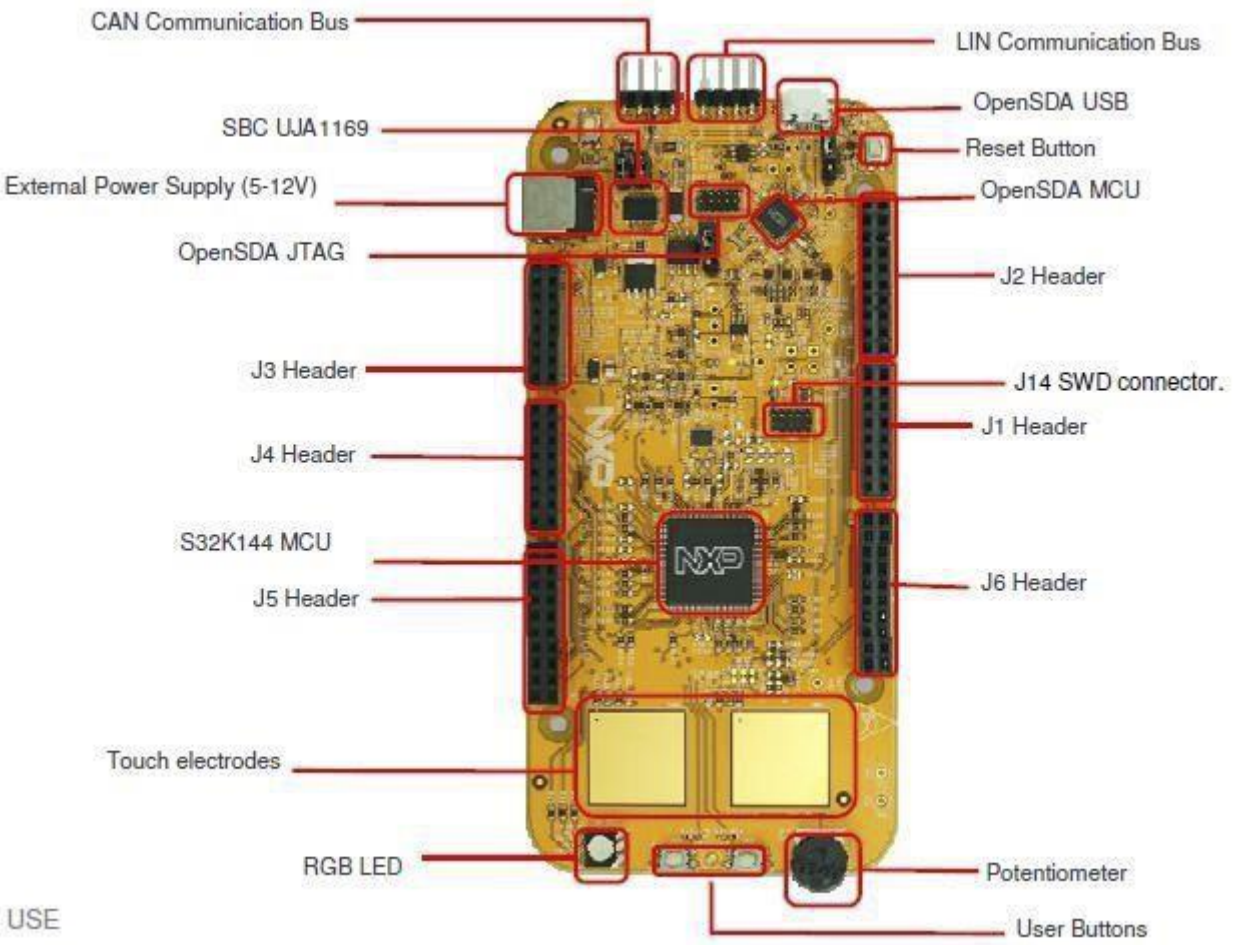
Introduction to S32K144, S32DS, FreeMASTER and MBDT

Contents

S32K144-EVB



NAL USE

S32K144 EVB Features:

- Supports **S32K144 100LQFP**
- Small form factor size supports up to 6" x 4"
- Arduino™ UNO footprint-compatible with expansion "shield" support
- Integrated open-standard serial and debug adapter (OpenSDA) with support for several industry-standard debug interfaces
- Easy access to the MCU I/O header pins for prototyping
- On-chip connectivity for CAN, LIN, UART/SCI.
- SBC UJA1169 and LIN phy TJA1027
- Potentiometer for precise voltage and analog measurement
- RGB LED
- Two push-button switches (SW2 and SW3) and two touch electrodes
- Flexible power supply options
  - microUSB or
  - external 12V power supply

Header/Pinout Mapping for S32K144

J3

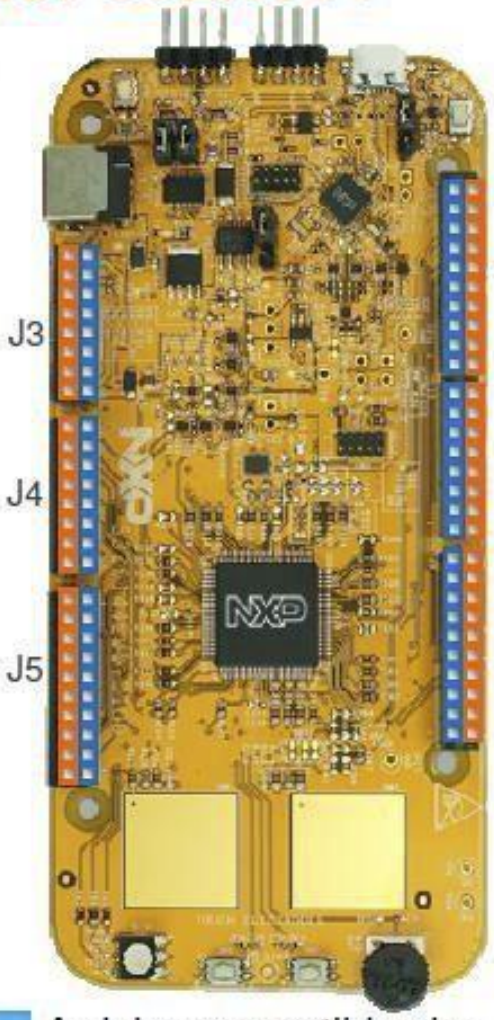
PIN	PORT	FUNCTION	PIN	PORT	FUNCTION
J3-02	PTB6*	GPIO	J3-01		VIN
J3-04	PTB7*	GPIO	J3-03		IOREF
J3-06	PTE0	GPIO	J3-05	PTA5	RESET
J3-08	PTE9	GPIO	J3-07		3V3
J3-10	PTC5	GPIO	J3-09		5V
J3-12	PTC4	GPIO	J3-11		GND
J3-14	PTA10	GPIO	J3-13		GND
J3-16	PTA4	GPIO	J3-15		VIN

J4

PIN	PORT	FUNCTION	PIN	PORT	FUNCTION
J4-02	PTC7	GPIO	J4-01	PTD4	ADC0
J4-04	PTC6	GPIO	J4-03	PTB12	ADC1
J4-06	PTB17	GPIO	J4-05	PTB0	ADC2
J4-08	PTB14	GPIO	J4-07	PTB1	ADC3
J4-10	PTB15	GPIO	J4-09	PTA6/PTE11/PTA2	ADC4
J4-12	PTB16	GPIO	J4-11	PTC0/PTE10/PTA3	ADC5
J4-14	PTC14	GPIO	J4-13	PTE2	ADC6
J4-16	PTC3	GPIO	J4-15	PTE6	ADC7

J5

PIN	PORT	FUNCTION	PIN	PORT	FUNCTION
J5-02	PTE16	GPIO	J5-01	PTA15/PTD11	ADC8
J5-04	PTE15	GPIO	J5-03	PTA16/PTD10	ADC9
J5-06	PTE14	GPIO	J5-05	PTA1	ADC10
J5-08	PTE13	GPIO	J5-07	PTA0	ADC11
J5-10		VDD	J5-09	PTA7	ADC12
J5-12		GND	J5-11	PTB13	ADC13
J5-14	PTE1	GPIO	J5-13	PTC1	ADC14
J5-16	PTD7	GPIO	J5-15	PTC2	ADC15
J5-18	PTD6	GPIO	J5-17	NC	GPIO
J5-20	PTC15	GPIO	J5-19	NC	N/A



J2

PIN	PORT	FUNCTION	PIN	PORT	FUNCTION
J2-19	PTE10/PTA3	D15/I2C_CLK	J2-20	NC	GPIO
J2-17	PTE11/PTA2	D14/I2C_SDA	J2-18	NC	GPIO
J2-15		ANALOGUE REF	J2-16	PTA14	GPIO
J2-13		GND	J2-14	PTE7	GPIO
J2-11	PTB2	D13/SPI_SCK	J2-12	PTC13	GPIO
J2-09	PTB3	D12/SPI_SIN	J2-10	PTC12	GPIO
J2-07	PTB4	D11/SPI_SOUT	J2-08	PTE8	GPIO
J2-05	PTB5	D10/SPI_CS	J2-06	PTD0	GPIO
J2-03	PTD14	D9/PWM	J2-04	PTD16	GPIO
J2-01	PTD13	D8/PWM	J2-02	PTD15	GPIO

J1

PIN	PORT	FUNCTION	PIN	PORT	FUNCTION
J1-15	PTC11/PTE8	D7	J1-16	PTE3	GPIO
J1-13	PTC10/PTC3	D6	J1-14	PTD3	GPIO
J1-11	PTB11	D5	J1-12	PTD5	GPIO
J1-09	PTB10	D4	J1-10	PTD12	GPIO
J1-07	PTB9	D3	J1-08	PTD11	GPIO
J1-05	PTB8	D2	J1-06	PTD10	GPIO
J1-03	PTA3	D1	J1-04	PTA17	GPIO
J1-01	PTA2	D0	J1-02	PTA11	GPIO

J6

PIN	PORT	FUNCTION	PIN	PORT	FUNCTION
J6-19	PTA9	D14	J6-20	PTE4	GPIO
J6-17	PTA8	D15	J6-18	PTE5	GPIO
J6-15	PTE12	D16	J6-16	PTA12	GPIO
J6-13	PTD17	D17	J6-14	PTA13	GPIO
J6-11	PTC9	D18	J6-12		GND
J6-09	PTC8	D19	J6-10		VDD
J6-07	PTD8	D20	J6-08	PTC16	GPIO
J6-05	PTD9	D21	J6-06	PTC17	GPIO
J6-03	PTD2	GPIO	J6-04	PTD3	GPIO
J6-01	PTD0	GPIO	J6-02	PTD1	GPIO

Arduino compatible pins

NXP pins

*0ohm resistor is not connected

Jumper Settings

Jumper	Configuration	Description
J104	1-2	Reset signal to OpenSDA, use to enter into OpenSDA Bootloader mode
	2-3 (Default)	Reset signal direct to the MCU, use to reset S32K144.
J107	1-2	S32K144 powered by 12V power source.
	2-3 (Default)	S32K144 powered by USB micro connector.
J109/J108	1-2 (Default)	Removes CAN termination resistor

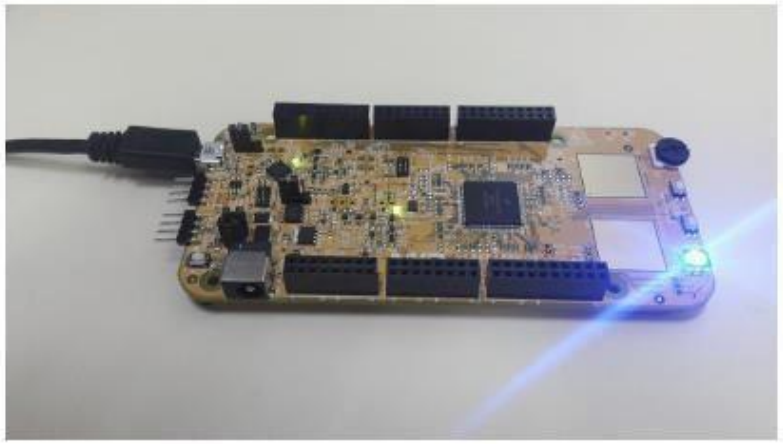
Please do not change the jumpers at the present time.

HMI mapping

Component	S32K144
Red LED	PTD15 (FTM0 CH0)
Blue LED	PTD0(FTM0 CH2)
Green LED	PTD16(FTM0 CH1)
Potentiometer	PTC14 (ADC0_SE12)
SW2	PTC12
SW3	PTC13
OpenSDA UART TX	PTC7(LPUART1_TX)
OpenSDA UART RX	PTC6(LPUART1_RX)
CAN TX	PTE5(CAN0_TX)
CAN RX	PTE4 (CAN0_RX)
LIN TX	PTD7(LPUART2_TX)
LIN RX	PTD6 (LPUART2_RX)
SBC_SCK	PTB14 (LPSPI1_SCK)
SBC_MISO	PTB15(LPSPI1_SIN)
SBC_MOSI	PTB16(LPSPI1_SOUT)
SBC_CS	PTB17(LPSPI1_PCS3)

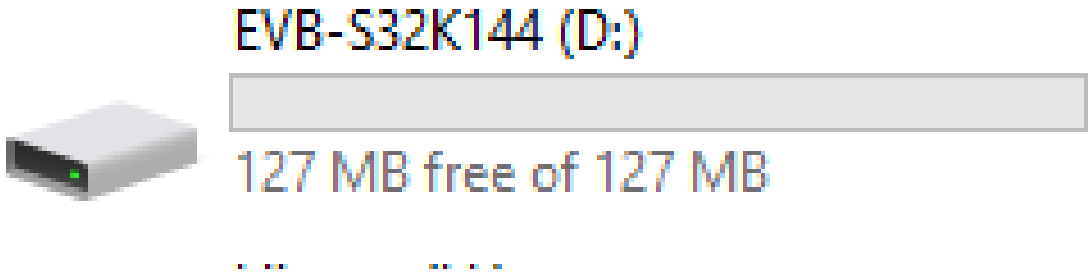
Power up the EVB board

- Powers the S32K144EVB evaluation board from a USB. By default, the USB power is enabled by J107 jumper (2-3 closed).
- Connect the USB cable to a PC and connect micro USB connector of the USB cable to micro-B port J7 on the S32K144EVB.
- Allow the PC to automatically configure the USB drivers if needed.
- When EVB is powered from USB, LEDs D2 and D3 should light green.
- The EVB board is preloaded with a software toggling the RGB LED colours periodically between RED-GREEN-BLUE.



• When powered through USB, LEDs D2 and D3 should light green

• Once the board is recognized, it should appear as a mass storage device
in your PC with the name EVB-S32K144.

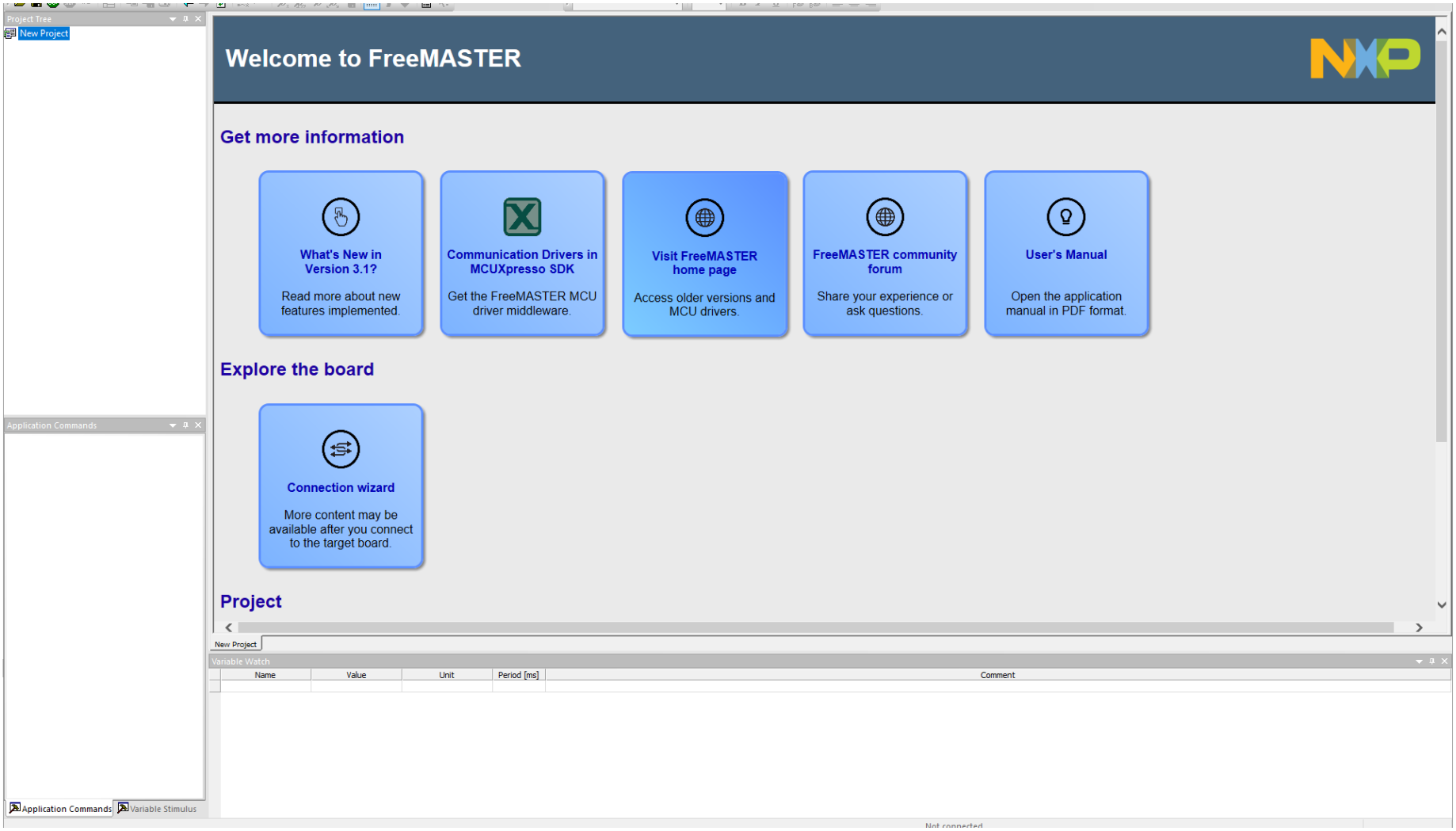


FreeMASTER

Install the FreeMASTER tool

• Download and install the FreeMASTER PC application www.nxp.com/FreeMASTER.

• Open the FreeMASTER application on your PC. You should see Welcome
page as given in next page.

