULL_NOT_ENA
            .slewRateCtrlSel           = PORT_LOW_SLEW_RATE,
            .invertOutput              = PORT_INVERT_OUTPUT_DISABLE
            .hysteresisSelect          = PORT_HYSTERESYS_CMOS,
            .inputBuffer               = PORT_INPUT_BUFFER_DISABLED
            .openDrain                 = PORT_OPEN_DRAIN_DISABLED,
            .outputBuffer              = PORT_OUTPUT_BUFFER_ENABLED
            .ddrConfiguration          =
            {
                .crosspointAdjustment = PORT_DDR_NO_CRPOINT,
                .trimmingAdjustment    = PORT_DDR_NO_TRIM,
                .onDieTermination      = PORT_DDR_NO_ODT,
                .ddrSelection          = PORT_DDR_DDR3_MODE,
                .ddrInputMode          = PORT_DDR_INPUT_CMOS,
                .trimmingDelay         = PORT_DDR_MIN_DELAY,
            },
        },
};
```

Figure 3: Pins Tool – Code changing highlighting

For **S32V23x,** the FCBGA 621 package is supported with pins configuration available for the following peripherals:

- BOOT
- CAN 0 - 1
- DCU
- DDR3_0
- DDR3_1
- ENET
- FCCU
- FR
- FTM 0 -1
- FXOSC
- I2C 0 - 2
- JTAG
- LFAST
- LINFlexD 0 - 1
- MC_CGM
- MC_RGM
- MIPICSI2 0 - 1
- QuadSPI
- SAR_ADC
- SIUL2
- SPI 0 - 3
- SSE

- SoC
- TRACE
- VIULite 0 - 1
- WKPU
- uSDHC

For **S32S247TV,** the BGA501 package is supported with pins configuration available for the following peripherals:

- BOOT
- CAN 0 - 7
- CTU AE 1 – 2
- FCCU
- FR 0 – 1
- FTM 0 – 1
- FlexPWM AE 1 – 2
- GMAC_0
- I2C 0 – 4
- JTAG
- LINFlexD 0 – 5
- Misc
- PSI5, PSI5 S0, PSI5 S1
- PowerAndGround
- QuadSPI
- SAR_ADC_AE_0_0, SAR_ADC_AE_1_0, SAR_ADC_AE_1_1, SAR_ADC_AE_2_0, SAR_ADC_AE_2_1
- SIUL2 0, SIUL2 1, SIUL2 AE
- SPI_0 – 5
- SRX_0 – 1
- SoC
- eMIOS_0 – 1
- eTImer_AE_1_0, eTimer_AE_1_1, eTimer_2_0, eTimer_AE_2_1
- uSDHC

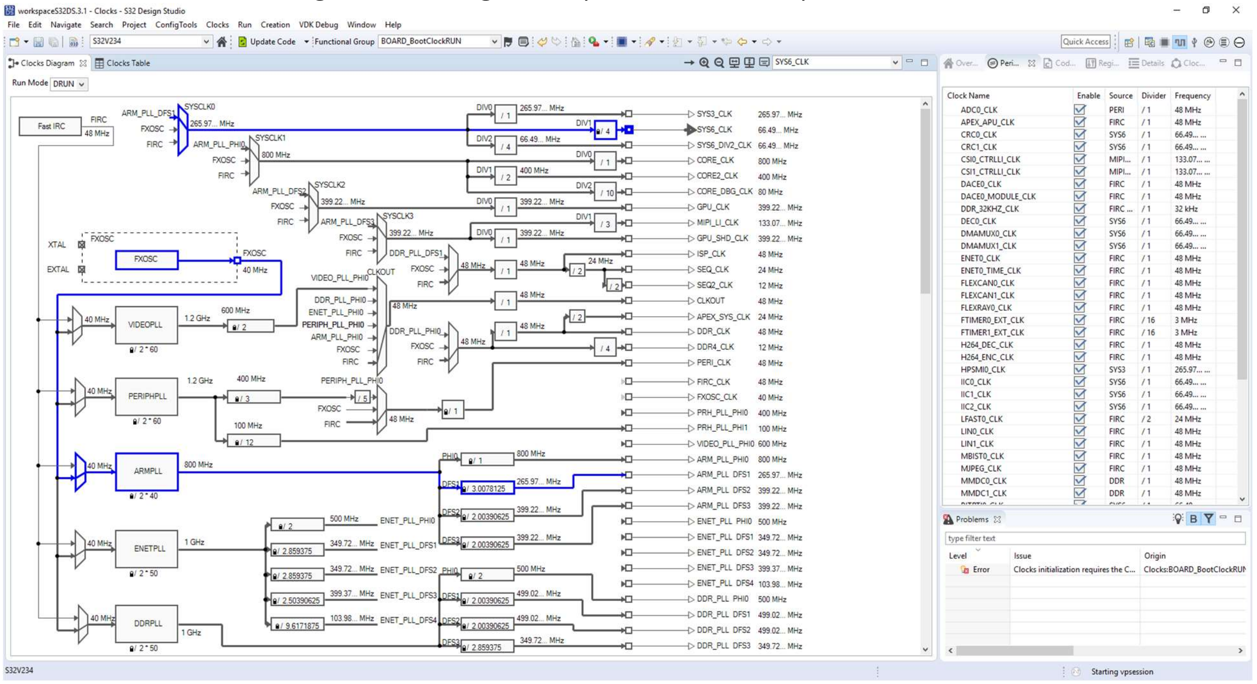
2.2 Clocks Tool

Clocks tool allows users to easily configure initialization of the system clock - core, system, bus, peripheral clocks - and generate C code with clock initialization functions and configuration structures. The tool validates clock elements and calculates resulting output and clock frequencies.

Advanced settings can be done in Diagram and Details views. Other capabilities are listed next:

- Inspect and modify configuration of elements on clock path from clock source up to the core/peripherals
- Validate clock elements settings and calculate the resulting output clock frequencies
- Generate a configuration code aligned with latest version of header files
- Table view of clock elements with their parameters allowing the user to modify the settings and see the outputs
- Diagram view allows easy navigation and displays important settings and frequencies

- Peripheral Clock view which contains the list of peripherals - if a chain in form of selector, divider and gate is found, an entry is created
- Find clock elements settings that fulfills given requirements for outputs


workspaceS32DS.3.1 - Clocks - S32 Design Studio
File Edit Navigate Search Project ConfigTools Clocks Run Creation VDK Debug Window Help
S32V234
Update Code
Functional Group BOARD_BootClockRUN
Quick Access
Clocks Diagram
Clocks Table
SYS6_CLK
Run Mode DRUN
Over...
Peri...
Cod...
Regi...
Details
Cloc...
S32V234

Clock Name	Enable	Source	Divider	Frequency
ADC0_CLK	✓	PERI	/ 1	48 MHz
APEX_APU_CLK	✓	FIRC	/ 1	48 MHz
CRC0_CLK	✓	SYS6	/ 1	66.49... ...
CRC1_CLK	✓	SYS6	/ 1	66.49... ...
CSI0_CTRLLI_CLK	✓	MIPI...	/ 1	133.07... ...
CSI1_CTRLLI_CLK	✓	MIPI...	/ 1	133.07... ...
DACE0_CLK	✓	FIRC	/ 1	48 MHz
DACE0_MODULE_CLK	✓	FIRC	/ 1	48 MHz
DDR_32KHZ_CLK	✓	FIRC ...	/ 1	32 kHz
DEC0_CLK	✓	SYS6	/ 1	66.49... ...
DMAMUX0_CLK	✓	SYS6	/ 1	66.49... ...
DMAMUX1_CLK	✓	SYS6	/ 1	66.49... ...
ENET0_CLK	✓	FIRC	/ 1	48 MHz
ENET0_TIME_CLK	✓	FIRC	/ 1	48 MHz
FLEXCAN0_CLK	✓	FIRC	/ 1	48 MHz
FLEXCAN1_CLK	✓	FIRC	/ 1	48 MHz
FLEXRAY0_CLK	✓	FIRC	/ 1	48 MHz
FTIMER0_EXT_CLK	✓	FIRC	/ 16	3 MHz
FTIMER1_EXT_CLK	✓	FIRC	/ 16	3 MHz
H264_DEC_CLK	✓	FIRC	/ 1	48 MHz
H264_ENC_CLK	✓	FIRC	/ 1	48 MHz
HPSMI0_CLK	✓	SYS3	/ 1	265.97... ...
IIC0_CLK	✓	SYS6	/ 1	66.49... ...
IIC1_CLK	✓	SYS6	/ 1	66.49... ...
IIC2_CLK	✓	SYS6	/ 1	66.49... ...