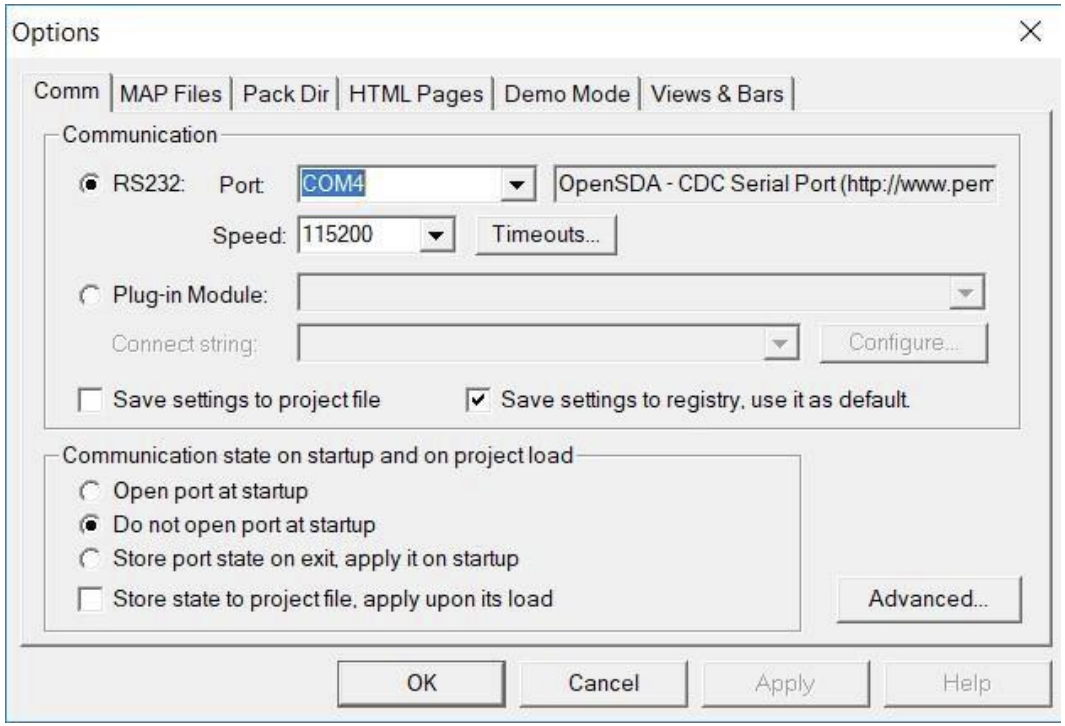

Project Tree
New Project
Application Commands
Application Commands
Variable Stimulus
Welcome to FreeMASTER
Get more information
What's New in Version 3.1?
Read more about new features implemented.
Communication Drivers in MCUXpresso SDK
Get the FreeMASTER MCU driver middleware.
Visit FreeMASTER home page
Access older versions and MCU drivers.
FreeMASTER community forum
Share your experience or ask questions.
User's Manual
Open the application manual in PDF format.
Explore the board
Connection wizard
More content may be available after you connect to the target board.
Project
New Project
Variable Watch
Name
Value
Unit
Period [ms]
Comment
Not connected

Setup of serial connection in the FreeMASTER tool

- Go to Project > Options > Comm.

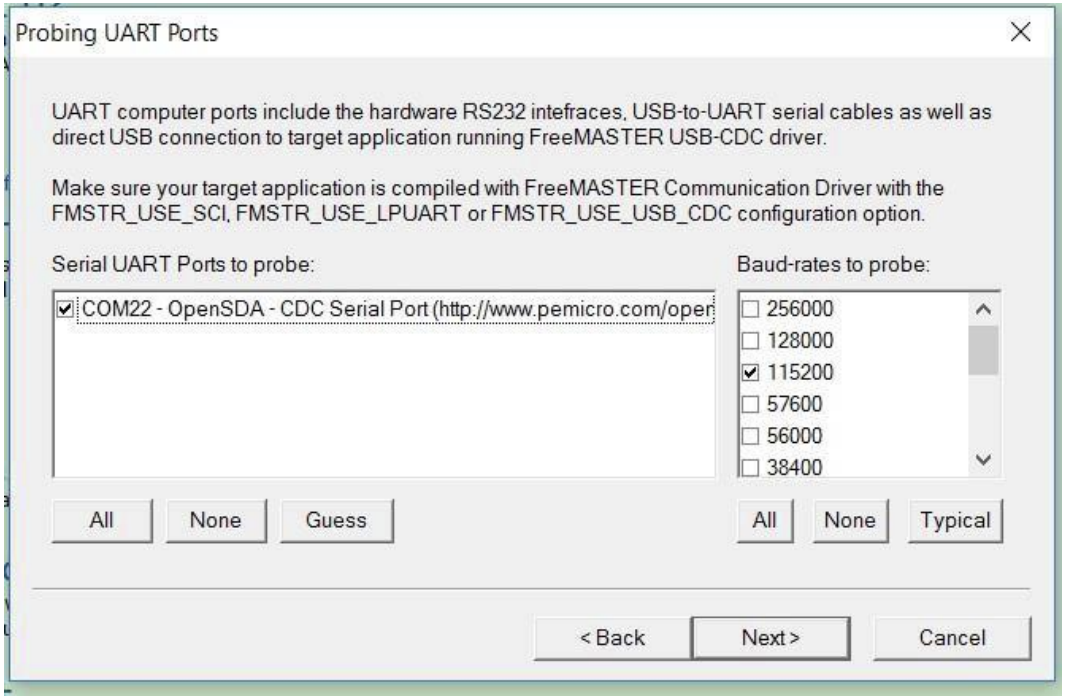
Setup communication port to the one available in the drop down.
“OpenSDA” and speed to 115200.

You may also use the following setup for serial connection in the FreeMASTER tool.



Another way to Setup the serial connection in the FreeMASTER tool

- Select Tools > Connection Wizard > Next.
- Select “use direct connection to onboard USB port”, Next
- Select OpenSDA COM port and 115200 baud rate > Next > Finish.



S32 Design Studio Installation and Creating a Project.

I. Download and Install S32DS

- Click on the following link:

https://www.nxp.com/support/developer-resources/run-timesoftware/s32-design-studio-ide/s32-design-studio-ide-for-arm-basedmcus:S32DS-ARM

- Select download

S32DS-ARM: S32 Design Studio IDE for Arm® based MCUs

Jump To

Overview

The S32 Design Studio IDE is a complimentary integrated development environment for Automotive and Ultra-Reliable Arm based MCUs that enables editing, compiling and debugging of designs. Based on, open-source software including Eclipse IDE, GNU Compiler Collection (GCC) and GNU Debugger (GDB), the S32 Design Studio IDE offers designers a straightforward development tool with no code-size limitations. NXP software, along with the S32 Design Studio IDE provide a comprehensive enablement environment that reduces development time.

Features

- Complementary, unlimited code size IDE for NXP Automotive Arm based MCUs
- GNU toolchain with GCC Compiler 4.9 from Arm includes : newlib, newlib-nano, ewl and ewl-nano
- Advanced FreeRTOS kernel aware debug support
- Peripherals Register View
- New Project wizard to create bare metal or SDK projects
- Example projects
- Integrated NXP Tools

More

Download

- Download S32 Design Studio for Arm 2.2:

IDE and Build Tools (4)

S32 Design Studio for ARM 2.2 – Windows/Linux(REV 2.2)

S32 Design Studio for Arm v2.2 contains GNU Build Tools for Arm Embedded Processors, along with various debugger options, fully integrated S32 SDK 3.0.2, includes S32DS Extension and Updates and extended Getting Started page, and much more. Read about the New Features, Device Support, and Bug Fixes in the release notes.

FLEXERA 1 KB S32DS-IDE-ARM-V2-X 2020-01-29 09:59:00

DOWNLOAD

S32 Design Studio for Arm 2018.R1 – Windows/Linux(REV 2018-R1)

FLEXERA 1 KB S32-DS-ARM_v2018-R1 2018-02-06 17:30:00

DOWNLOAD

S32 Design Studio for Arm v2.0 - Windows/Linux(REV 2.0)

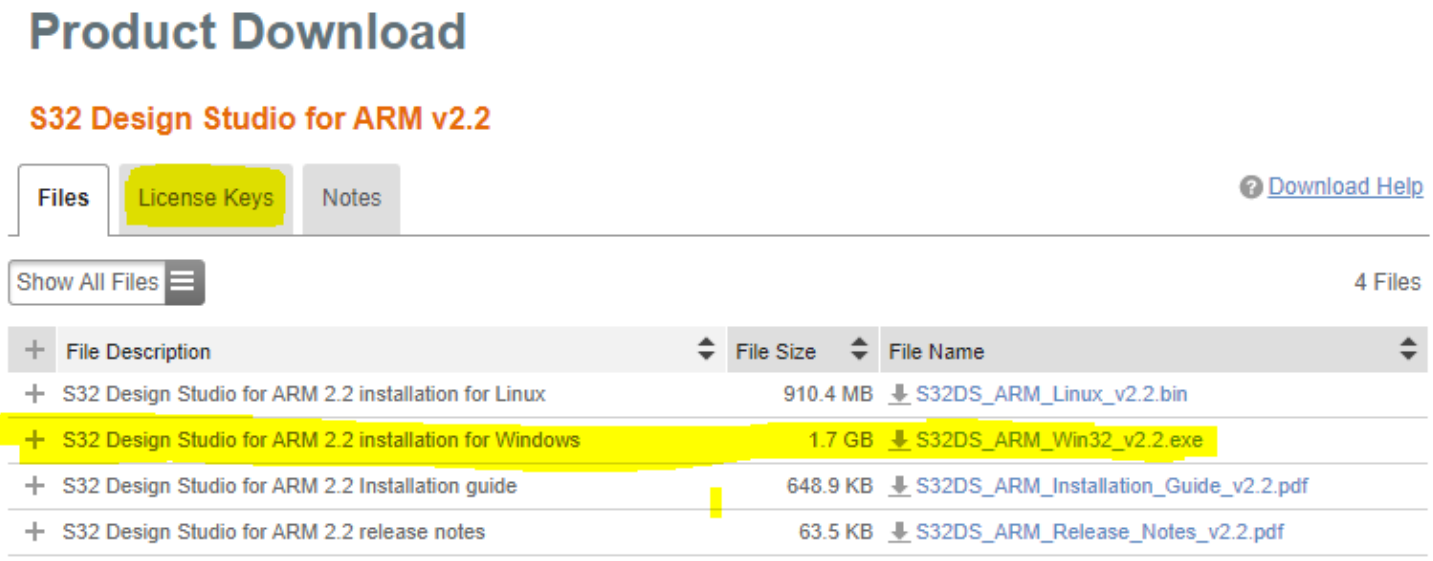
S32 Design Studio for Arm v2.0 for Automotive and Ultra-Reliable MCUs, based on the Eclipse Neon 4.6 framework, contains GNU Build Tools for Arm Embedded Processors, along with various debugger options and much more. Read about the New Features, New Device Support, and Bug Fixes in the release notes.

FLEXERA 1 KB S32DS-ARM-2-0 2017-08-15 19:00:00

DOWNLOAD

• You will be prompted to sign in. Sign in/ Sign Up

Please note that you need to generate the license key for installing the
software by clicking on the “License Keys” tab. Use the



“activation code” to complete your installation.

View

Item Description	**S32 Design Studio for ARM v2.2**
Order Number	S32DS-IDE-ARM-V2-X_152545377
Purchase Order Number	
Total Number of Licenses:	101
Activation Code	AF7D-68BD-7CFC-C0BF

License Applicable to Product(s):

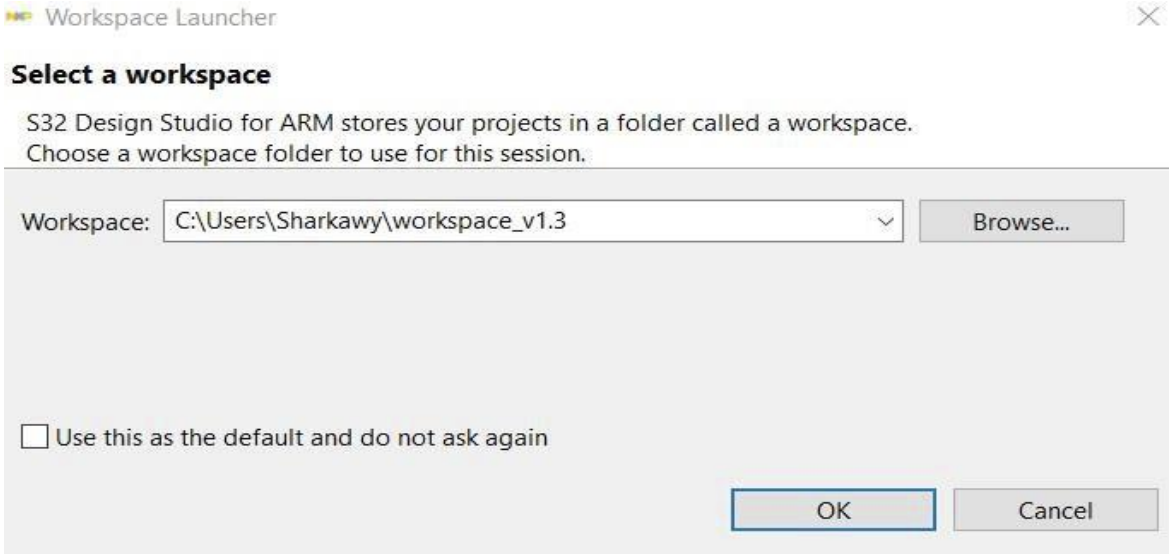
Version	Description
2.2	S32 Design Studio for ARM v2.2 (View EULA)
	100 Available

License Quantity: 1
Fulfillment ID: 181006337
Expiration Date: Jan 20, 2025
Product: S32 Design Studio for ARM v2.2
Machine: 93C43CD5155E92553101AF0B660CE04065ADFADB
Activation Code: AF7D-68BD-7CFC-C0BF

Complete the installation of S32 DS.

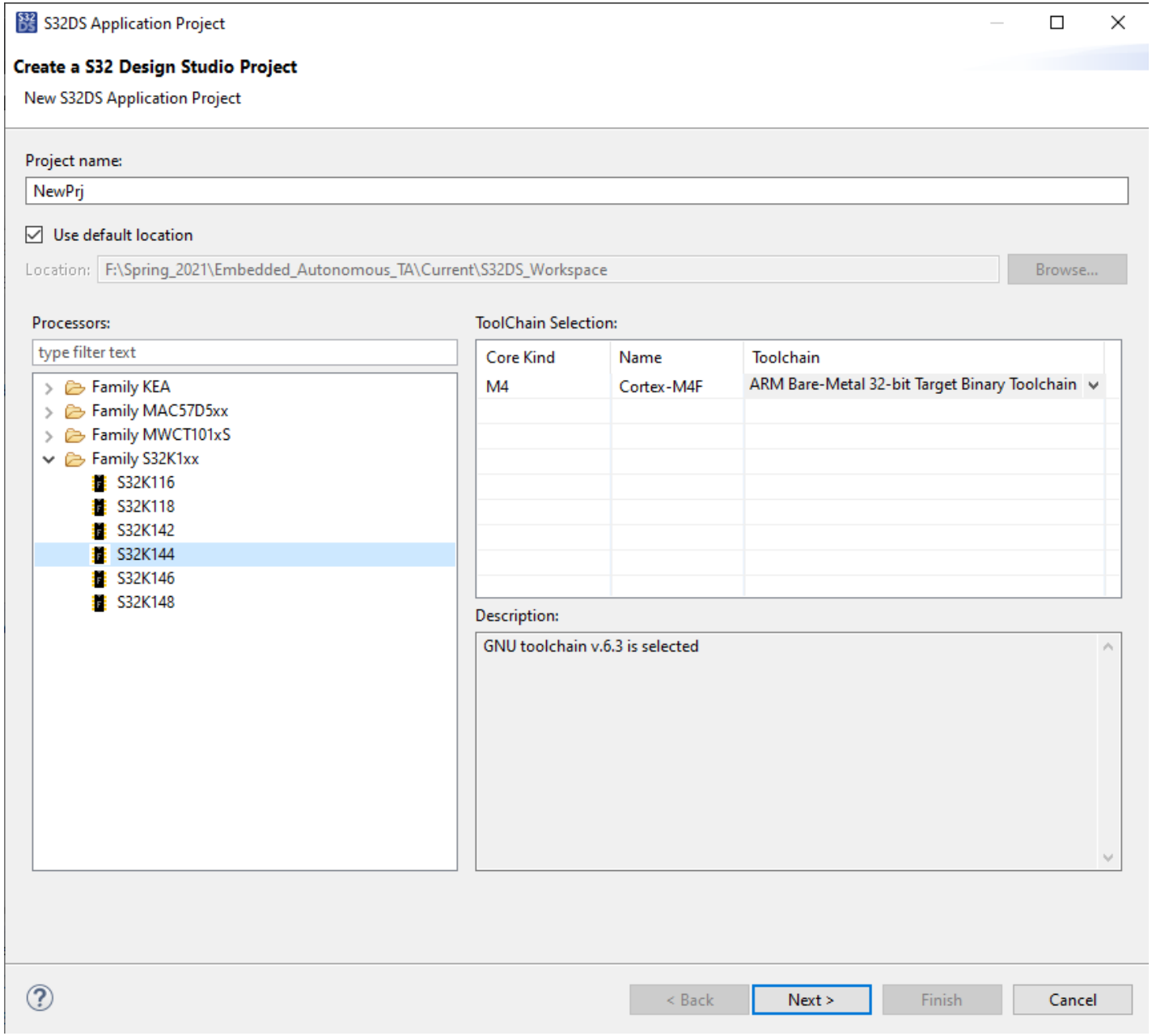
II. Create a S32 SDK project using the New S32DS Project wizard

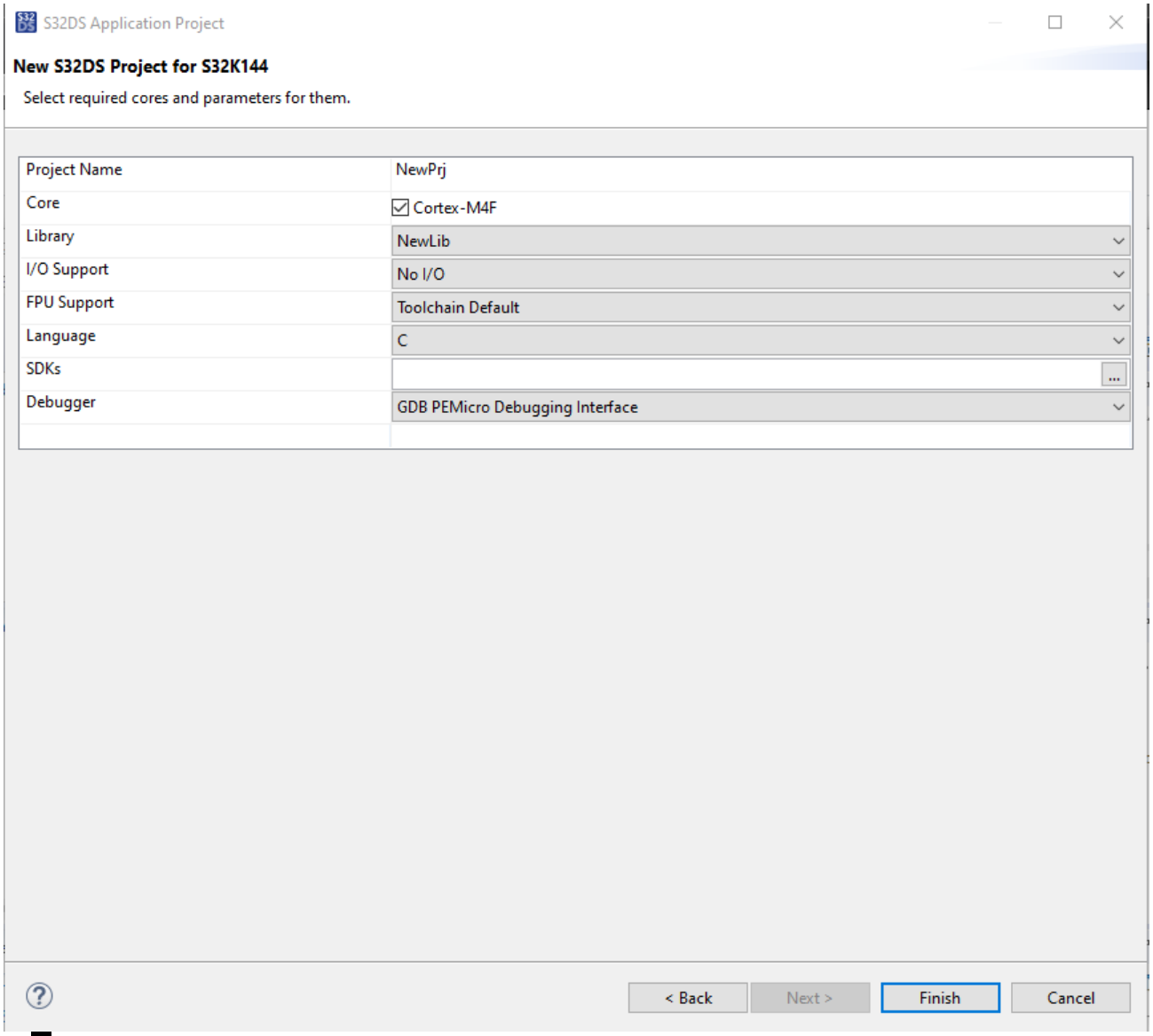
- Start -> programs -> Click on “S32 Design Studio” icon
- Select workspace:
  - Choose default or specify new one
  - Suggestion: Uncheck the box “Use this as the default and do not ask again”
  - Click OK



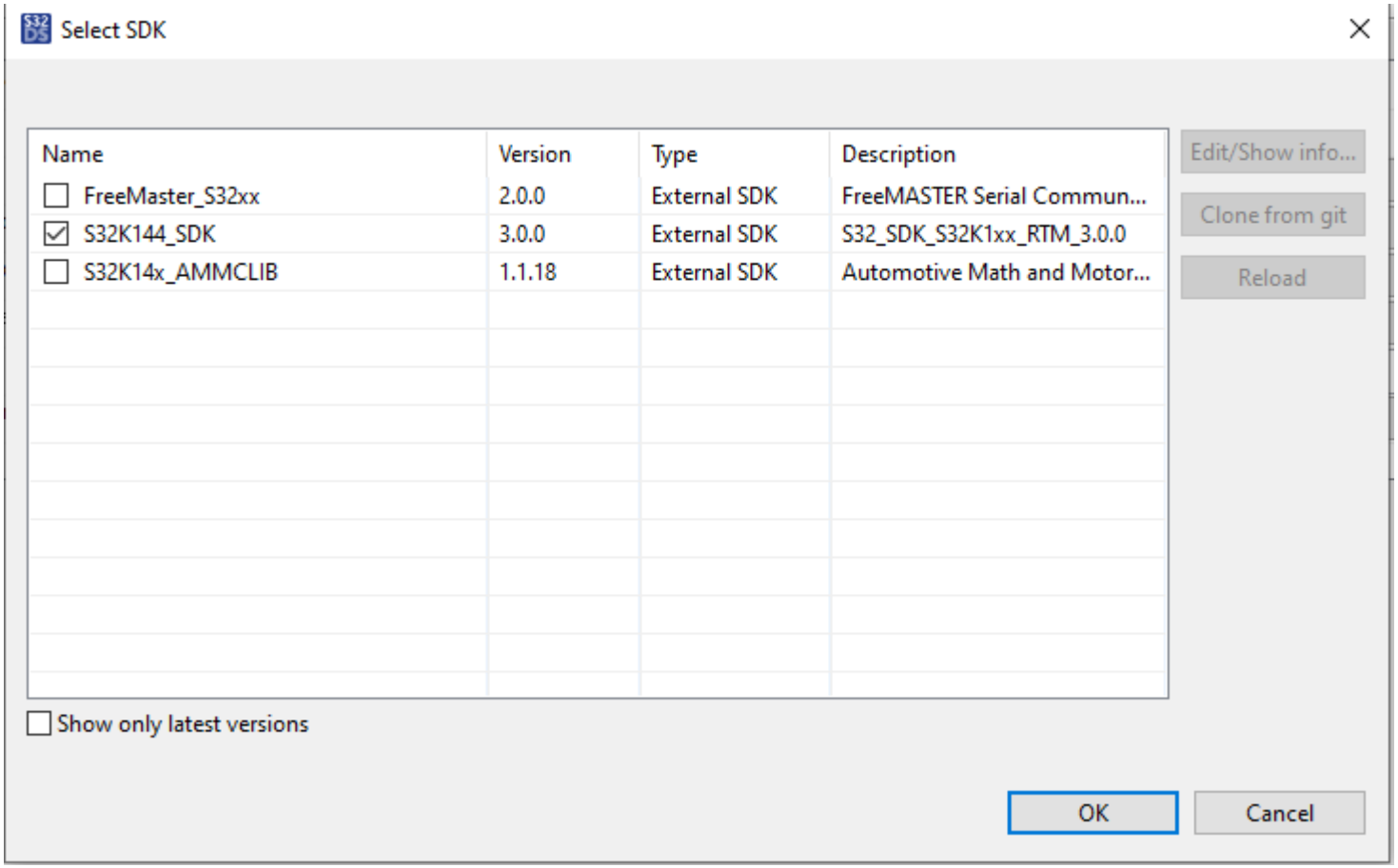
Create a S32 SDK project using the New S32DS Project wizard.

- Select File > New > Application Project, from the IDE menu bar.
- Specify a name for the new project. For example, enter the project name as Project1.
- Select the Processor S32K144 from Processors-> Family S32K1xx section.
- Click Next.

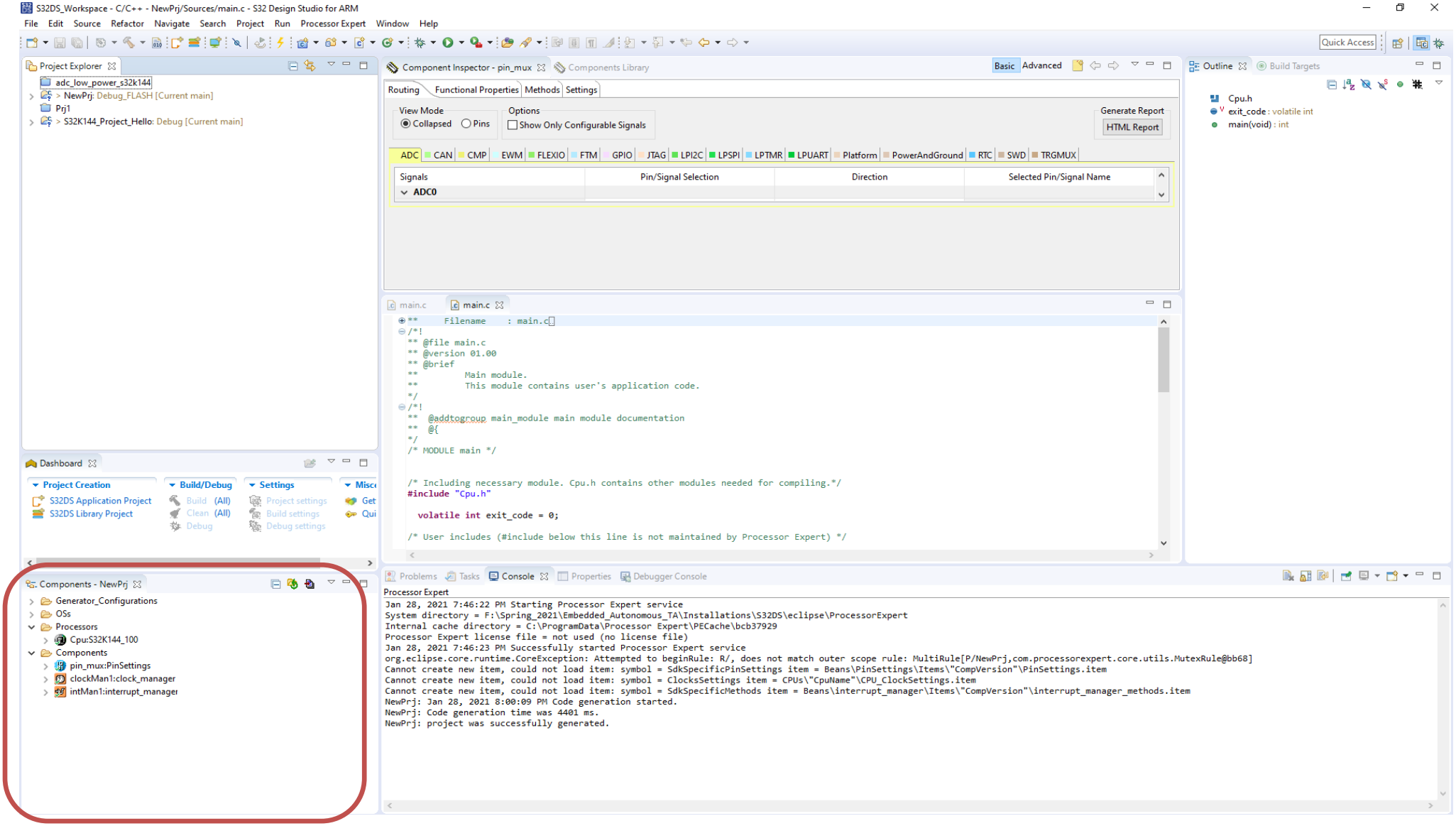

S32DS Application Project
Create a S32 Design Studio Project
New S32DS Application Project
Project name:
NewPrj
Use default location
Location:
F:\Spring_2021\Embedded_Autonomous_TA\Current\S32DS_Workspace
Browse...
Processors:
type filter text
Family KEA
Family MAC57D5xx
Family MWCT101xS
Family S32K1xx
S32K116
S32K118
S32K142
S32K144
S32K146
S32K148
ToolChain Selection:
Core Kind
Name
Toolchain
M4
Cortex-M4F
ARM Bare-Metal 32-bit Target Binary Toolchain
Description:
GNU toolchain v.6.3 is selected
< Back
Next >
Finish
Cancel



- Click on button ... to select available SDKs.



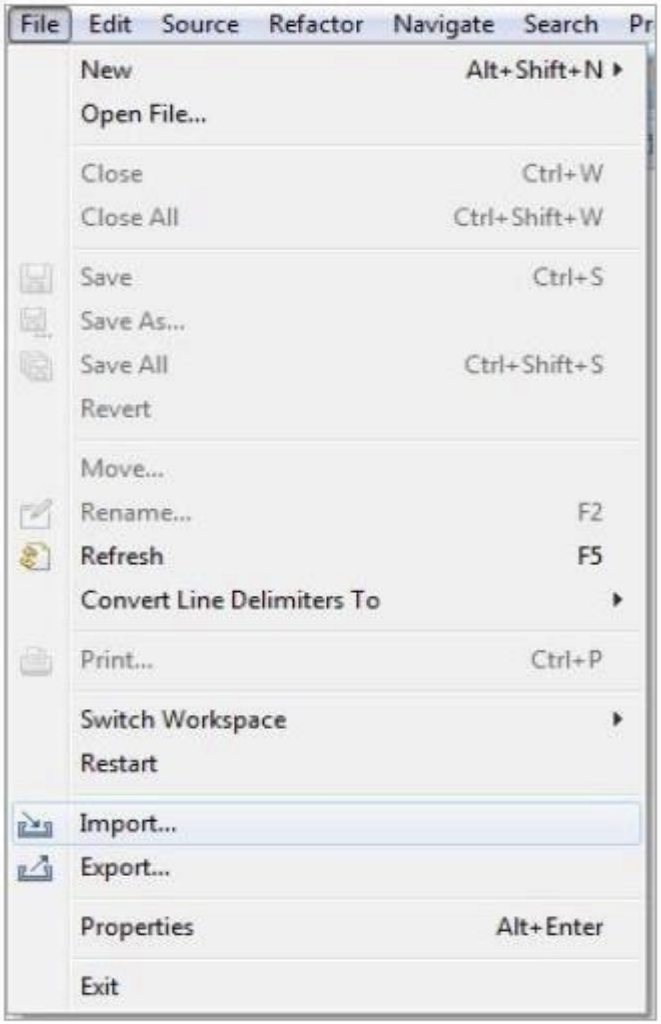
- Click OK.
- Click Finish.


S32DS_Workspace - C/C++ - NewPrj/Sources/main.c - S32 Design Studio for ARM
File Edit Source Refactor Navigate Search Project Run Processor Expert Window Help
Project Explorer
adc_low_power_s32k144
> NewPrj: Debug_FLASH [Current main]
Prj1
> S32K144_Project_Hello: Debug [Current main]
Component Inspector - pin_mux
Components Library
Basic Advanced
Routing Functional Properties Methods Settings
View Mode
Collapsed Pins
Options
Show Only Configurable Signals
Generate Report
HTML Report
ADC CAN CMP EWM FLEXIO FTM GPIO JTAG LPI2C LPSPI LPTMR LPUART Platform PowerAndGround RTC SWD TRGMUX
Signals
Pin/Signal Selection
Direction
Selected Pin/Signal Name
ADC0
Outline
Build Targets
Cpu.h
exit_code : volatile int
main(void) : int
main.c
main.c
** Filename : main.c
/*!
** @file main.c
** @version 01.00
** @brief
** Main module.
** This module contains user's application code.
*/
/*!
** @addtogroup main_module main module documentation
** @{
*/
/* MODULE main */
/* Including necessary module. Cpu.h contains other modules needed for compiling.*/
#include "Cpu.h"
volatile int exit_code = 0;
/* User includes (#include below this line is not maintained by Processor Expert) */
Dashboard
Project Creation
S32DS Application Project
S32DS Library Project
Build/Debug
Build (All)
Clean (All)
Debug
Settings
Project settings
Build settings
Debug settings
Components - NewPrj
Generator_Configurations
OSs
Processors
Cpu:S32K144_100
Components
pin_mux:PinSettings
clockMan1:clock_manager
intMan1:interrupt_manager
Problems Tasks Console Properties Debugger Console
Processor Expert
Jan 28, 2021 7:46:22 PM Starting Processor Expert service
System directory = F:\Spring_2021\Embedded_Autonomous_TA\Installations\S32DS\eclipse\ProcessorExpert
Internal cache directory = C:\ProgramData\Processor Expert\PECache\bcb37929
Processor Expert license file = not used (no license file)
Jan 28, 2021 7:46:23 PM Successfully started Processor Expert service
org.eclipse.core.runtime.CoreException: Attempted to beginRule: R/, does not match outer scope rule: MultiRule[P/NewPrj,com.processorexpert.core.utils.MutexRule@bb68]
Cannot create new item, could not load item: symbol = SdkSpecificPinSettings item = Beans\PinSettings\Items\"CompVersion"\PinSettings.item
Cannot create new item, could not load item: symbol = ClocksSettings item = CPUs\"CpuName"\CPU_ClockSettings.item
Cannot create new item, could not load item: symbol = SdkSpecificMethods item = Beans\interrupt_manager\Items\"CompVersion"\interrupt_manager_methods.item
NewPrj: Jan 28, 2021 8:00:09 PM Code generation started.
NewPrj: Code generation time was 4401 ms.
NewPrj: project was successfully generated.

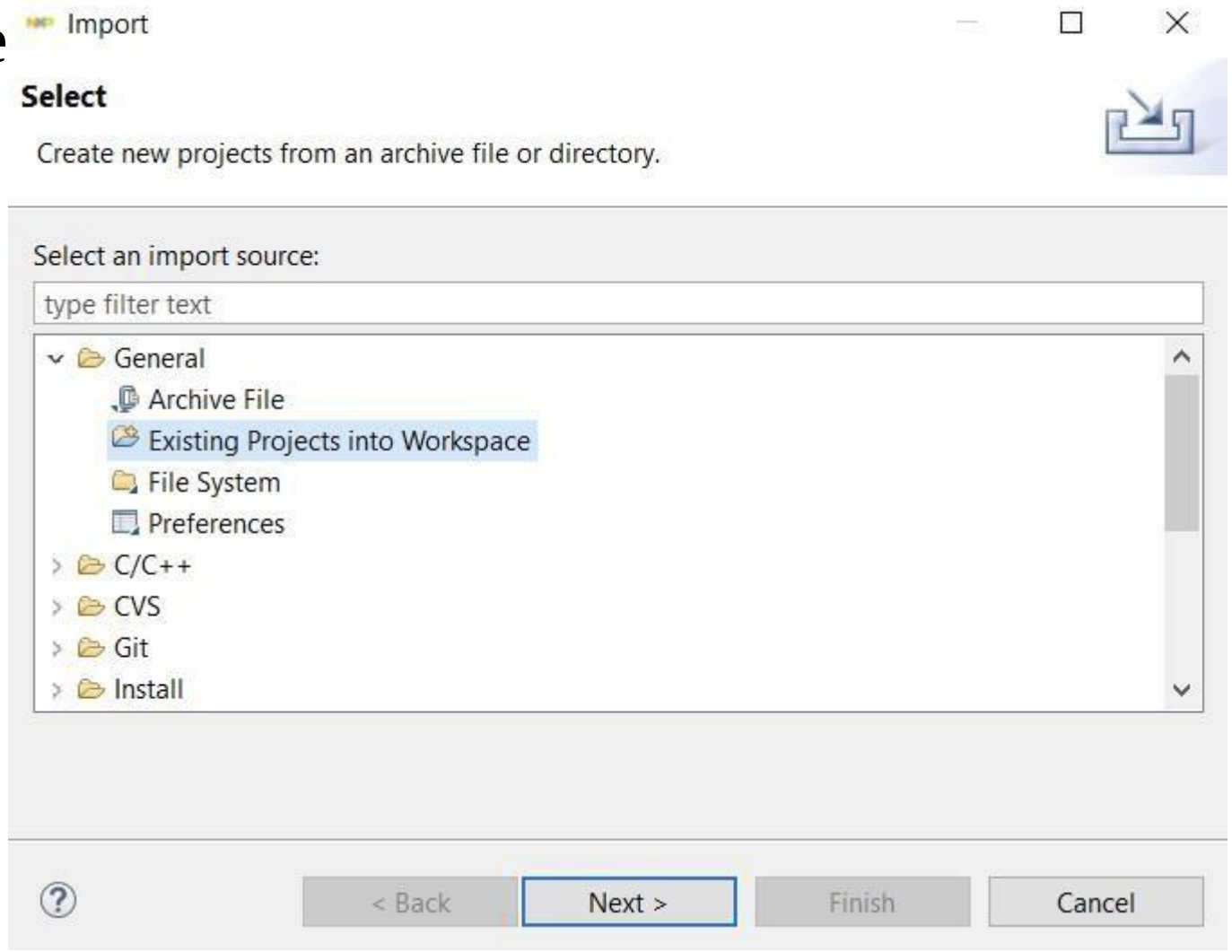
- Processor Expert components can be observed at the Components View.
- Start project build by clicking on Build  Icon or invoking right click menu on project and selecting Build Project.
- This build should execute without any errors.

III. Importing an existing project

- Select File > Import, from the IDE menu.



• Expand General tree and Select Existing Projects into Workspace



• Click Next.