LFAST0_CLK	✓	FIRC	/ 2	24 MHz
LIN0_CLK	✓	FIRC	/ 1	48 MHz
LIN1_CLK	✓	FIRC	/ 1	48 MHz
MBIST0_CLK	✓	FIRC	/ 1	48 MHz
MJPEG_CLK	✓	FIRC	/ 1	48 MHz
MMDC0_CLK	✓	DDR	/ 1	48 MHz
MMDC1_CLK	✓	DDR	/ 1	48 MHz

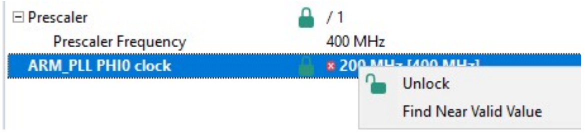
Problems

Level	Issue	Origin
Error	Clocks initialization requires the C...	Clocks:BOARD_BootClockRUN

Starting vpsession

Figure 4: Clocks Tool

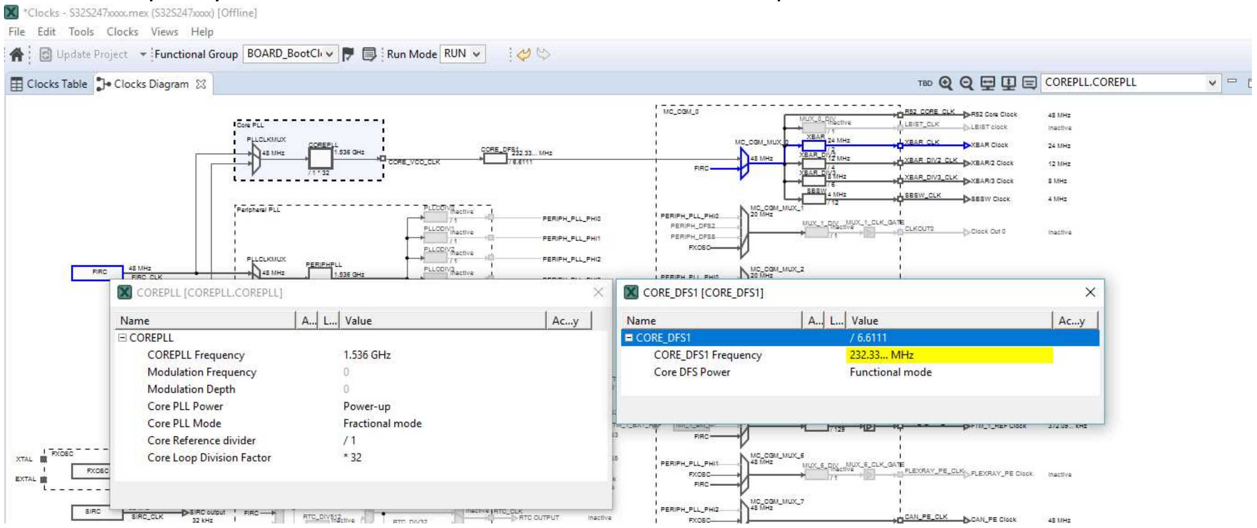
- Find near value for clock frequencies



⊟ Prescaler		/ 1
Prescaler Frequency		400 MHz
ARM_PLL PHI0 clock		200 MHz [400 MHz]

Figure 5: Clocks Tool – Find near value

- Reverse frequency calculation onto fractional elements in clocks path


*Clocks - S32S247xxxx.mex (S32S247xxxx) [Offline]
File Edit Tools Clocks Views Help
Update Project
Functional Group BOARD_BootCl
Run Mode RUN
Clocks Table
Clocks Diagram
COREPLL.COREPLL

COREPLL [COREPLL.COREPLL]

Name	A...	L...	Value	Ac...y
⊟ COREPLL				
COREPLL Frequency			1.536 GHz	
Modulation Frequency			0	
Modulation Depth			0	
Core PLL Power			Power-up	
Core PLL Mode			Fractional mode	
Core Reference divider			/ 1	
Core Loop Division Factor			* 32	

CORE_DFS1 [CORE_DFS1]

Name	A...	L...	Value	Ac...y
⊟ CORE_DFS1			/ 6.6111	
CORE_DFS1 Frequency			232.33... MHz	
Core DFS Power			Functional mode	

Figure 6: Clocks Tool – Fractional elements

- Allow automatic update of enable state for clock elements based on power modes
- Re-order tabs in Clock perspective, Clocks Diagram and Peripherals Clock views have focus
- Easy to find elements in clock tree, use Find hotkey to search for elements

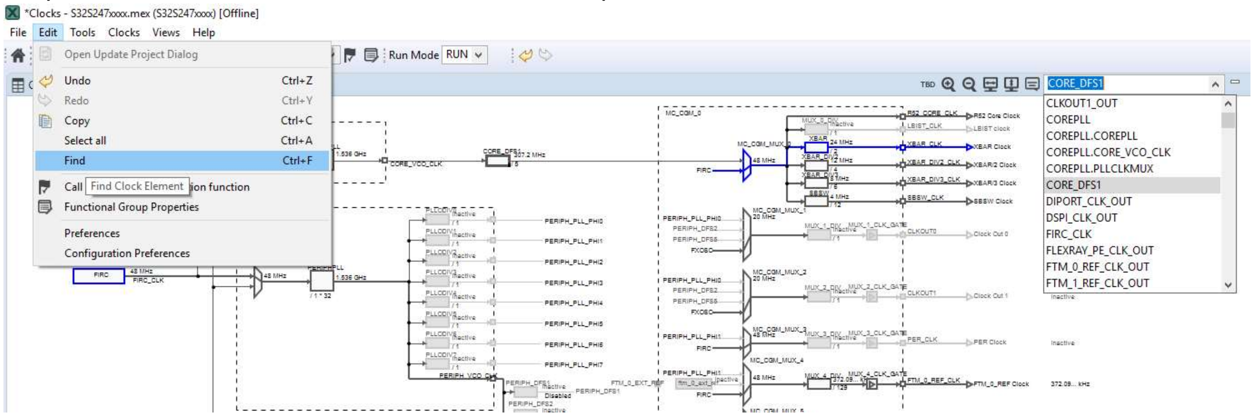


Figure 7: Clocks Tool – find elements

- Export Clock diagram

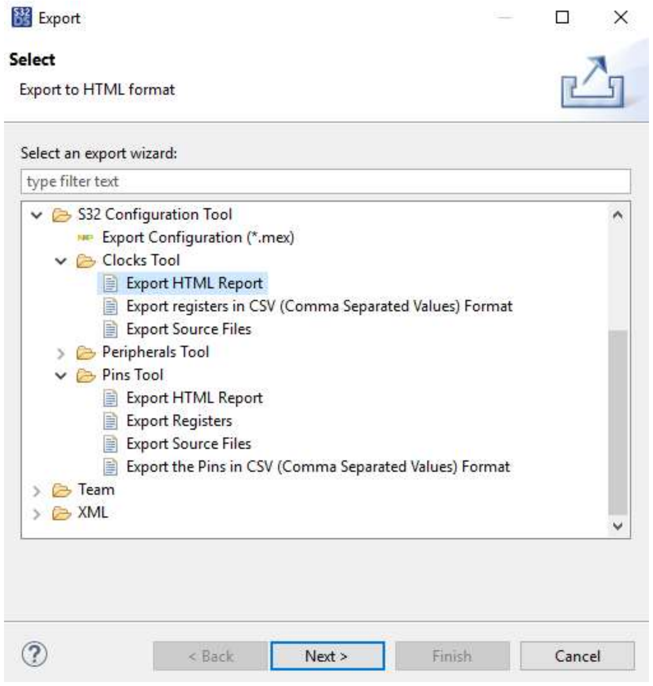


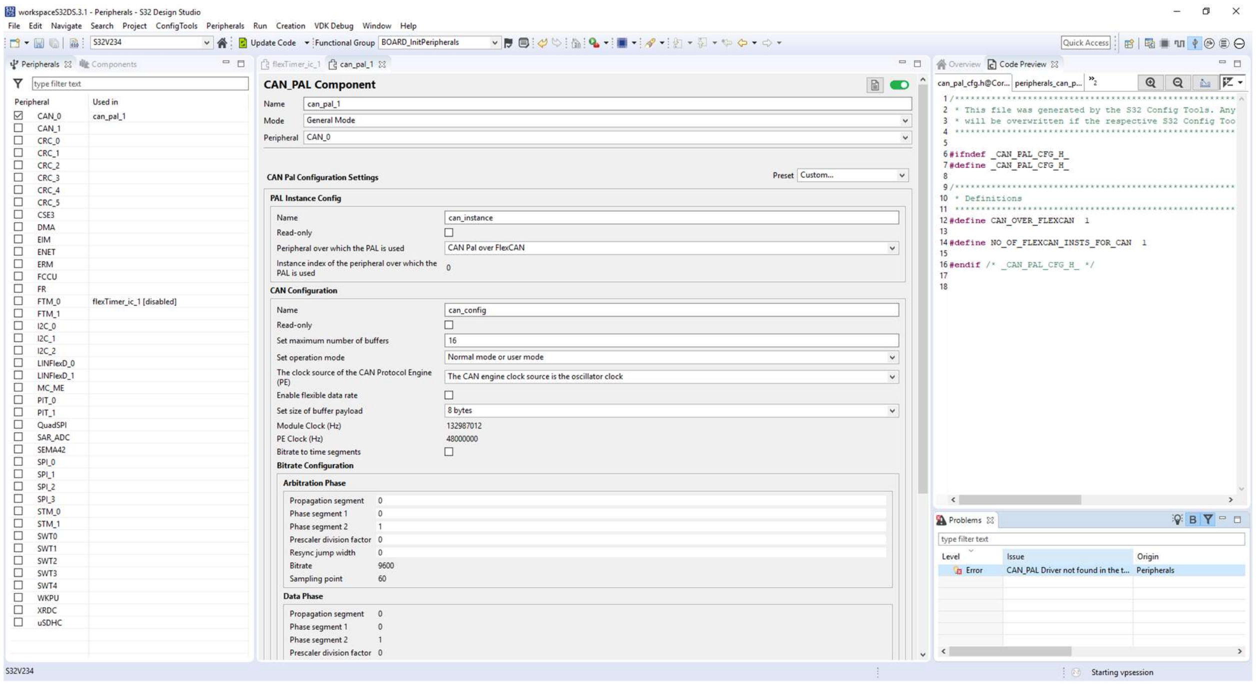
Figure 8: Clocks Tool – Export diagram

- Add support for zoom on cursor in Clocks Diagram

2.3 Peripherals Tool

Peripherals Tool offers support to initialize, configure peripherals and generate code for S32 SDK drivers. Two S32 SDK are supported:

- S32 SDK for S32V23x 1.0.0 RTM release
- S32 SDK for S32S2 0.8.0 EAR release



Peripheral	Used in
☑ CAN_0	can_pal_1
☐ CAN_1	
☐ CRC_0	
☐ CRC_1	
☐ CRC_2	
☐ CRC_3	
☐ CRC_4	
☐ CRC_5	
☐ CSE3	
☐ DMA	
☐ EIM	
☐ ENET	
☐ ERM	
☐ FCCU	
☐ FR	
☐ FTM_0	flexTimer_ic_1 [disabled]
☐ FTM_1	
☐ I2C_0	
☐ I2C_1	
☐ I2C_2	
☐ LINFlexD_0	
☐ LINFlexD_1	
☐ MC_ME	
☐ PIT_0	
☐ PIT_1	
☐ QuadSPI	
☐ SAR_ADC	
☐ SEMA42	
☐ SPI_0	
☐ SPI_1	
☐ SPI_2	
☐ SPI_3	
☐ STM_0	
☐ STM_1	
☐ SWT0	
☐ SWT1	
☐ SWT2	
☐ SWT3	
☐ SWT4	
☐ WKPU	
☐ XRDC	
☐ uSDHC	

CAN_PAL Component

Name: can_pal_1
Mode: General Mode
Peripheral: CAN_0

CAN Pal Configuration Settings Preset: Custom...