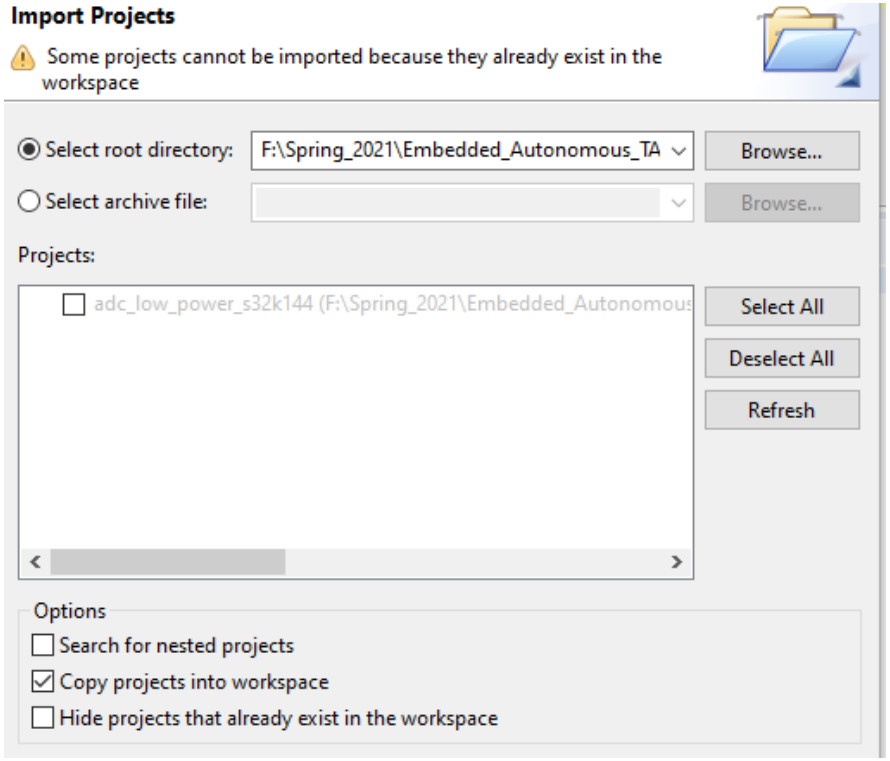
- Click Browse and select the example folder from SDK installation directory to search for the

  “ADC_LOW_POWER” Eclipse project (For example, the “ADC_LOW_POWER” folder can be located at:

  {S32 DS Installation Location}\ S32DS\ software\ S32SDK_S32K1xx_RTM_3.0.0\ examples\ S32K144\ demo_apps\ adc_low_power.
  C:\NXP\S32DS_ARM_v1.3\S32DS\software\S32SDK_S32K1xx_RTM_3.0.0\ examples\S32K144\ demo_apps\ adc_low_power).

- It is useful that you copy project to your workspace by checking the “copy project into workspace”. This will create a copy of the project in your workspace and will not affect the original after changes are made.

- Click Finish to load the project.

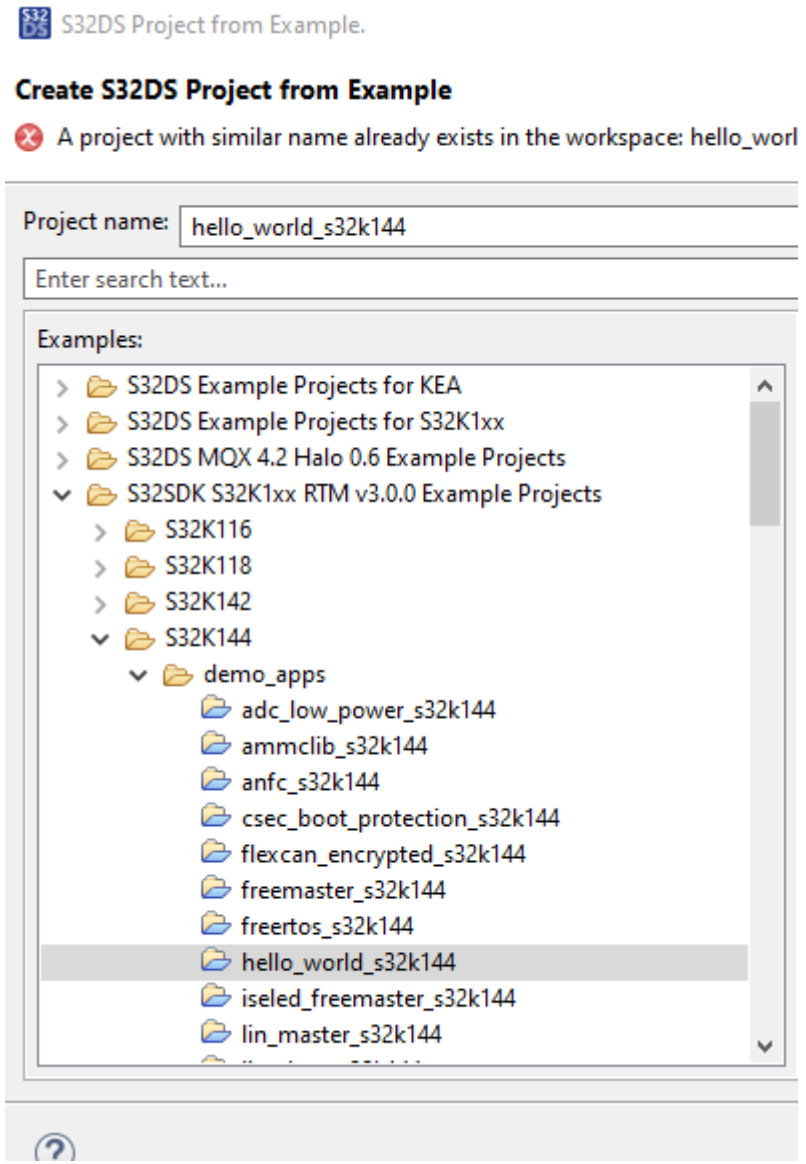


Using New Project from Example

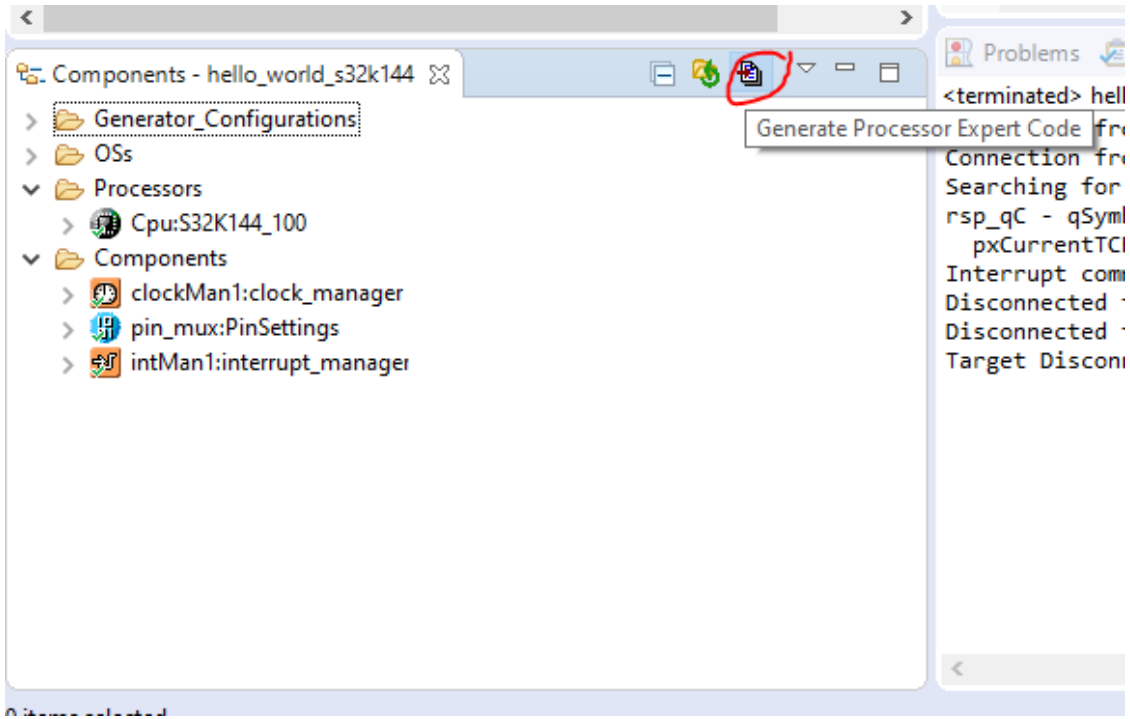
- Select

File -> New ->
S32DS Project from Example ->
S32DS S32K1xx RTM v3.00
Example Projects ->
S32K144 -> hello_world_s32k144.

- This is an LED blink project.
- The project will be copied in your workspace.



- Build the project.
- The error generated is due to missing files of cpu.h, clockman1.h, pin_mux.h, These are produced by the components “**Processor expert code**” generators.
- Click on the Components window
- Select **Generate Processor Expert Code Icon** to generate those header files to support your project.
- Then Build again. The errors should be solved now.



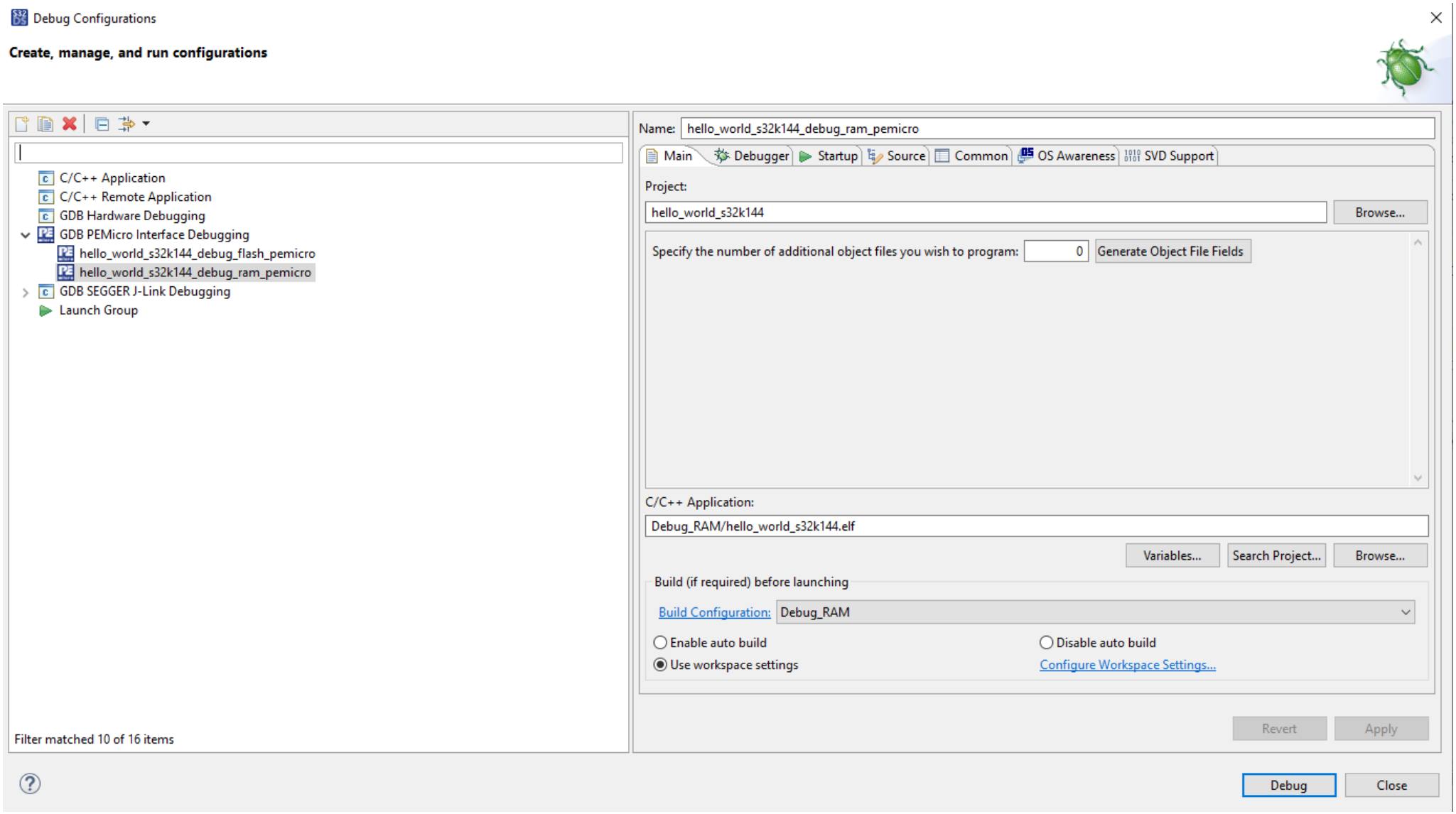
Debugging Projects

- Select the project.

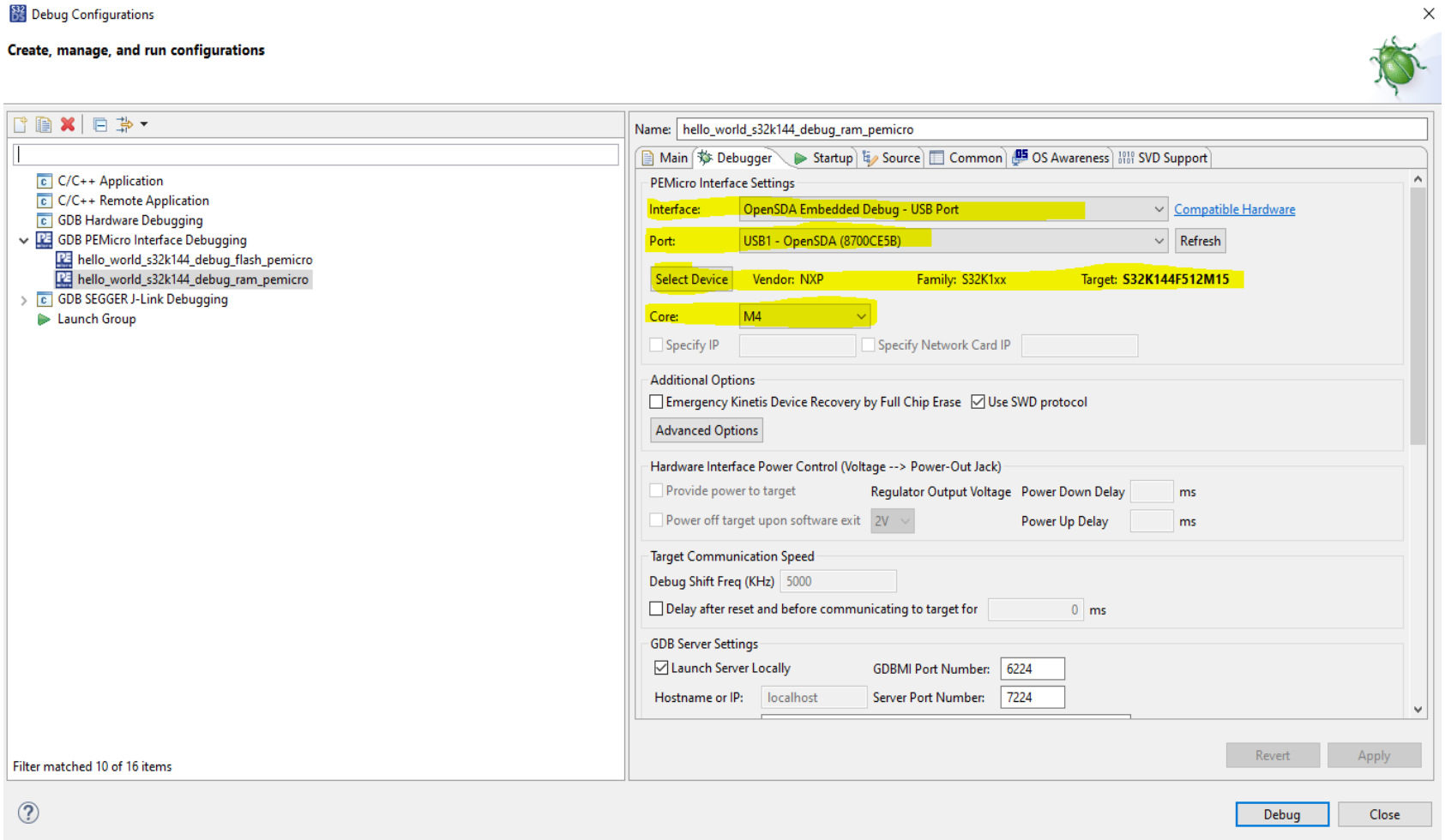
- Click on  select Debug Configurations.

  Alternatively, you can select Run > Debug Configurations from the IDE menu bar.

  (Please be sure that the S32K144 board is connected to your laptop)


Debug Configurations
Create, manage, and run configurations
C/C++ Application
C/C++ Remote Application
GDB Hardware Debugging
GDB PEMicro Interface Debugging
hello_world_s32k144_debug_flash_pemicro
hello_world_s32k144_debug_ram_pemicro
GDB SEGGER J-Link Debugging
Launch Group
Filter matched 10 of 16 items
Name: hello_world_s32k144_debug_ram_pemicro
Main
Debugger
Startup
Source
Common
OS Awareness
SVD Support
Project:
hello_world_s32k144
Browse...
Specify the number of additional object files you wish to program:
0
Generate Object File Fields
C/C++ Application:
Debug_RAM/hello_world_s32k144.elf
Variables...
Search Project...
Browse...
Build (if required) before launching
Build Configuration:
Debug_RAM
Enable auto build
Disable auto build
Use workspace settings
Configure Workspace Settings...
Revert
Apply
Debug
Close

- Verify the values in the Debug tab with the highlighted values in the Image and then Click Debug.



• This will compile and download the code to your S32K board and the RGB LED on the board will be blinking as your code executes on your evaluation board.

Model Based Design Toolbox (MBDT)

Prerequisite for MATLAB for MBDT to work:

1. MATLAB coder (From MATLAB Addons)
2. Simulink Coder (From MATLAB Addons)
3. Embedded Coder (From MATLAB Addons)

Installing Model Based Design Toolbox (MBDT)

Go to www.nxp.com/mctoolbox

1. Click on “Download”.
2. Login with your NXP login.
3. Select “Previous Tab”. Check next page for reference.
4. Select version 4.0.0: Model-Based Design Toolbox for S32K14x Automotive Microprocessors Family.
5. Read and Accept the terms & conditions.

Product Information

Automotive SW - Model-Based Design Toolbox

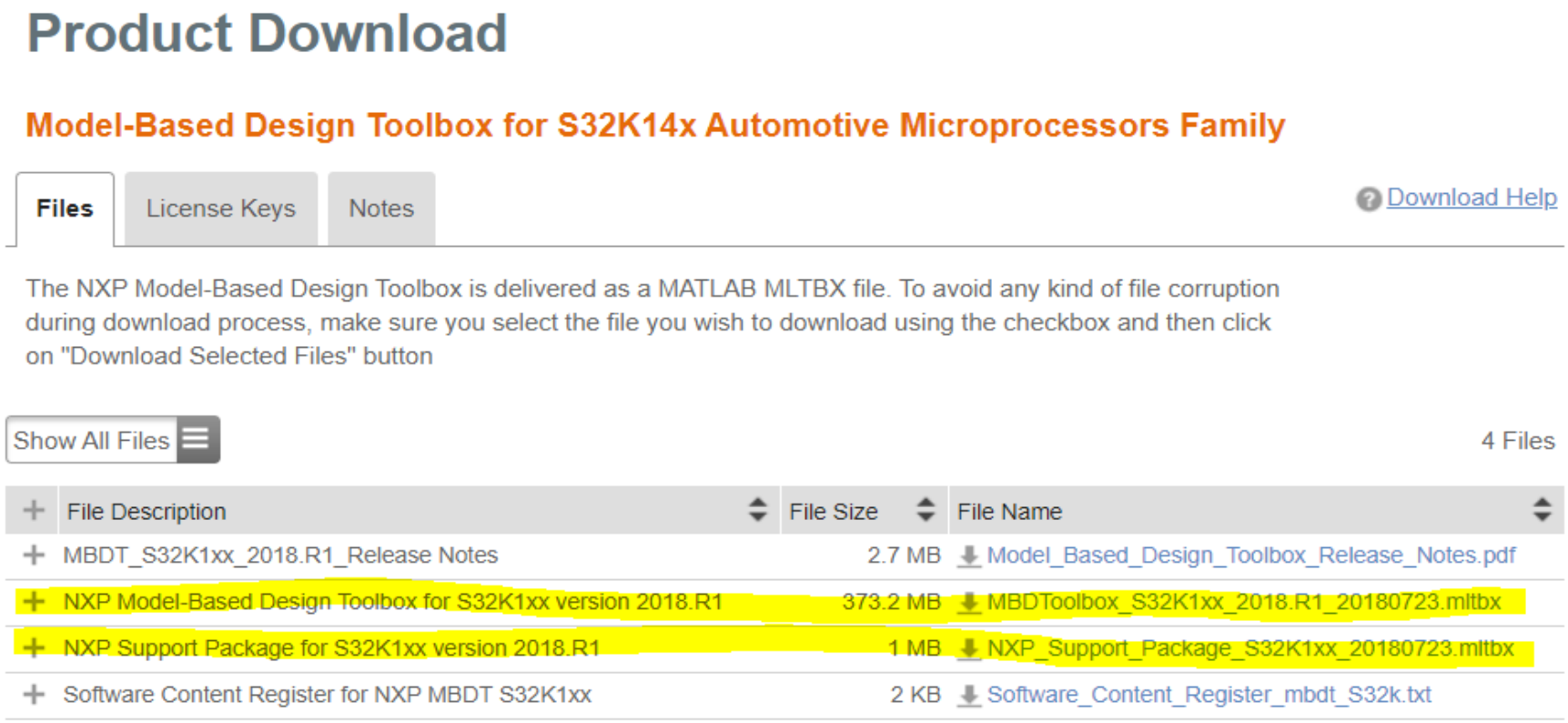
To register a New Product please click on the button below

Register

Current | **Previous**

Version	Description	
3.1.0	Model-Based Design Toolbox for MPC57xx Automotive Microprocessors Family	Download Log
4.1.0	Model-Based Design Toolbox for S32K1xx Automotive Microprocessors Family	Download Log
3.0.0	Model-Based Design Toolbox for MPC57xx Automotive Microprocessors Family	Download Log
4.0.0	Model-Based Design Toolbox for S32K14x Automotive Microprocessors Family	Download Log
2.0.0	Model-Based Design Toolbox for MATLAB/Simulink MBD supporting MPC574xP	Download Log
1.2.0	Motor Control Toolbox for MATLAB/Simulink MBD supporting MC9S12ZVMx	Download Log
1.1.0	Motor Control Toolbox for MATLAB/Simulink MBD supporting MPC574xP	Download Log
1.0.0	Motor Control Toolbox for MATLAB/Simulink MBD for Kinetis V series	Download Log
3.0.0	Model-Based Design Toolbox for MATLAB/Simulink MBD supporting S32K14x	Download Log
2.0.0	Model-Based Design Toolbox for MATLAB/Simulink MBD supporting S32K14x	Download Log
1.0.0	Motor Control Toolbox for MATLAB/Simulink MBD supporting S32K14x	Download Log

6. Download the two Highlighted MATLAB toolboxes.



Note: While downloading these your PC will tend to change the extension to “*.zip”; please correct them by making it “.mltbx”.

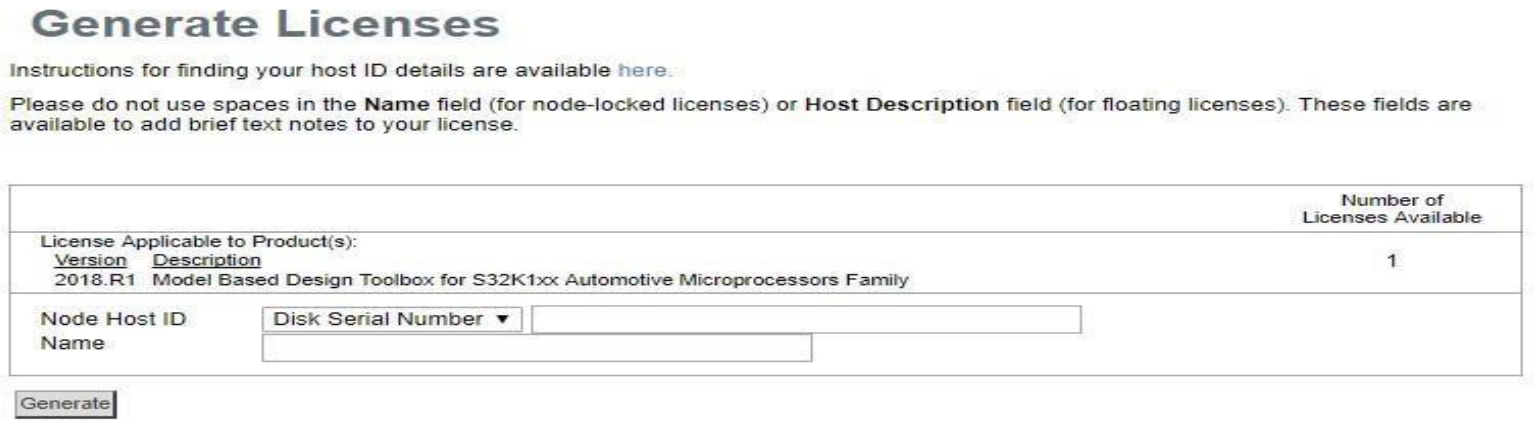
7. As downloading is in progress, go to the **Licence Keys** tab in the above image and we will generate license from there.

8. If Windows OS, Open command prompt and type the following.

vol C:

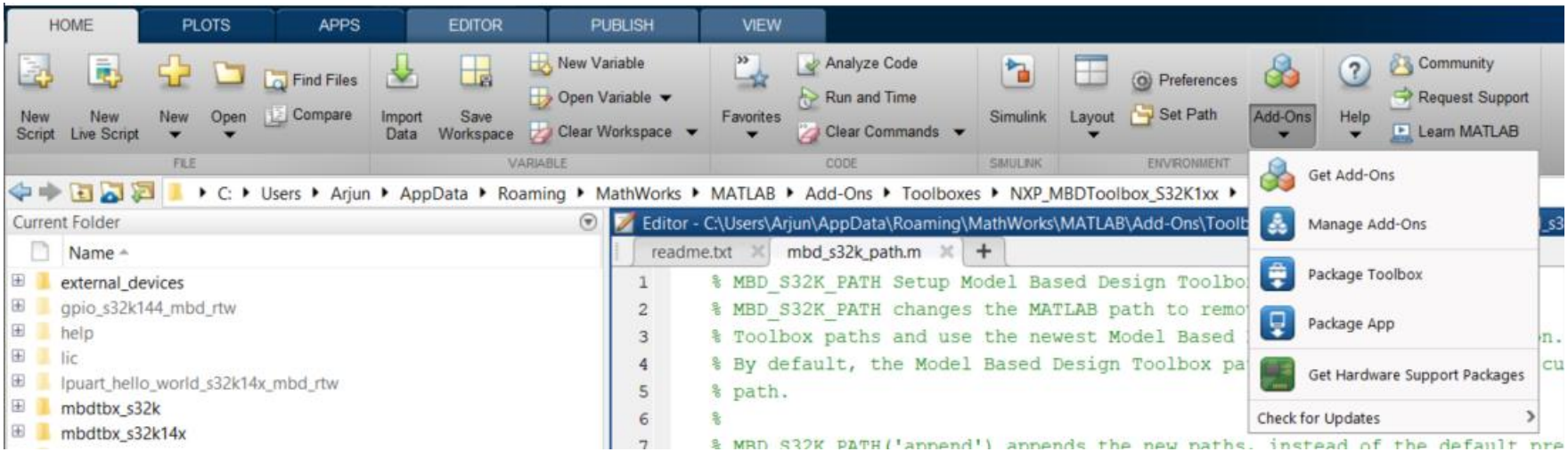
This will give you, your PC's **Disk Serial Number** you can use this to generate the license. After license is generated press **Save All** to download the license and save the file as "license.lic" to store them into your PC.

9.

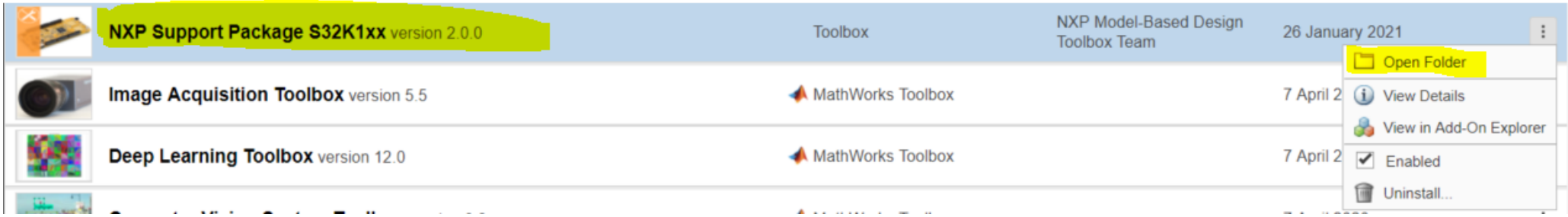


10. Open MATLAB 2018b and drag and drop this downloaded **NXP_Support_Package_S32K1xx_20180723.mltbx** to the command window. This will help you in a step-by-step guide to install the downloaded toolboxes.

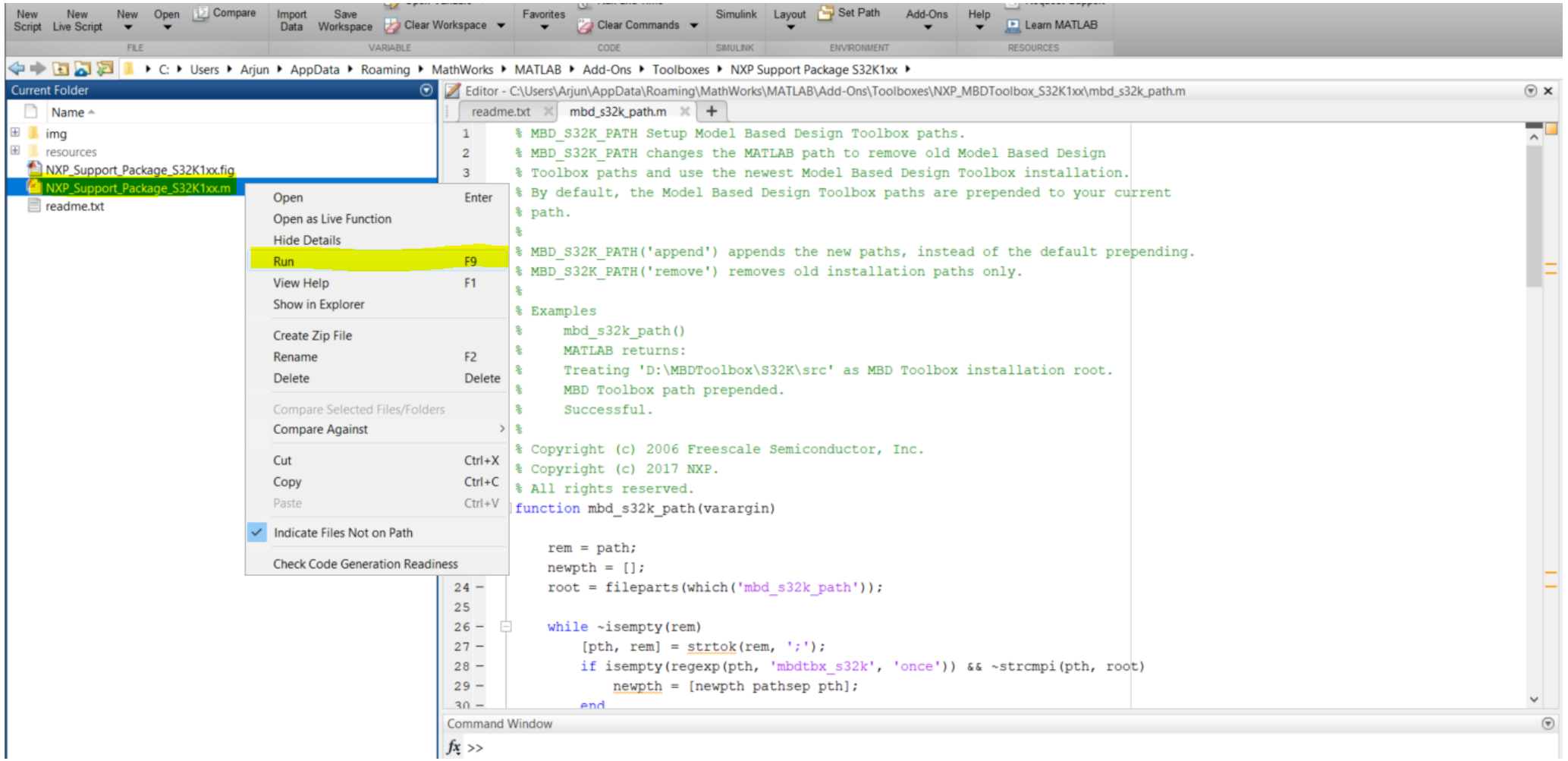
11. Open Addons dropdown options -> Manage Add-ons.



12. Select NXP Support package s32k1xx -> open folder as in the below figure:

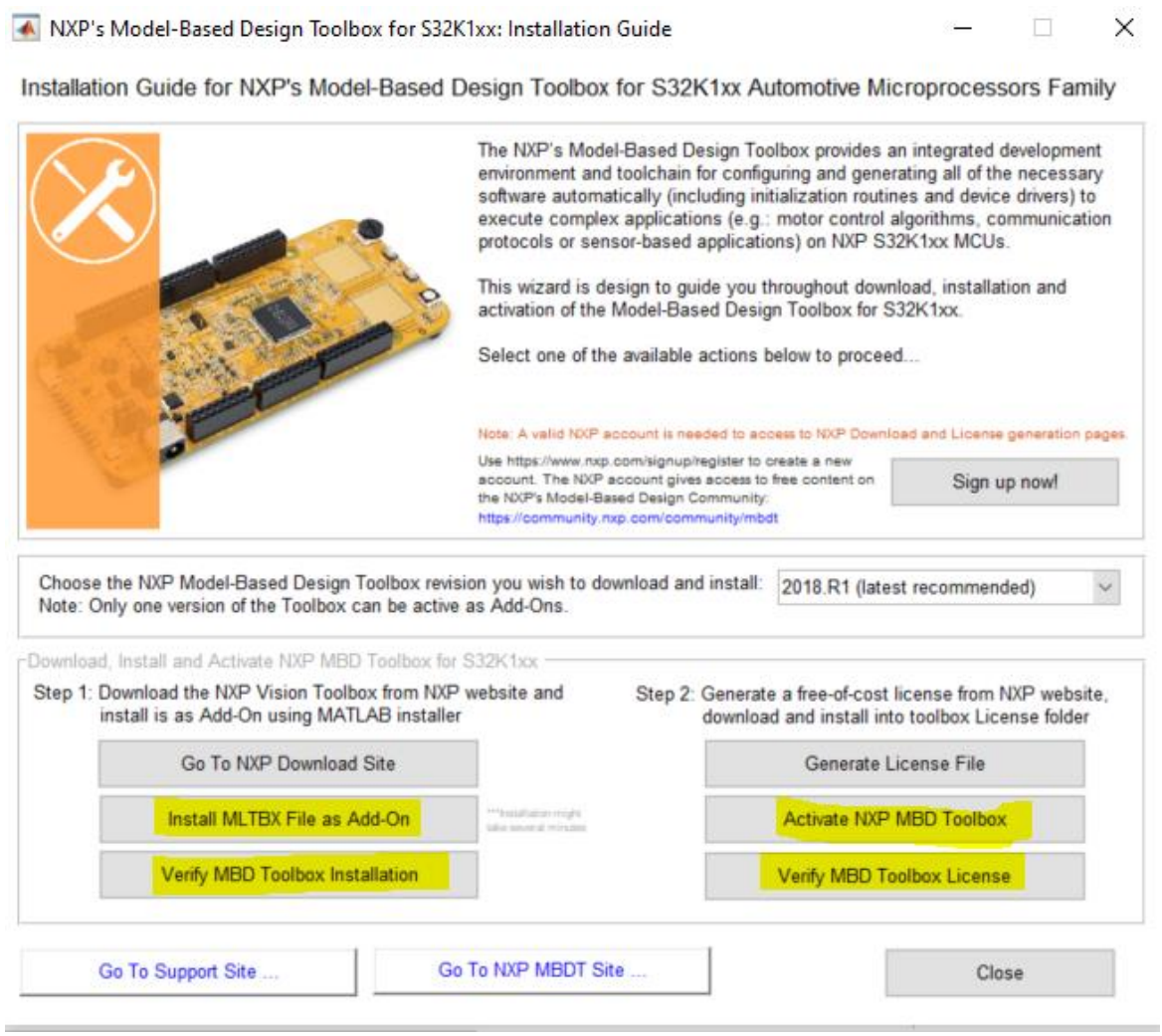


13. Run the NXP_Support_Package_S32K1xx.m as in the following figure:



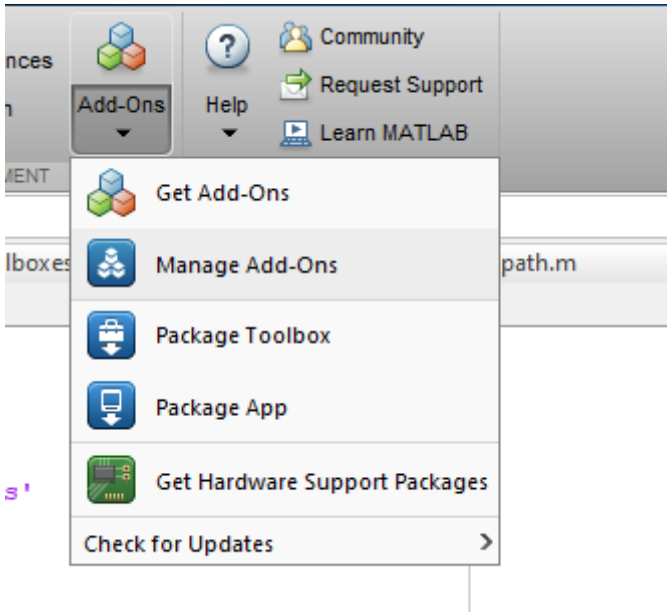
14. Complete All the highlighted steps by clicking on the buttons in the pop up that comes from the previous step. Please refer following image.

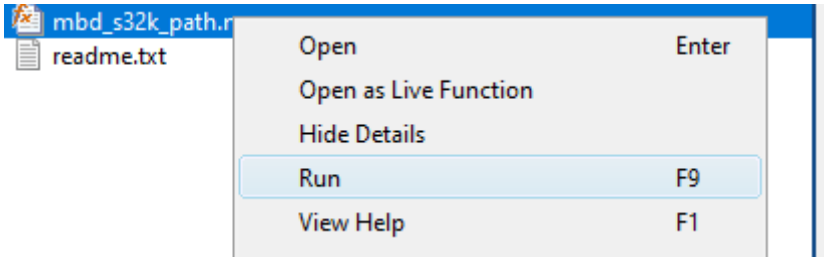
15. The MLTBX file is the second file that was downloaded, i.e, “M**BDToolbox_S32K1xx_2018.R1_20180723.mltbx**”



Setting the Path for MBDT in MATLAB

- Setting the Path for Model-Based Design Toolbox for MALAB to recognize the Model-Based Design Toolbox, the path needs to be setup in the MATLAB environment.
- Start MATLAB 2018b. Go to Addon -> Manage Addons.
- Select the options for NXP_MBDToolbox_S321xx, and click Open Folder
- Run the "mbd_s32k_path.m" path script. Right Click -> Run.