PAL Instance Config

Name	can_instance
Read-only	☐
Peripheral over which the PAL is used	CAN Pal over FlexCAN
Instance index of the peripheral over which the PAL is used	0

CAN Configuration

Name	can_config
Read-only	☐
Set maximum number of buffers	16
Set operation mode	Normal mode or user mode
The clock source of the CAN Protocol Engine (PE)	The CAN engine clock source is the oscillator clock
Enable flexible data rate	☐
Set size of buffer payload	8 bytes
Module Clock (Hz)	132987012
PE Clock (Hz)	48000000
Bitrate to time segments	☐

Bitrate Configuration

Arbitration Phase

Propagation segment	0
Phase segment 1	0
Phase segment 2	1
Prescaler division factor	0
Resync jump width	0
Bitrate	9600
Sampling point	60

Data Phase

Propagation segment	0
Phase segment 1	0
Phase segment 2	1
Prescaler division factor	0

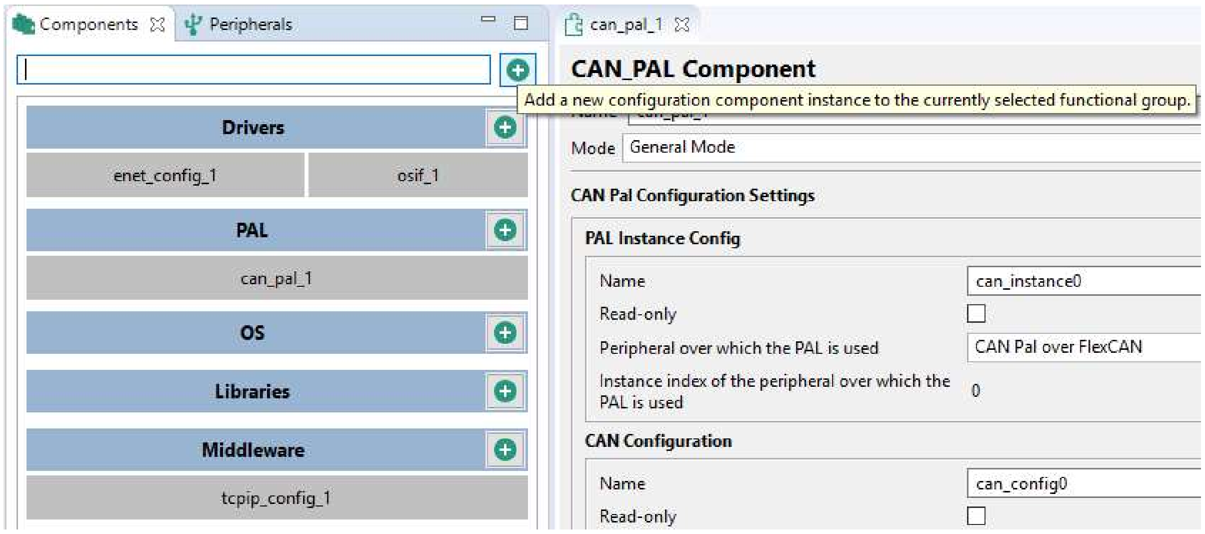
```
/*****************************************************************
 * This file was generated by the S32 Config Tools. Any
 * will be overwritten if the respective S32 Config Too
 *****************************************************************

#ifndef _CAN_PAL_CFG_H_
#define _CAN_PAL_CFG_H_

/*****************************************************************
 * Definitions
 *****************************************************************
#define CAN_OVER_FLEXCAN  1

#define NO_OF_FLEXCAN_INSTS_FOR_CAN  1

#endif /* _CAN_PAL_CFG_H_ */
```

Level	Issue	Origin
Error	CAN_PAL Driver not found in the t...	Peripherals

Figure 9: Peripherals Tool

Other capabilities are listed next:

- Configuration and initialization for SDK drivers
- User friendly user interface allowing to inspect and modify settings
- Smart configuration component selection along the SDK drivers used in toolchain project



CAN_PAL Component

Add a new configuration component instance to the currently selected functional group.

Mode: General Mode

CAN Pal Configuration Settings

PAL Instance Config

Name	can_instance0
Read-only	☐
Peripheral over which the PAL is used	CAN Pal over FlexCAN
Instance index of the peripheral over which the PAL is used	0

CAN Configuration

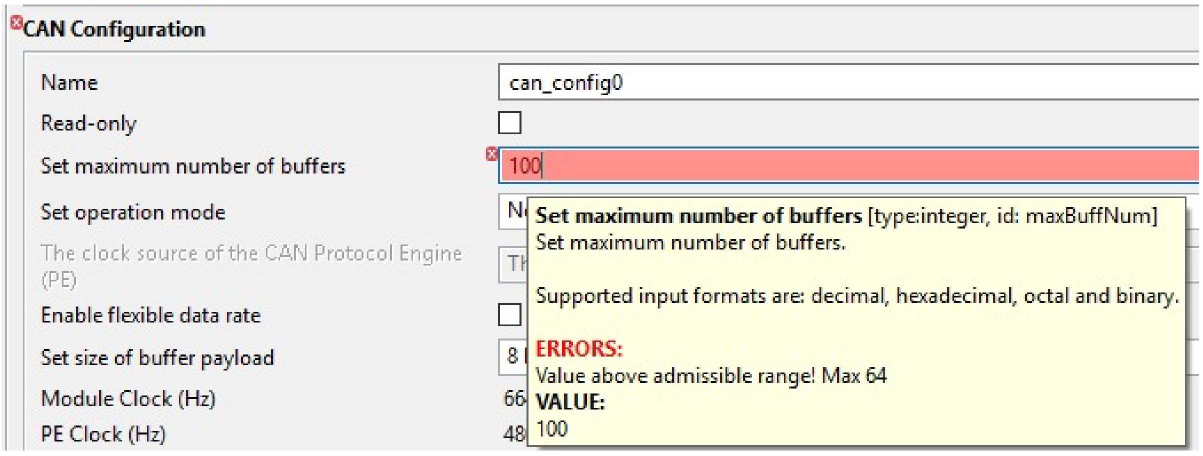
Name	can_config0
Read-only	☐

Figure 10: Peripherals Tool – SDK components

- Generation of initialization source code using SDK function calls
- Multiple function groups support for initialization alternatives

- Configuration problems are shown in Problems view and marked with decorators in other views

- Instant validation of basic constraints and problems in configuration


CAN Configuration

Name	can_config0
Read-only	☐
Set maximum number of buffers	100
Set operation mode	N
The clock source of the CAN Protocol Engine (PE)	Th
Enable flexible data rate	☐
Set size of buffer payload	8
Module Clock (Hz)	66
PE Clock (Hz)	48

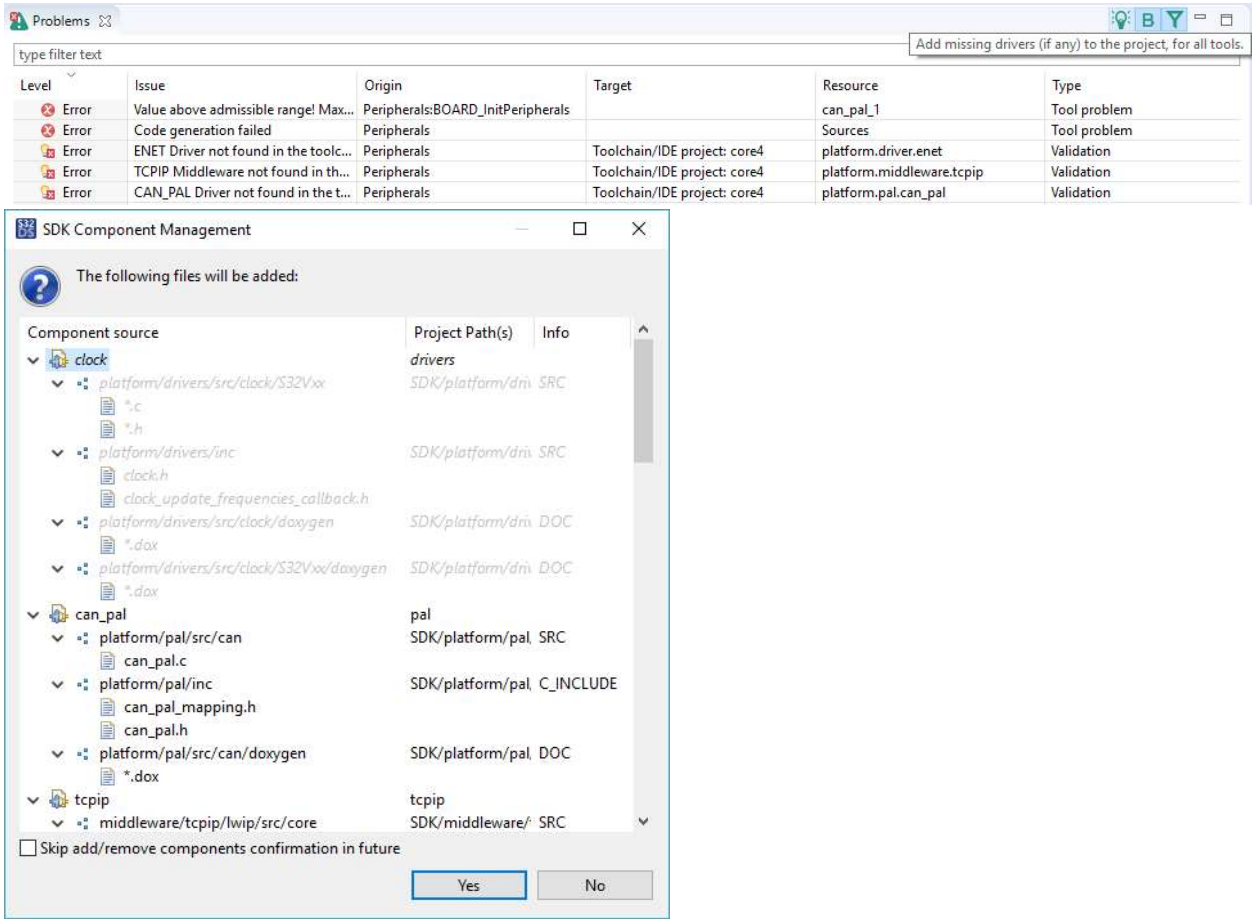
Set maximum number of buffers [type:integer, id: maxBuffNum]
Set maximum number of buffers.