- After running it successfully, the following message should be seen in
  the command window.

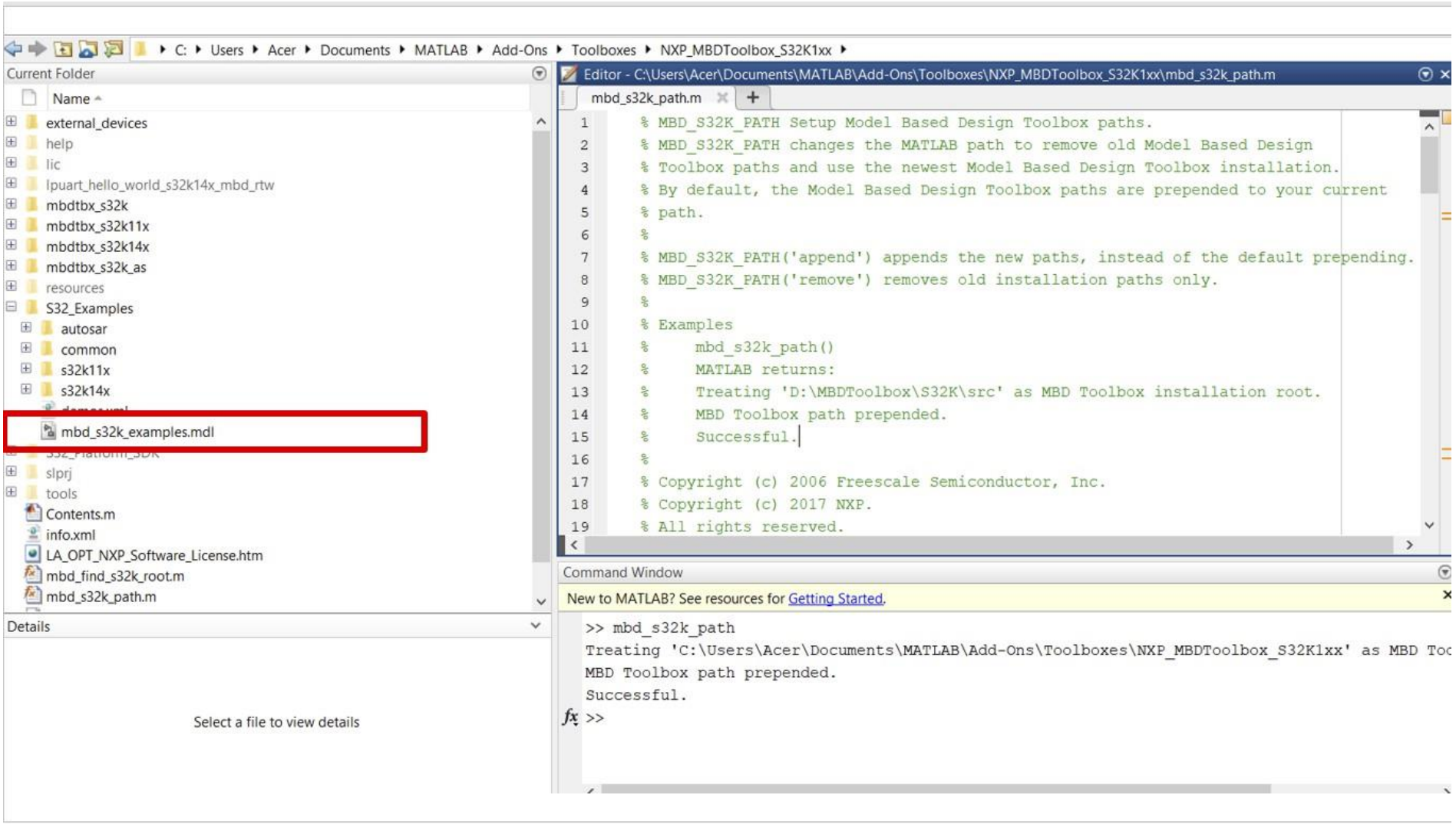
```
mbd_s32k_path
Treating 'C:\MBDToolbox\mbdtbx_S32K' as MBD Toolbox installation root.
MBD Toolbox path prepended.
Successful.
```

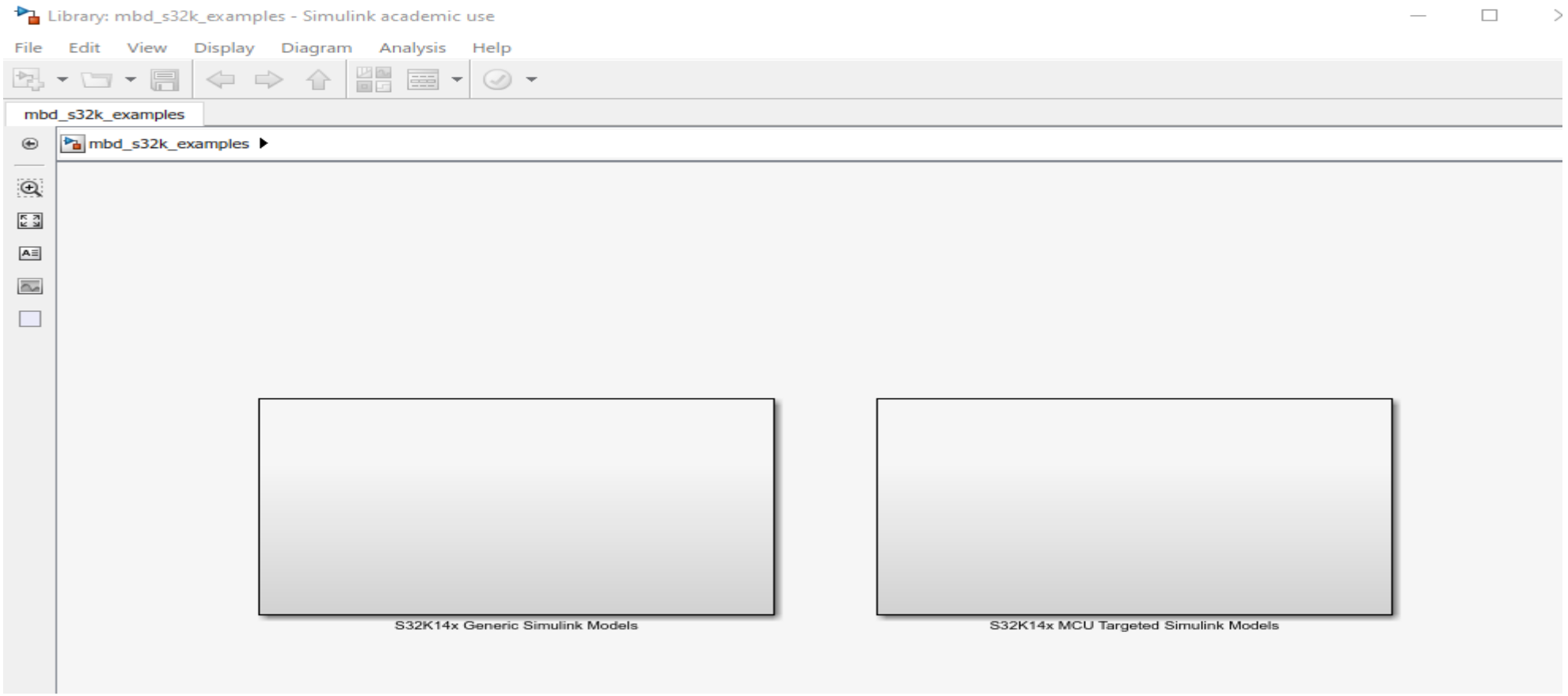
Run Models: Examples Library

- The Model Based Design Toolbox for S32K14x comes with an Examples Library collection that let you test different MCU on-chip modules and run complex applications.

- The Examples Library mbd_s32k_examplesmdl can be opened from “{Model Based Design Install Directory}\S32_Examples” folder

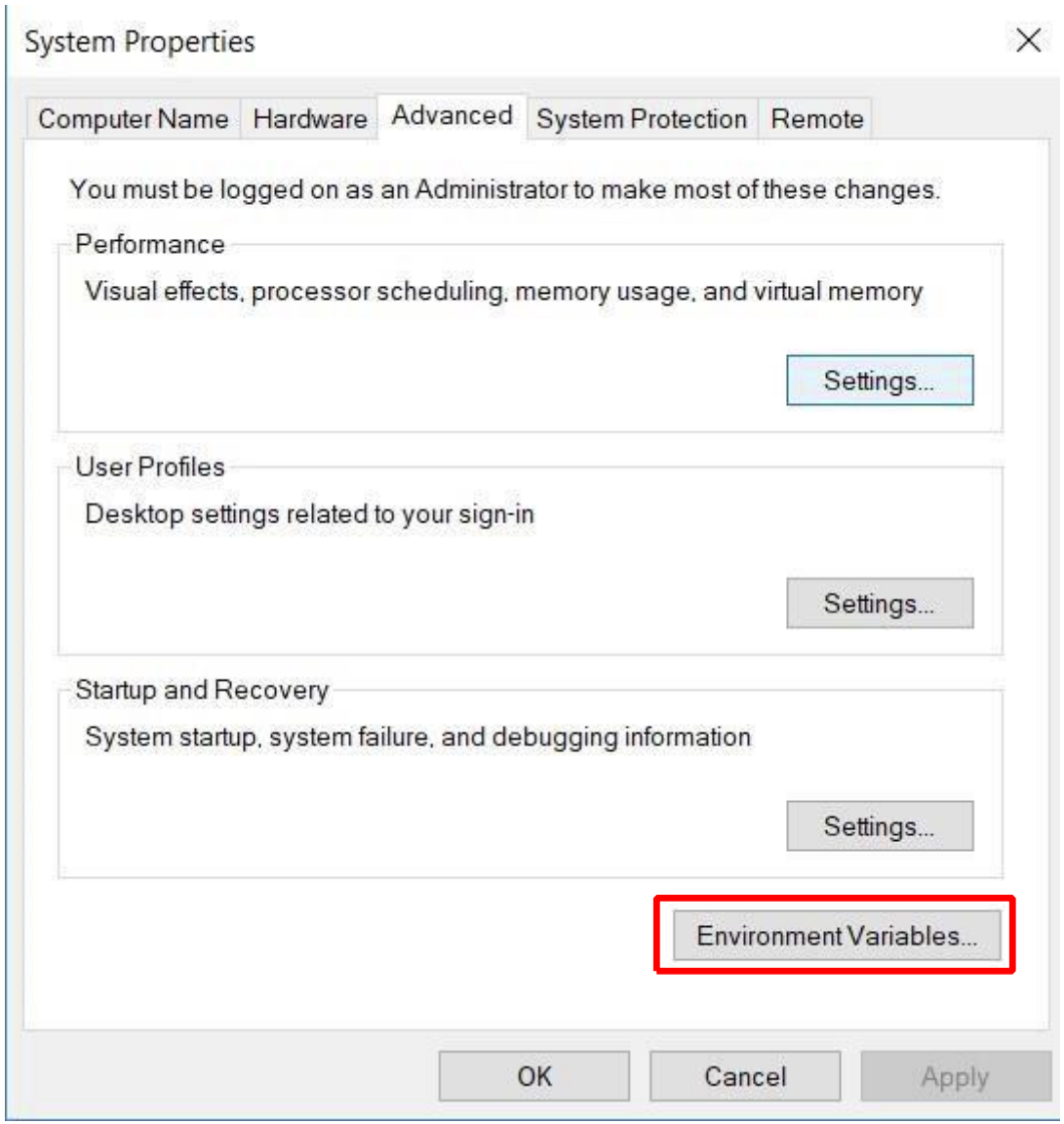
- Shown Below:

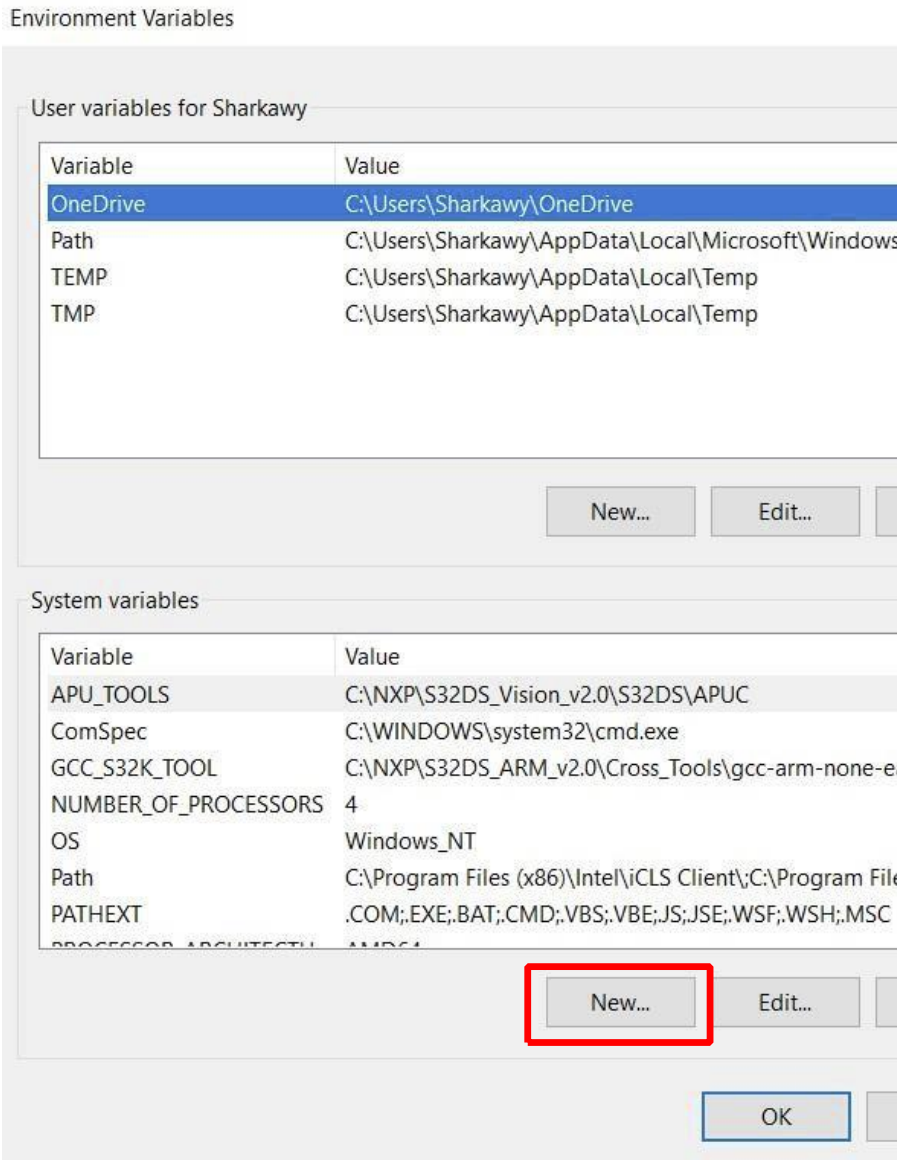

C: ▸ Users ▸ Acer ▸ Documents ▸ MATLAB ▸ Add-Ons ▸ Toolboxes ▸ NXP_MBDToolbox_S32K1xx ▸
Current Folder
Name
external_devices
help
lic
lpuart_hello_world_s32k14x_mbd_rtw
mbdtbx_s32k
mbdtbx_s32k11x
mbdtbx_s32k14x
mbdtbx_s32k_as
resources
S32_Examples
autosar
common
s32k11x
s32k14x
mbd_s32k_examples.mdl
slprj
tools
Contents.m
info.xml
LA_OPT_NXP_Software_License.htm
mbd_find_s32k_root.m
mbd_s32k_path.m
Details
Select a file to view details
Editor - C:\Users\Acer\Documents\MATLAB\Add-Ons\Toolboxes\NXP_MBDToolbox_S32K1xx\mbd_s32k_path.m
mbd_s32k_path.m
1 % MBD_S32K_PATH Setup Model Based Design Toolbox paths.
2 % MBD_S32K_PATH changes the MATLAB path to remove old Model Based Design
3 % Toolbox paths and use the newest Model Based Design Toolbox installation.
4 % By default, the Model Based Design Toolbox paths are prepended to your current
5 % path.
6 %
7 % MBD_S32K_PATH('append') appends the new paths, instead of the default prepending.
8 % MBD_S32K_PATH('remove') removes old installation paths only.
9 %
10 % Examples
11 % mbd_s32k_path()
12 % MATLAB returns:
13 % Treating 'D:\MBDToolbox\S32K\src' as MBD Toolbox installation root.
14 % MBD Toolbox path prepended.
15 % Successful.
16 %
17 % Copyright (c) 2006 Freescale Semiconductor, Inc.
18 % Copyright (c) 2017 NXP.
19 % All rights reserved.
Command Window
New to MATLAB? See resources for Getting Started.
>> mbd_s32k_path
Treating 'C:\Users\Acer\Documents\MATLAB\Add-Ons\Toolboxes\NXP_MBDToolbox_S32K1xx' as MBD Too
MBD Toolbox path prepended.
Successful.
>>



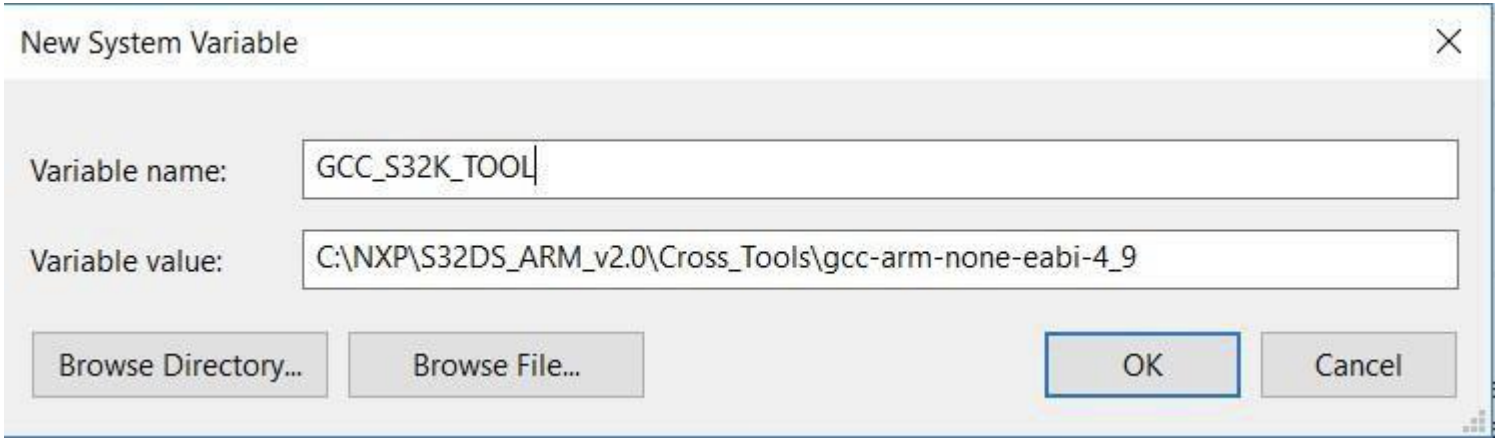
- Each category contains multiple examples that showcase different Model Based Design Toolbox example and blocks.