Supported input formats are: decimal, hexadecimal, octal and binary.

ERRORS:
Value above admissible range! Max 64
VALUE:
100

Figure 11: Peripherals Tool – constraints

- Missing drivers are added to project with one click using dedicated option


Problems

Add missing drivers (if any) to the project, for all tools.

type filter text

Level	Issue	Origin	Target	Resource	Type
Error	Value above admissible range! Max...	Peripherals:BOARD_InitPeripherals		can_pal_1	Tool problem
Error	Code generation failed	Peripherals		Sources	Tool problem
Error	ENET Driver not found in the toolc...	Peripherals	Toolchain/IDE project: core4	platform.driver.enet	Validation
Error	TCPIP Middleware not found in th...	Peripherals	Toolchain/IDE project: core4	platform.middleware.tcpip	Validation
Error	CAN_PAL Driver not found in the t...	Peripherals	Toolchain/IDE project: core4	platform.pal.can_pal	Validation

SDK Component Management

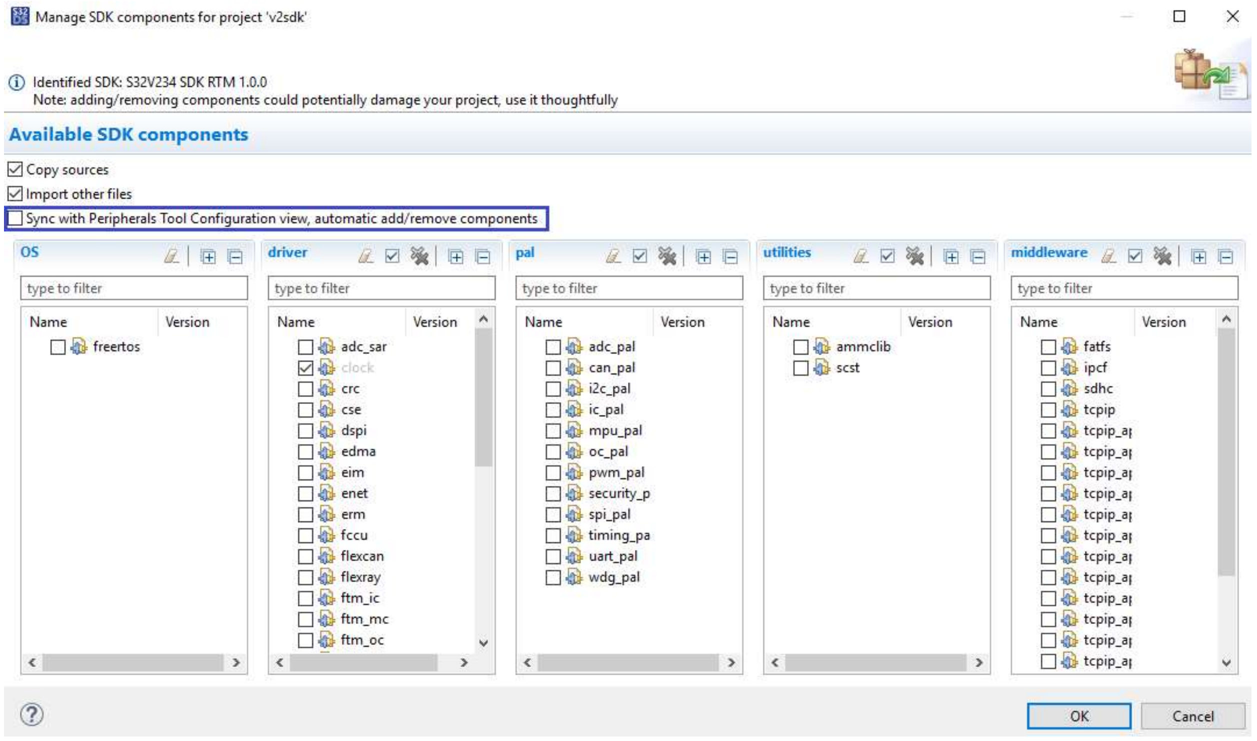
The following files will be added:

Component source	Project Path(s)	Info
clock	drivers	
platform/drivers/src/clock/S32Vxx	SDK/platform/dri	SRC
*.c		
*.h		
platform/drivers/inc	SDK/platform/dri	SRC
clock.h		
clock_update_frequencies_callback.h		
platform/drivers/src/clock/doxygen	SDK/platform/dri	DOC
*.dox		
platform/drivers/src/clock/S32Vxx/doxygen	SDK/platform/dri	DOC
*.dox		
can_pal	pal	
platform/pal/src/can	SDK/platform/pal	SRC
can_pal.c		
platform/pal/inc	SDK/platform/pal	C_INCLUDE
can_pal_mapping.h		
can_pal.h		
platform/pal/src/can/doxygen	SDK/platform/pal	DOC
*.dox		
tcpip	tcpip	
middleware/tcpip/lwip/src/core	SDK/middleware/	SRC

☐ Skip add/remove components confirmation in future

Figure 12: Peripherals Tool – SDK Component Management, add files

- Manage SDK Components in sync with Peripherals view



Manage SDK components for project 'v2sdk'

Identified SDK: S32V234 SDK RTM 1.0.0
Note: adding/removing components could potentially damage your project, use it thoughtfully

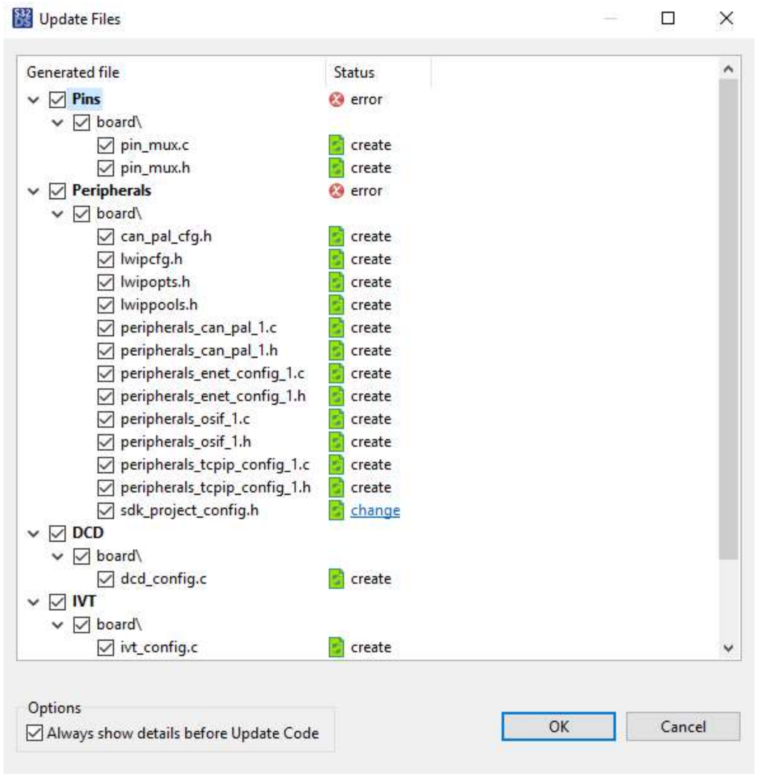
Available SDK components

☑ Copy sources
☑ Import other files
☐ Sync with Peripherals Tool Configuration view, automatic add/remove components

OS	driver	pal	utilities	middleware
☐ freertos	☐ adc_sar	☐ adc_pal	☐ ammclib	☐ fatfs
	☑ clock	☐ can_pal	☐ scst	☐ ipcf
	☐ crc	☐ i2c_pal		☐ sdhc
	☐ cse	☐ ic_pal		☐ tcpip
	☐ dspi	☐ mpu_pal		☐ tcpip_a
	☐ edma	☐ oc_pal		☐ tcpip_a
	☐ eim	☐ pwm_pal		☐ tcpip_a
	☐ enet	☐ security_p		☐ tcpip_a
	☐ erm	☐ spi_pal		☐ tcpip_a
	☐ fccu	☐ timing_pa		☐ tcpip_a
	☐ flexcan	☐ uart_pal		☐ tcpip_a
	☐ flexray	☐ wdg_pal		☐ tcpip_a
	☐ ftm_ic			☐ tcpip_a
	☐ ftm_mc			☐ tcpip_a
	☐ ftm_oc			☐ tcpip_a
				☐ tcpip_a

Figure 13: Peripherals Tool – SDK Component Management

- Move generated files to project for all tools



Update Files

Generated file	Status
☑ **Pins**	error
☑ board\	
☑ pin_mux.c	create
☑ pin_mux.h	create
☑ **Peripherals**	error
☑ board\	
☑ can_pal_cfg.h	create
☑ lwipcfg.h	create
☑ lwipopts.h	create
☑ lwippools.h	create
☑ peripherals_can_pal_1.c	create
☑ peripherals_can_pal_1.h	create
☑ peripherals_enet_config_1.c	create
☑ peripherals_enet_config_1.h	create
☑ peripherals_osif_1.c	create
☑ peripherals_osif_1.h	create
☑ peripherals_tcpip_config_1.c	create
☑ peripherals_tcpip_config_1.h	create
☑ sdk_project_config.h	change
☑ **DCD**	
☑ board\	
☑ dcd_config.c	create
☑ **IVT**	
☑ board\	
☑ ivt_config.c	create

Options
☑ Always show details before Update Code

Figure 14: Peripherals Tool – Update Code

- Help mechanism for peripheral components

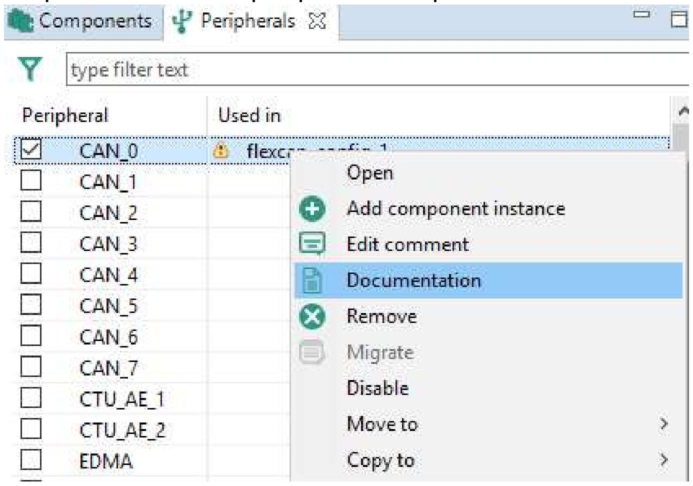


Figure 15: Peripherals Tool – Show documentation

2.4 Device Configuration Data Tool

Device Configuration Data (DCD) tool generates the DCD image using the format and constraints specified in the BootROM reference manual. BootROM reads and interprets the DCD image to configure various peripherals on the device. The location of the DCD is determined by the pointer in Image Vector Table (IVT) image.

DCD contains three types of commands:

- Write, write a list of bytes
- Check, test a list of bytes
- Nop, has no effect

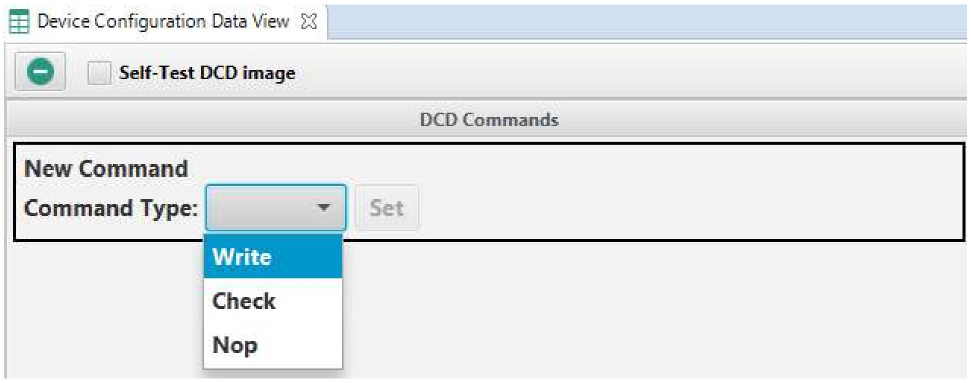
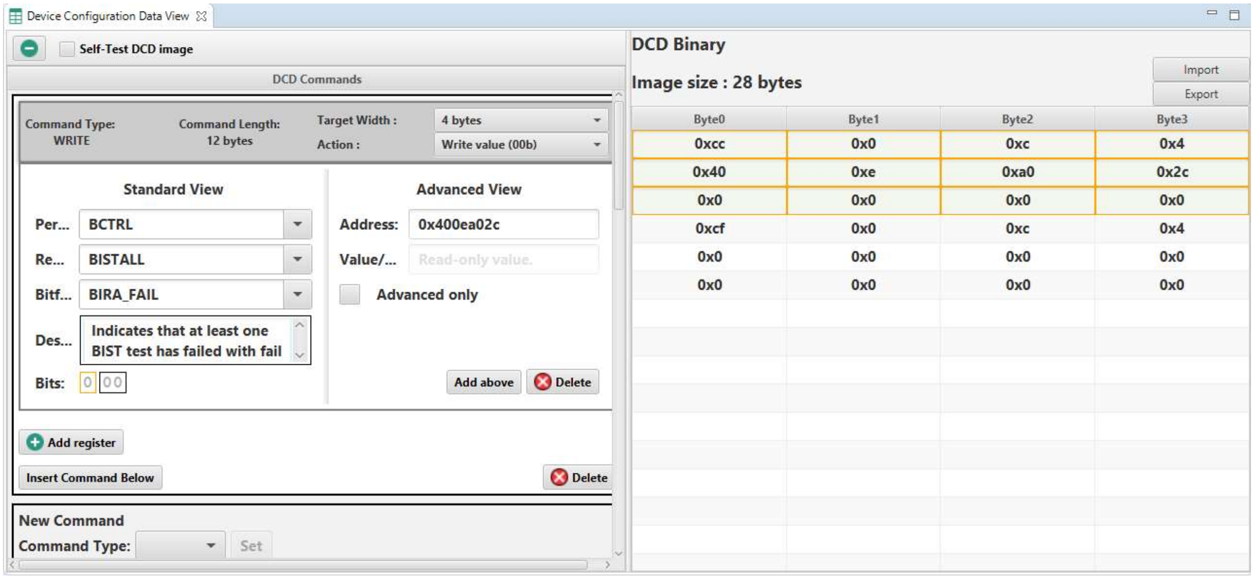


Figure 16: DCD Tool – Commands

There are two views:

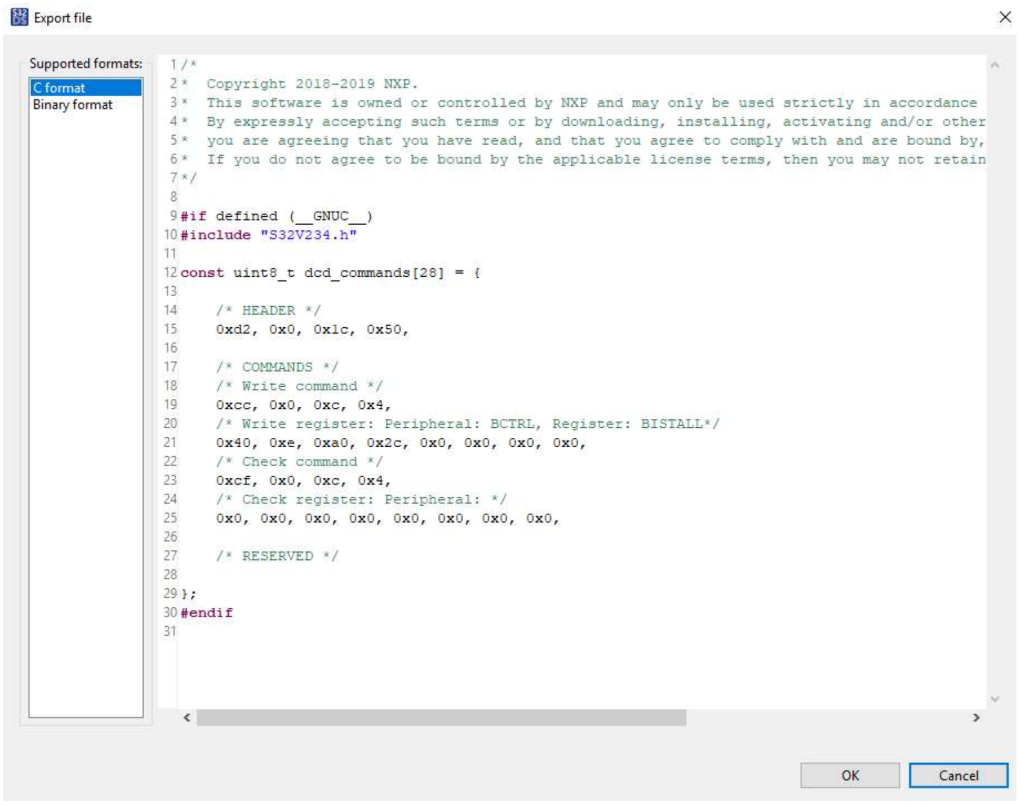
- Binary view displays DCD image in a binary format
- DCD Command view



Byte0	Byte1	Byte2	Byte3
0xcc	0x0	0xc	0x4
0x40	0xe	0xa0	0x2c
0x0	0x0	0x0	0x0
0xcf	0x0	0xc	0x4
0x0	0x0	0x0	0x0
0x0	0x0	0x0	0x0

Figure 17: DCD Tool – Views

Device configuration records can be saved in a variety of formats (C array, binary and proprietary .mex format):



```
/*
 *  Copyright 2018-2019 NXP.
 *  This software is owned or controlled by NXP and may only be used strictly in accordance
 *  By expressly accepting such terms or by downloading, installing, activating and/or other
 *  you are agreeing that you have read, and that you agree to comply with and are bound by,
 *  If you do not agree to be bound by the applicable license terms, then you may not retain
*/

#if defined (__GNUC__)
#include "S32V234.h"

const uint8_t dcd_commands[28] = {

    /* HEADER */
    0xd2, 0x0, 0x1c, 0x50,

    /* COMMANDS */
    /* Write command */
    0xcc, 0x0, 0xc, 0x4,
    /* Write register: Peripheral: BCTRL, Register: BISTALL*/
    0x40, 0xe, 0xa0, 0x2c, 0x0, 0x0, 0x0, 0x0,
    /* Check command */
    0xcf, 0x0, 0xc, 0x4,
    /* Check register: Peripheral: */
    0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0,

    /* RESERVED */

};
#endif
```

Figure 18: DCD Tool – Export image

Values can be reset to processor's defaults:

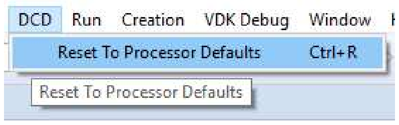


Figure 19: DCD Tool – Reset to Defaults

All reserved memory segments are set to 0xFF.