(Skip Until There's Error) You may need to add a New System variable





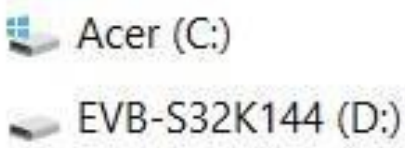
- Add the New System Variable and click OK.



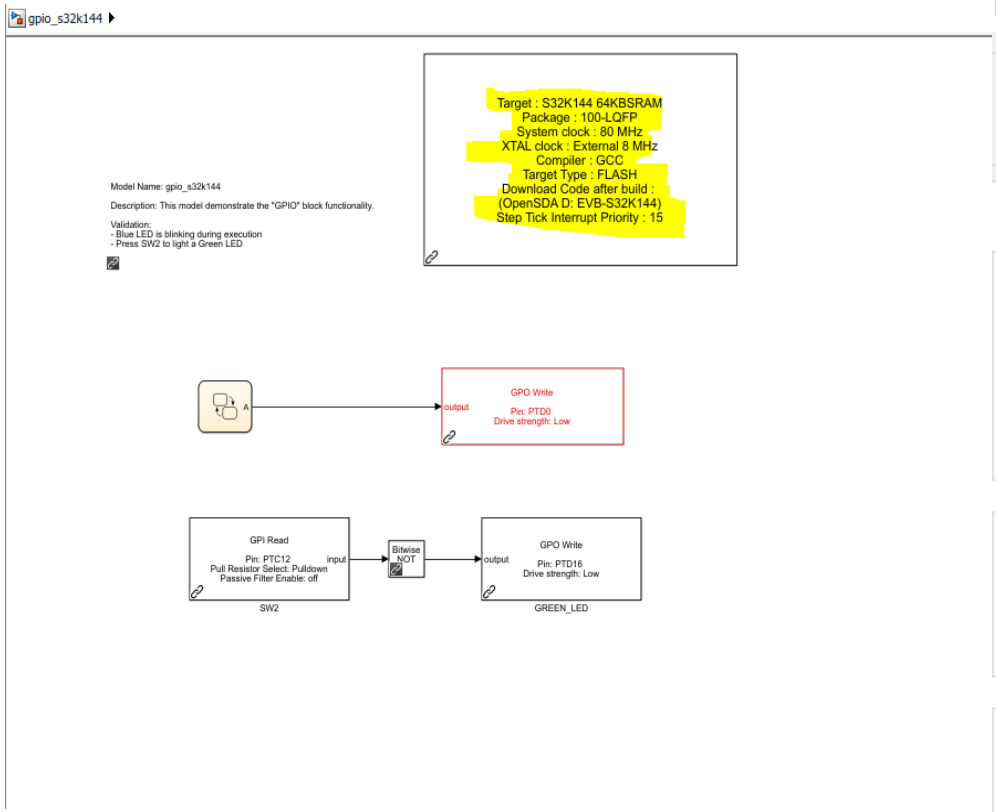
Hello World Example

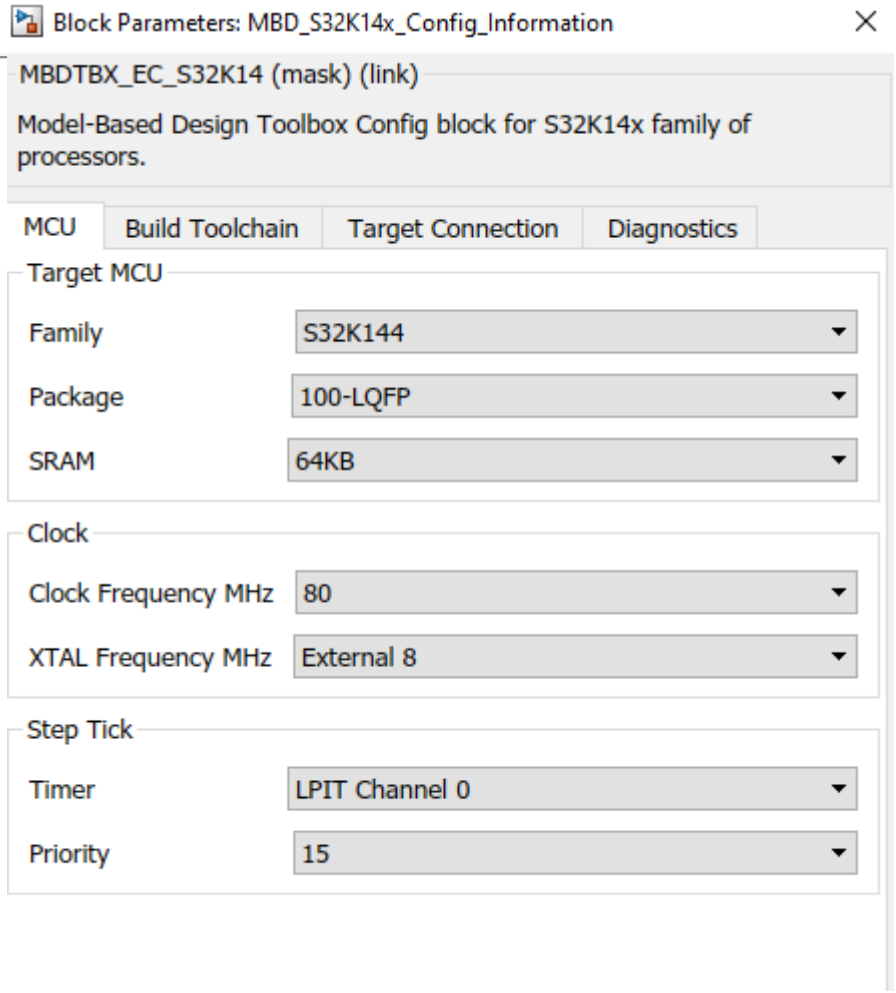
• Check that the virtual COM port is created and visible in Control
Panel -> Device Manager -> Port (COM & LPT)

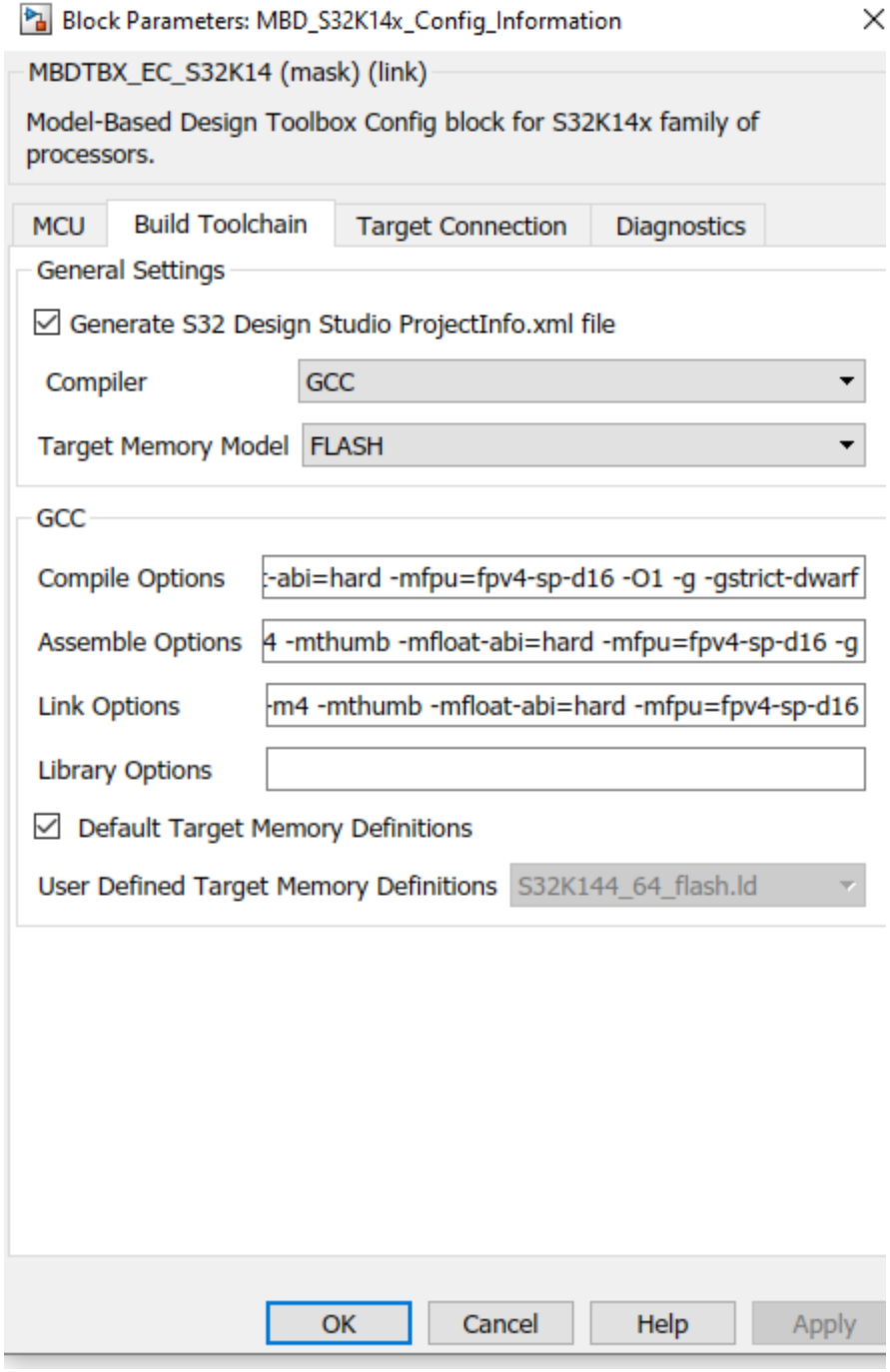
• Check that the virtual mass storage device is present.

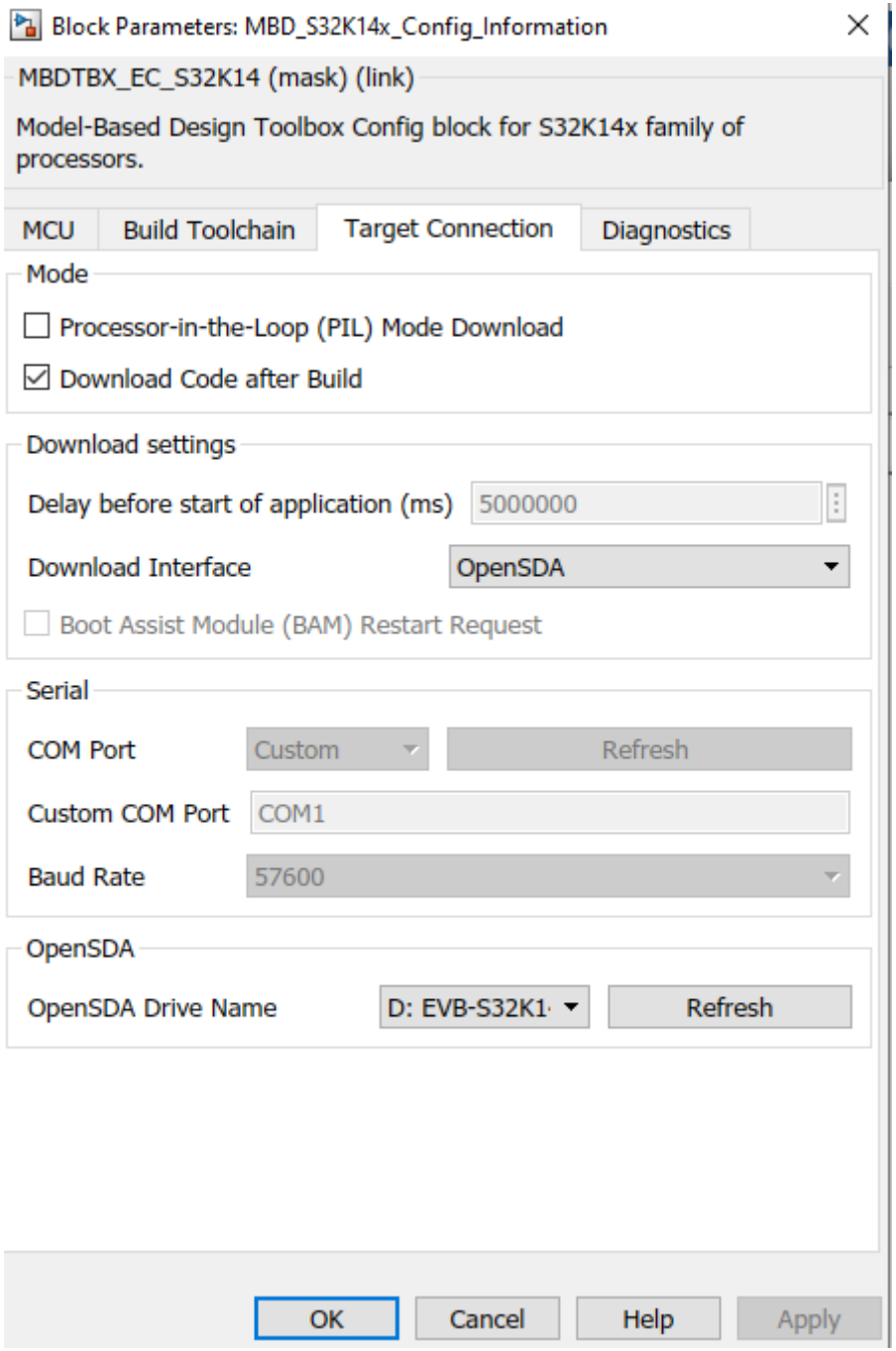


1. Open Matlab 2018b.
2. Open the mbd_s32k_examples.mdl from the previous sections.
3. In the Simulink model examples browser follow below steps.
4. Select **S32K14x MCU Targeted Simulink Models**.
5. Select **GPIO**, then select **GPIO on S32K144 LED& Buttons**.
6. This will load the hello world example.
7. Configure the model as highlighted in the below image, by double licking on the configuration block.


gpio_s32k144
Target : S32K144 64KBSRAM
Package : 100-LQFP
System clock : 80 MHz
XTAL clock : External 8 MHz
Compiler : GCC
Target Type : FLASH
Download Code after build :
(OpenSDA D: EVB-S32K144)
Step Tick Interrupt Priority : 15
Model Name: gpio_s32k144
Description: This model demonstrate the "GPIO" block functionality.
Validation:
- Blue LED is blinking during execution
- Press SW2 to light a Green LED
A
output
GPO Write
Pin: PTD0
Drive strength: Low
GPI Read
Pin: PTC12
Pull Resistor Select: Pulldown
Passive Filter Enable: off
input
SW2
Bitwise NOT
output
GPO Write
Pin: PTD16
Drive strength: Low
GREEN_LED


Block Parameters: MBD_S32K14x_Config_Information
MBDTBX_EC_S32K14 (mask) (link)
Model-Based Design Toolbox Config block for S32K14x family of processors.
MCU
Build Toolchain
Target Connection
Diagnostics
Target MCU
Family
S32K144
Package
100-LQFP
SRAM
64KB
Clock
Clock Frequency MHz
80
XTAL Frequency MHz
External 8
Step Tick
Timer
LPIT Channel 0
Priority
15


Block Parameters: MBD_S32K14x_Config_Information
MBDTBX_EC_S32K14 (mask) (link)
Model-Based Design Toolbox Config block for S32K14x family of processors.
MCU
Build Toolchain
Target Connection
Diagnostics
General Settings
Generate S32 Design Studio ProjectInfo.xml file
Compiler
GCC
Target Memory Model
FLASH
GCC
Compile Options
-abi=hard -mfpu=fpv4-sp-d16 -O1 -g -gstrict-dwarf
Assemble Options
4 -mthumb -mfloat-abi=hard -mfpu=fpv4-sp-d16 -g
Link Options
-m4 -mthumb -mfloat-abi=hard -mfpu=fpv4-sp-d16
Library Options
Default Target Memory Definitions
User Defined Target Memory Definitions
S32K144_64_flash.ld
OK
Cancel
Help
Apply


Block Parameters: MBD_S32K14x_Config_Information
MBDTBX_EC_S32K14 (mask) (link)
Model-Based Design Toolbox Config block for S32K14x family of processors.
MCU
Build Toolchain
Target Connection
Diagnostics
Mode
Processor-in-the-Loop (PIL) Mode Download
Download Code after Build
Download settings
Delay before start of application (ms)
5000000
Download Interface
OpenSDA
Boot Assist Module (BAM) Restart Request
Serial
COM Port
Custom
Refresh
Custom COM Port
COM1
Baud Rate
57600
OpenSDA
OpenSDA Drive Name
D: EVB-S32K1
Refresh
OK
Cancel
Help
Apply

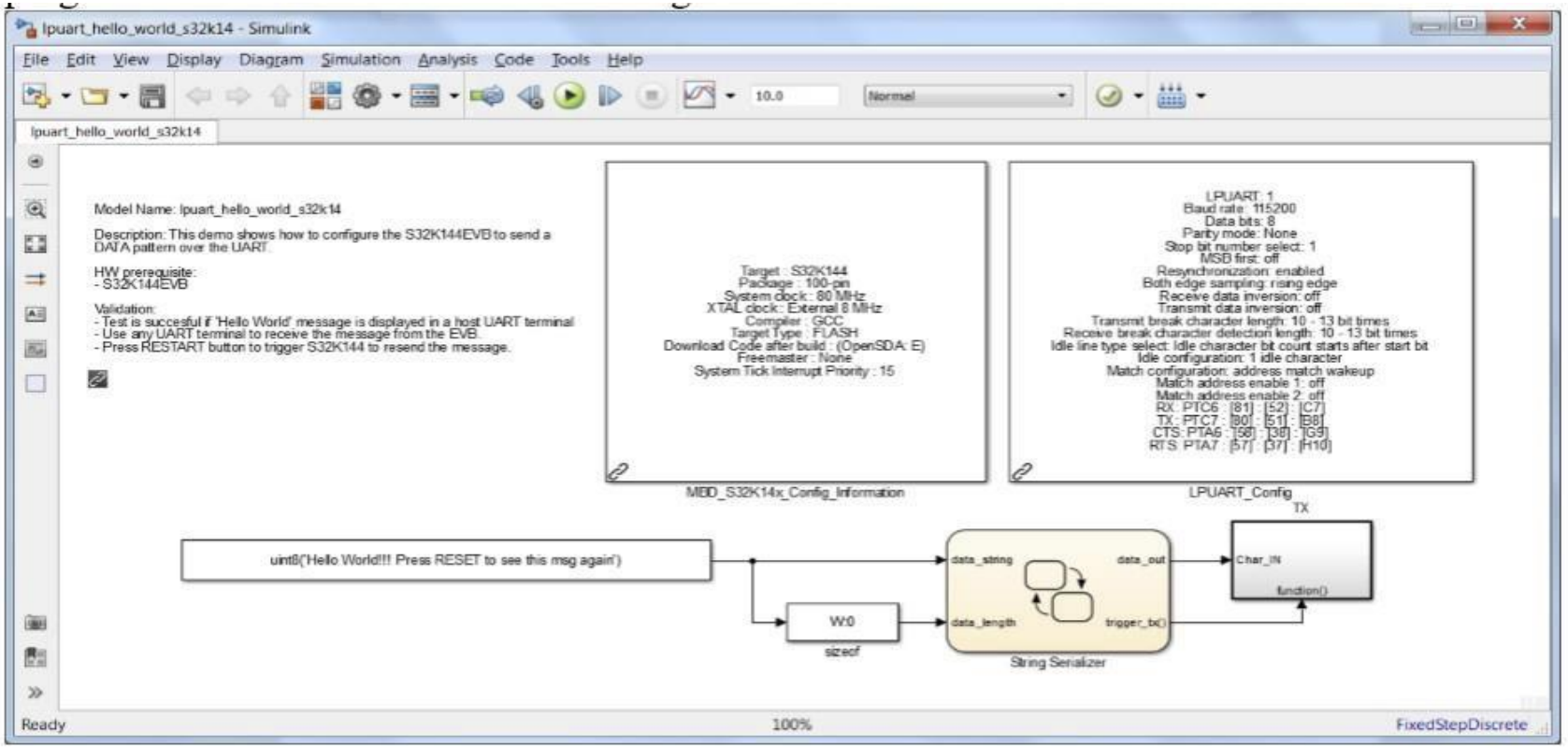
8. Connect your S32K144 EVB to the PC and Build the model using the  Build Icon.