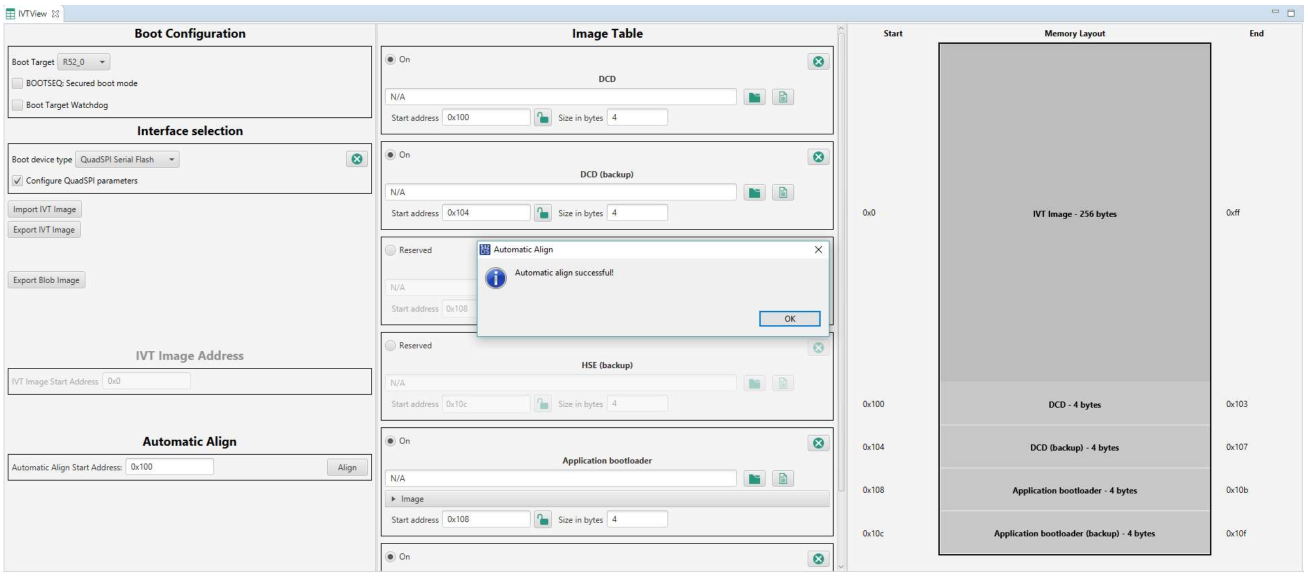
2.5 Image Vector Table Tool

Image Vector Table (IVT) tool configures and generates the IVT image, the first data structure that BootROM reads from boot device. The IVT contains the required data components like image entry point, pointer to DCD and other pointers used by the BootROM during boot process.

User can start the configuration using three views:

- Boot Configuration panel allows configuration of boot specific options
- Image Table view allows configuration of IVT pointers by selecting size and start address
- Memory Layout is a graphical representation of the memory to visualize the IVT segments

IVT tool segments can be automatically aligned to eliminate overlaps.



Boot Target: R52_0
BOOTSEQ: Secured boot mode
Boot Target Watchdog
Boot device type: QuadSPI Serial Flash
✓ Configure QuadSPI parameters
IVT Image Start Address: 0x0
Automatic Align Start Address: 0x100

Segment	State	File	Start address	Size in bytes
DCD	On	N/A	0x100	4
DCD (backup)	On	N/A	0x104	4
	Reserved	N/A	0x108	
HSE (backup)	Reserved	N/A	0x10c	4
Application bootloader	On	N/A	0x108	4

Automatic align successful!

Start	Memory Layout	End
0x0	IVT Image - 256 bytes	0xff
0x100	DCD - 4 bytes	0x103
0x104	DCD (backup) - 4 bytes	0x107
0x108	Application bootloader - 4 bytes	0x10b
0x10c	Application bootloader (backup) - 4 bytes	0x10f

Figure 21: IVT Tool – Align

Application bootloader image can be generated and a pre-validation mechanism is used.

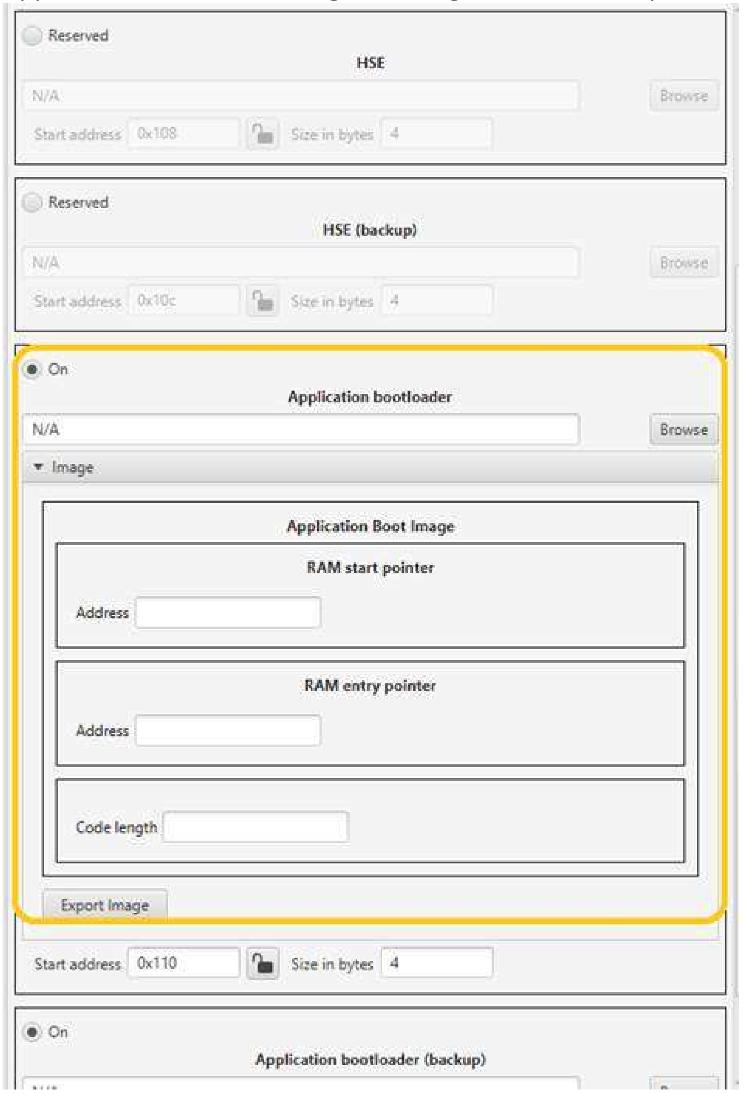


Figure 20: IVT Tool – Application Boot Image

Control mechanism for reserved sections and pointers will be set to 0xFFFFFFFF.

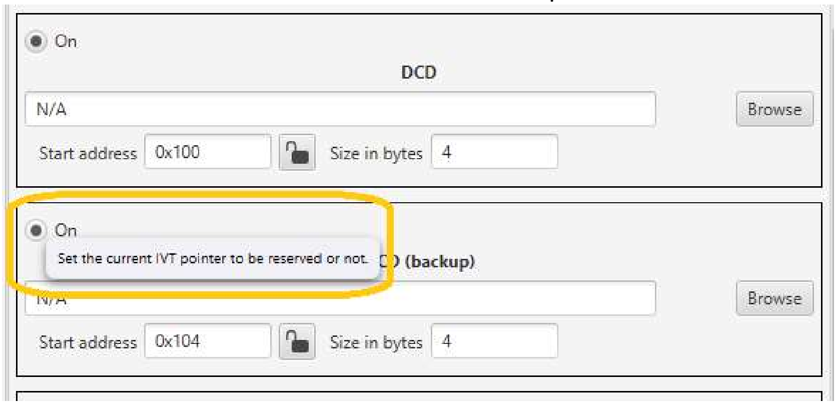


Figure 23: IVT Tool – Reserved pointers

Exporting all existing images as Blob file including IVT.bin, DCD.bin, HSE.bin, Application_bootloader.bin etc. is also possible.

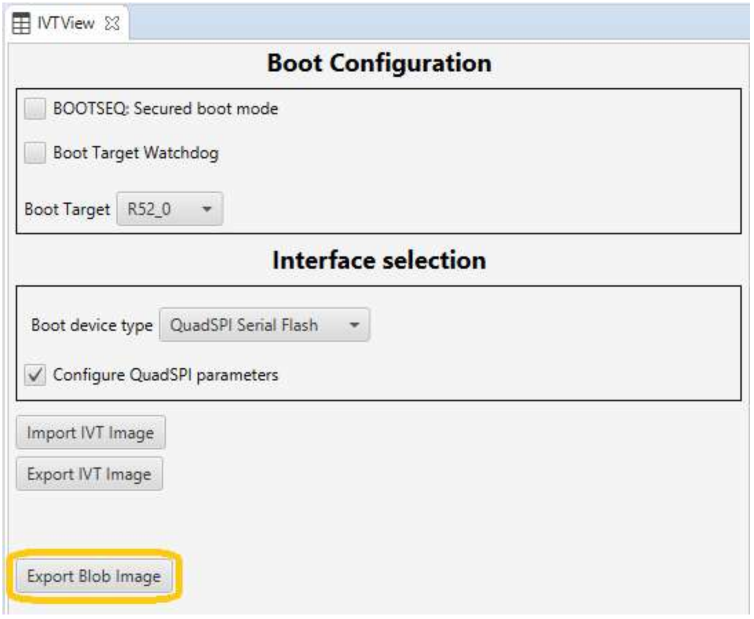
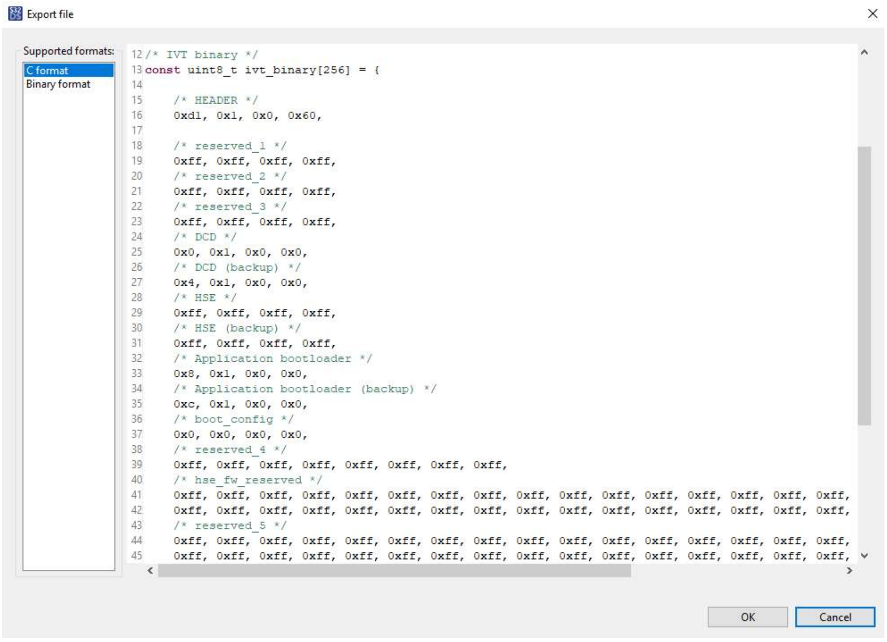


Figure 22: IVT Tool – Export Blob Image

The currently available export options are:

- Exporting IVT Image as binary
- Exporting IVT Image as C code
- Exporting Blob Image as binary
- Exporting Application Bootloader sub-image as binary



```
/* IVT binary */
const uint8_t ivt_binary[256] = {

    /* HEADER */
    0xd1, 0x1, 0x0, 0x60,

    /* reserved_1 */
    0xff, 0xff, 0xff, 0xff,
    /* reserved_2 */
    0xff, 0xff, 0xff, 0xff,
    /* reserved_3 */
    0xff, 0xff, 0xff, 0xff,
    /* DCD */
    0x0, 0x1, 0x0, 0x0,
    /* DCD (backup) */
    0x4, 0x1, 0x0, 0x0,
    /* HSE */
    0xff, 0xff, 0xff, 0xff,
    /* HSE (backup) */
    0xff, 0xff, 0xff, 0xff,
    /* Application bootloader */
    0x8, 0x1, 0x0, 0x0,
    /* Application bootloader (backup) */
    0xc, 0x1, 0x0, 0x0,
    /* boot_config */
    0x0, 0x0, 0x0, 0x0,
    /* reserved_4 */
    0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,
    /* hse_fw_reserved */
    0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,
    0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,
    /* reserved_5 */
    0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,
    0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,
```

Figure 24: IVT Tool – Export image

3 S32 Design Studio Integration

3.1 Open S32 Configuration Tool

To get using S32 Configuration tools user must first setup a project in S32 Design Studio. This is done from File > New > S32DS Application Project.

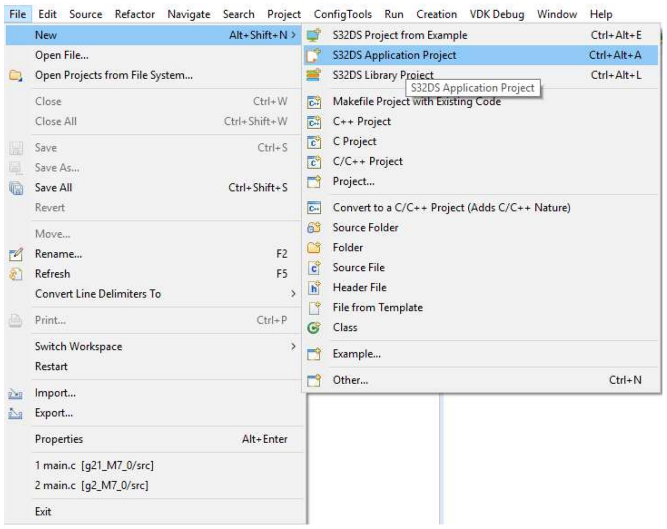
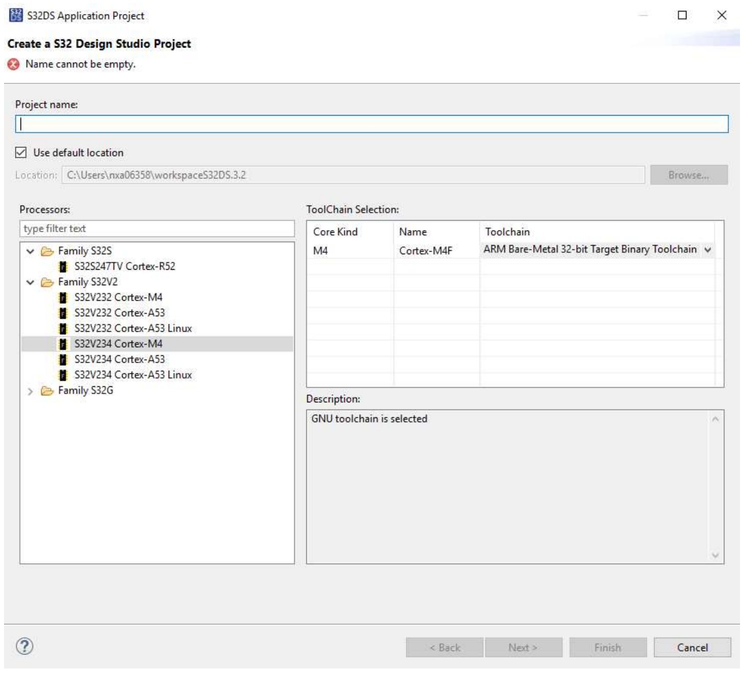


Figure 25: Open S32 Configuration Tool

You will then be prompted to Create a S32 Design Studio Project wizard. There you must choose Project name, Project location and Processor used.


S32DS Application Project

Create a S32 Design Studio Project

Name cannot be empty.

Project name:

☑ Use default location

Location: C:\Users\nxa06358\workspaceS32DS.3.2

Processors:

type filter text

- Family S32S
  - S32S247TV Cortex-R52
- Family S32V2
  - S32V232 Cortex-M4
  - S32V232 Cortex-A53
  - S32V232 Cortex-A53 Linux
  - S32V234 Cortex-M4
  - S32V234 Cortex-A53
  - S32V234 Cortex-A53 Linux
- Family S32G

ToolChain Selection:

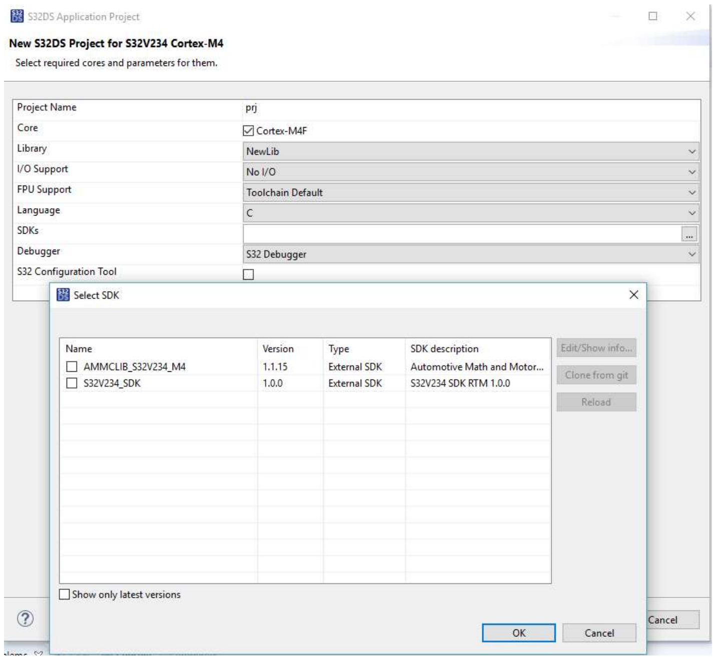
Core Kind	Name	Toolchain
M4	Cortex-M4F	ARM Bare-Metal 32-bit Target Binary Toolchain

Description:

GNU toolchain is selected

Figure 26: Integration - Create Project

Next user must Select required cores and parameters for them. Here you will choose preferred SDK from the ones installed.


S32DS Application Project

New S32DS Project for S32V234 Cortex-M4
Select required cores and parameters for them.

Project Name	prj
Core	☑ Cortex-M4F
Library	NewLib
I/O Support	No I/O
FPU Support	Toolchain Default
Language	C
SDKs	
Debugger	S32 Debugger
S32 Configuration Tool	☐

Select SDK

Name	Version	Type	SDK description
☐ AMMCLIB_S32V234_M4	1.1.15	External SDK	Automotive Math and Motor...
☐ S32V234_SDK	1.0.0	External SDK	S32V234 SDK RTM 1.0.0

☐ Show only latest versions

Figure 27: Integration – Select SDK

By clicking on Finish the project is all setup. To open S32 Configuration Tool views right-click on the project and go to S32 Configuration Tool option or select the same option from the upper toolbar.

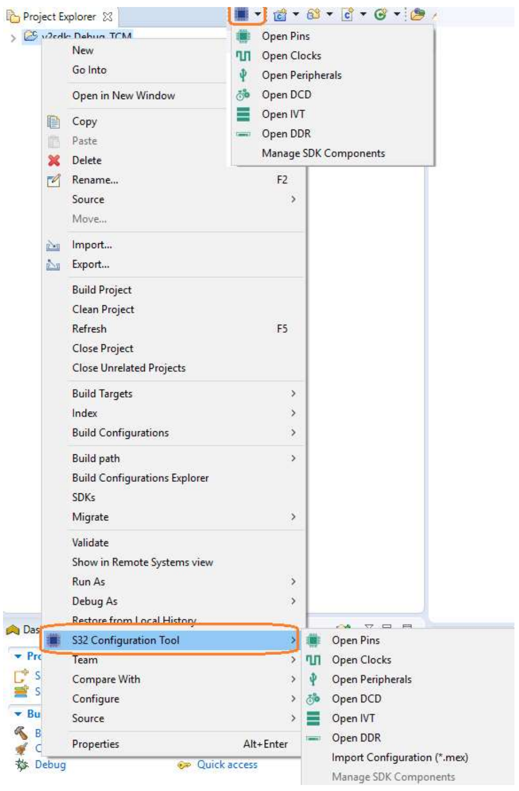


Figure 28: Integration – Open configuration

3.2 Add SDK drivers to project

When the configuration is opened in one of the S32 Configuration tools' perspectives, user can just choose "Manage SDK components" button from the toolbar.

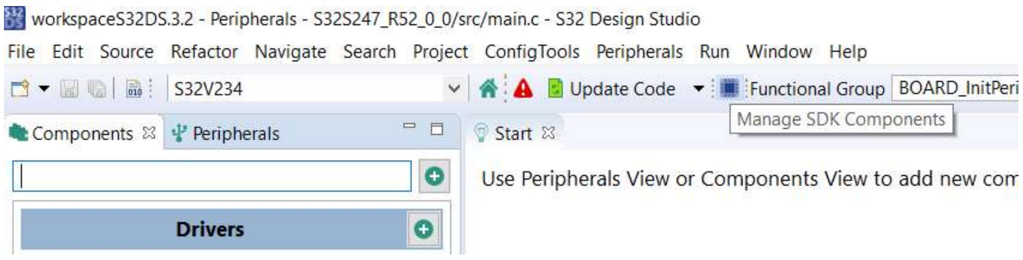
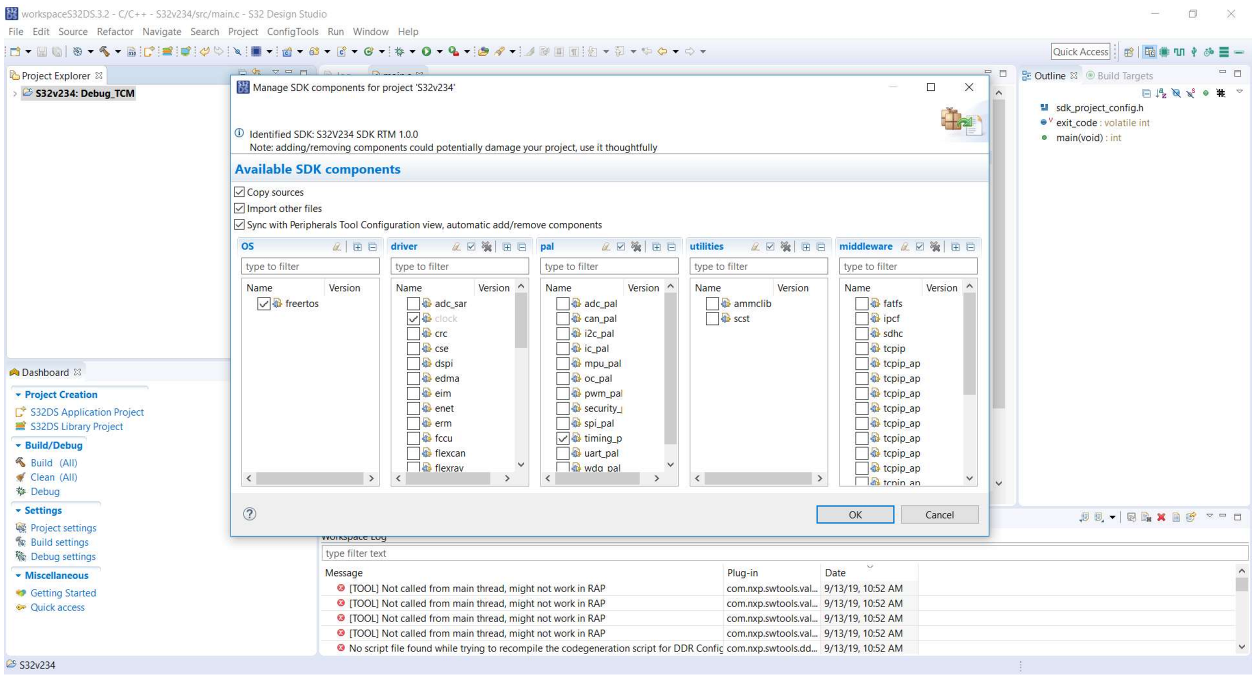

workspaceS32DS.3.2 - Peripherals - S32S247_R52_0_0/src/main.c - S32 Design Studio
File Edit Source Refactor Navigate Search Project ConfigTools Peripherals Run Window Help
S32V234
Update Code
Functional Group BOARD_InitPeri
Manage SDK Components
Components
Peripherals
Start
Use Peripherals View or Components View to add new com
Drivers

Figure 29: Integration – Open Manage SDK Components

After the option is selected, a dialog with the supported S32 SDK components is displayed:



Manage SDK components for project 'S32v234'

Identified SDK: S32V234 SDK RTM 1.0.0
Note: adding/removing components could potentially damage your project, use it thoughtfully

Available SDK components

- [✓] Copy sources
- [✓] Import other files
- [✓] Sync with Peripherals Tool Configuration view, automatic add/remove components

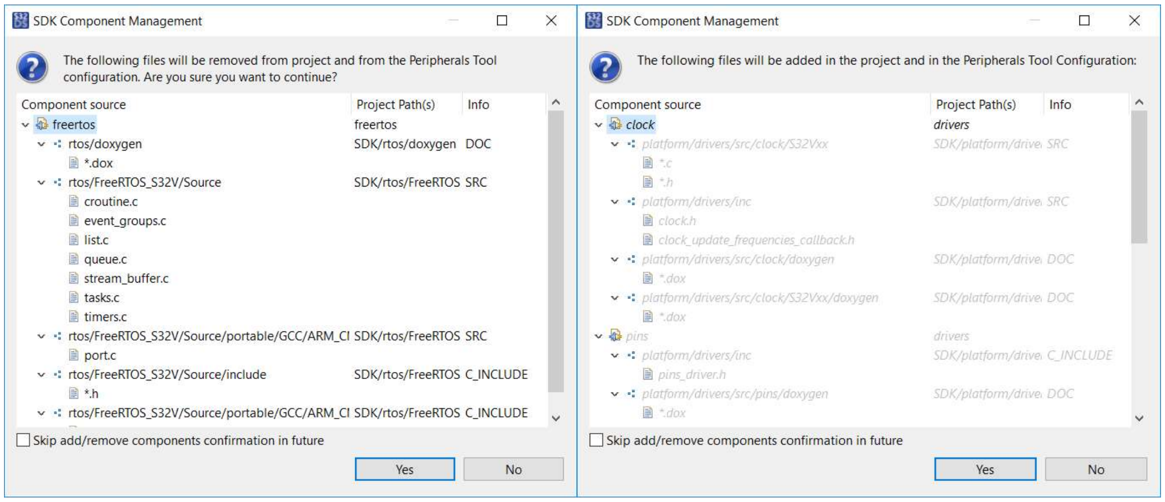
Message	Plug-in	Date
[TOOL] Not called from main thread, might not work in RAP	com.nxp.swtools.val...	9/13/19, 10:52 AM
[TOOL] Not called from main thread, might not work in RAP	com.nxp.swtools.val...	9/13/19, 10:52 AM
[TOOL] Not called from main thread, might not work in RAP	com.nxp.swtools.val...	9/13/19, 10:52 AM
[TOOL] Not called from main thread, might not work in RAP	com.nxp.swtools.val...	9/13/19, 10:52 AM
No script file found while trying to recompile the codegeneration script for DDR Confic	com.nxp.swtools.dd...	9/13/19, 10:52 AM

Figure 30: Integration – Manage components

The available options that can be selected are:

- Copy sources - if checked, sources are copied to the workspace project, otherwise they are linked from existing sources in the SDK.
- Import other files - if checked, other files listed in the SDK components and/or example.xml will be imported.
- Sync with Peripherals Tool Configuration view, automatic add/remove components - if true, for each dependency, if available, in Peripherals Tool Configuration the drivers will be added/removed only if the configuration for the selected project is opened.

When “OK” button is pressed, a pop-up with all the drivers’ sources are shown for each operation, add to project or remove from project:



SDK Component Management

The following files will be removed from project and from the Peripherals Tool configuration. Are you sure you want to continue?

Component source	Project Path(s)	Info
freertos	freertos	
rtos/doxygen	SDK/rtos/doxygen	DOC
*.dox		
rtos/FreeRTOS_S32V/Source	SDK/rtos/FreeRTOS	SRC
croutine.c		
event_groups.c		
list.c		
queue.c		
stream_buffer.c		
tasks.c		
timers.c		
rtos/FreeRTOS_S32V/Source/portable/GCC/ARM_CI	SDK/rtos/FreeRTOS	SRC
port.c		
rtos/FreeRTOS_S32V/Source/include	SDK/rtos/FreeRTOS	C_INCLUDE
*.h		
rtos/FreeRTOS_S32V/Source/portable/GCC/ARM_CI	SDK/rtos/FreeRTOS	C_INCLUDE

Skip add/remove components confirmation in future

SDK Component Management

The following files will be added in the project and in the Peripherals Tool Configuration:

Component source	Project Path(s)	Info
clock	drivers	
platform/drivers/src/clock/S32Vxx	SDK/platform/drive	SRC
*.c		
*.h		
platform/drivers/inc	SDK/platform/drive	SRC
clock.h		
clock_update_frequencies_callback.h		
platform/drivers/src/clock/doxygen	SDK/platform/drive	DOC
*.dox		
platform/drivers/src/clock/S32Vxx/doxygen	SDK/platform/drive	DOC
*.dox		
pins	drivers	
platform/drivers/inc	SDK/platform/drive	C_INCLUDE
pins_driver.h		
platform/drivers/src/pins/doxygen	SDK/platform/drive	DOC
*.dox		

Skip add/remove components confirmation in future

Figure 31: Integration – Add/remove drivers from project