9. This will download the code to the board after the build.

UART Example

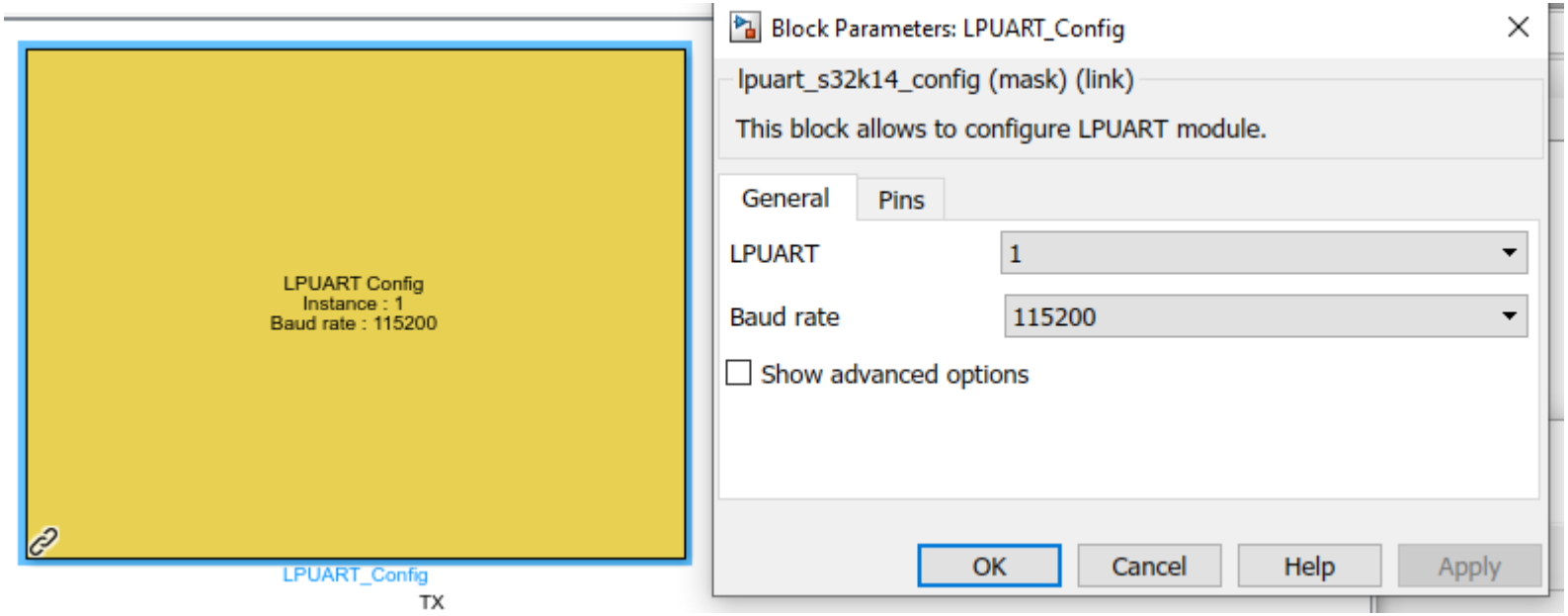
1. Open Matlab 2018b.
2. Open the mbd_s32k_examples.mdl from the previous sections.
3. In the Simulink model examples browser follow below steps.
4. Select **S32K14x Generic Simulink Models**.
5. Select **Communications**, then select **UART Hello World**.
6. This will load the UART hello world example.

7. Configure the model as highlighted in the below image, by double licking on the configuration block.



8. Configure the Target Configuration same as in the hello world example.

9. Configure LPUART Config as the following.



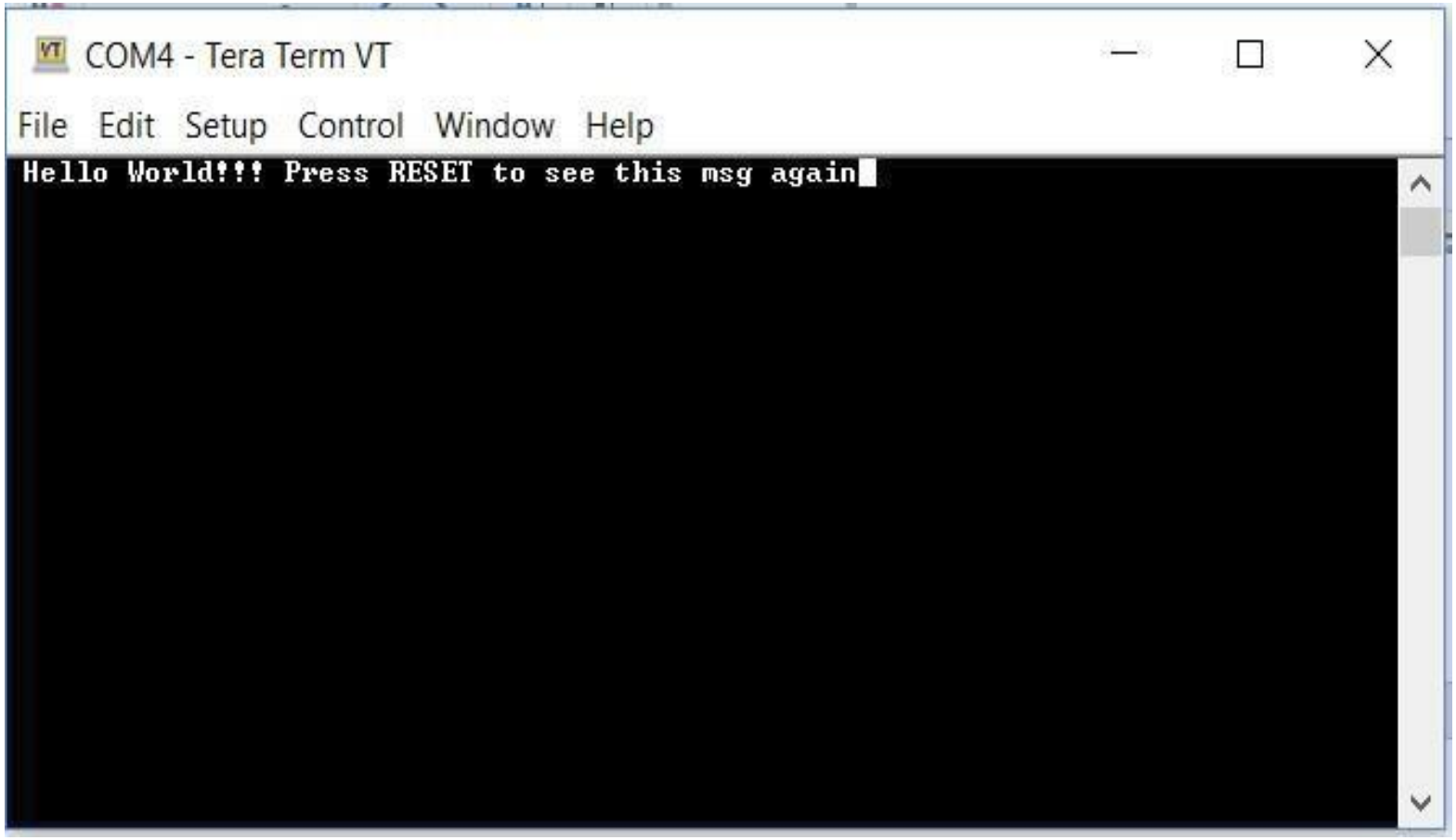
10. Build and download the code the S32K144 Board.

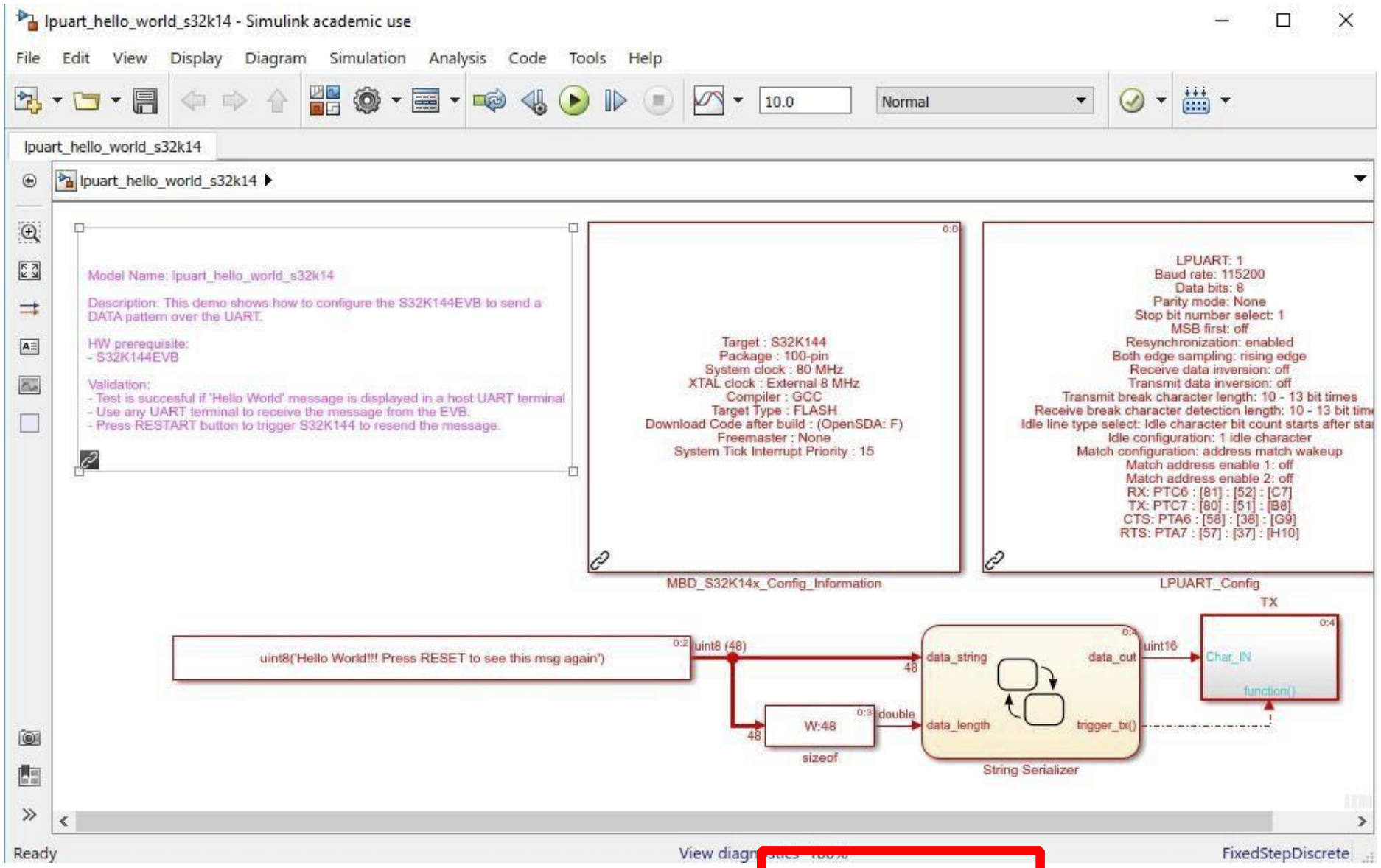
11. Open any UART terminal for the virtual COM port assigned and set up the

baud rate at 115200, data bits 8 and parity none. Putty or Tera-Term can be used.

12. Press the reset button on the evaluation board.

13. The S32K MCU sends “Hello World!!! Press RESET to see this msg again”
    message over the UART and the UART terminal should display it.


COM4 - Tera Term VT
File Edit Setup Control Window Help
Hello World!!! Press RESET to see this msg again


lpuart_hello_world_s32k14 - Simulink academic use
File Edit View Display Diagram Simulation Analysis Code Tools Help
10.0
Normal
lpuart_hello_world_s32k14
Model Name: lpuart_hello_world_s32k14
Description: This demo shows how to configure the S32K144EVB to send a DATA pattern over the UART.
HW prerequisite:
- S32K144EVB
Validation:
- Test is succesful if 'Hello World' message is displayed in a host UART terminal
- Use any UART terminal to receive the message from the EVB.
- Press RESTART button to trigger S32K144 to resend the message.
Target : S32K144
Package : 100-pin
System clock : 80 MHz
XTAL clock : External 8 MHz
Compiler : GCC
Target Type : FLASH
Download Code after build : (OpenSDA: F)
Freemaster : None
System Tick Interrupt Priority : 15
MBD_S32K14x_Config_Information
LPUART: 1
Baud rate: 115200
Data bits: 8
Parity mode: None
Stop bit number select: 1
MSB first: off
Resynchronization: enabled
Both edge sampling: rising edge
Receive data inversion: off
Transmit data inversion: off
Transmit break character length: 10 - 13 bit times
Match configuration: address match wakeup
Match address enable 1: off
Match address enable 2: off
RX: PTC6 : [81] : [52] : [C7]
TX: PTC7 : [80] : [51] : [B8]
CTS: PTA6 : [58] : [38] : [G9]
RTS: PTA7 : [57] : [37] : [H10]
LPUART_Config
TX
uint8('Hello World!!! Press RESET to see this msg again')
uint8 (48)
48
W:48
double
sizeof
data_string
data_length
data_out
trigger_tx()
String Serializer
uint16
Char_IN
function()
Ready
FixedStepDiscrete

Note:

• Click on View diagnostics at the bottom of your project to find the name of the
generated executable file, for example:
“lpuart_hello_world_s32K14ṁot”

```
Created executable: lpuart_hello_world_s32k14.elf
Generating S-record...
"C:/NXP/S32DS_ARM_v1.3/Cross_Tools/gcc-arm-none-eabi-4_9/bin/arm-none-eabi-objcopy" -O srec
lpuart_hello_world_s32k14.elf lpuart_hello_world_s32k14.mot
Created S-record: lpuart_hello_world_s32k14.mot
Building target all
*** Created executable: lpuart_hello_world_s32k14.mot
### Successful completion of build procedure for model: lpuart_hello_world_s32k14
```

• Locate the “lpuart_hello_world_s32K14ṁot” at your directory

• You can copy and paste the file to your “EVB_S32K144” drive at any time to
run your project.

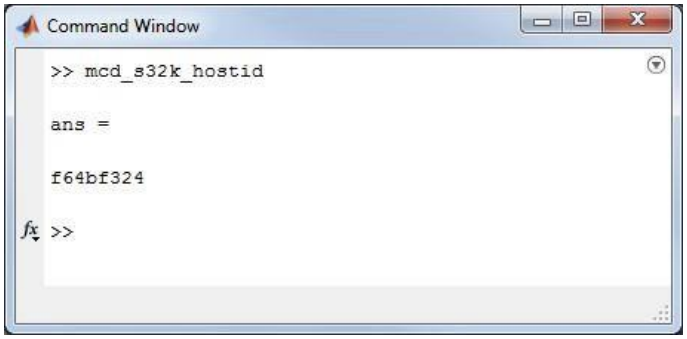
Appendix A

Locating the Host ID

If the Disk ID is used for the Host ID in the Model Based Design Toolbox software license, there are some different ways to obtain this:

A. From MATLAB Command

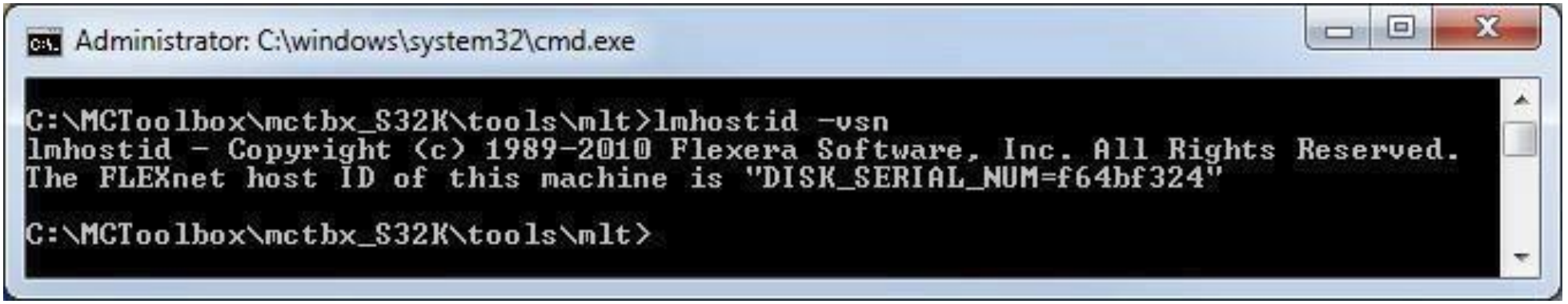
1. Open Matlab
2. In Command Window, enter “mbd_s32k_hostid”.
3. The hostid is the code returned.



In this example, Host ID is: f64bf324 (not case sensitive)

B. From DOS Command

1. Open CMD Prompt at {Model Based Design Toolbox installation folder}\mbdtbx_S32K\tools\mlt
2. In CMD Prompt window, enter “lmhostid -vsn”
3. Host ID is the value that follows DISK_SERIAL_NUM.



In this example, Host ID is: f64bf324 (not case sensitive)

Appendix B

Introduction to OpenSDA: 1 of 2

OpenSDA is an open-standard serial and debug adapter. It bridges serial and debug communications between a USB host ar embedded target processor. OpenSDA software includes a flash-resident USB mass-storage device (MSD) bootloader and a collection of OpenSDA Applications. S32K144 EVB comes with the MSD Flash Programmer OpenSDA Application preinstalle Follow these instructions to run the OpenSDA Bootloader and update or change the installed OpenSDA Application.

Enter OpenSDA Bootloader Mode

1. Unplug the USB cable if attached
2. Set J104 on position 1-2.
3. Press and hold the Reset button (SW5)
4. Plug in a USB cable (not included) between a USB host and the OpenSDA USB connector (labeled "SDA")
5. Release the Reset button

A removable drive should now be visible in the host file system with a volume label of BOOTLOADER. You are now in OpenSDA Bootloader mode.

IMPORTANT NOTE: Follow the "Load an OpenSDA Application" instructions to update the MSD Flash Programmer on your S32K144 EVB to the latest version.

Load an OpenSDA Application

1. While in OpenSDA Bootloader mode, double-click ***SDA_INFO.HTML*** in the ***BOOTLOADER*** drive. A web browser will open the OpenSDA homepage containing the name and version of the installed Application. This information can also be read as text directly from ***SDA_INFO.HTML***
2. Locate the ***OpenSDA Applications***
3. Copy & paste or drag & drop the MSD Flash Programmer Application *to the* ***BOOTLOADER*** *drive*
4. Unplug the USB cable and plug it in again. The new OpenSDA Application should now be running and a ***S32K144 EVB*** drive should be visible in the host file system

You are now running the latest version of the MSD Flash Programmer. Use this same procedure to load other OpenSDA Applications.

Introduction to OpenSDA: 2 of 2

The MSD Flash Programmer is a composite USB application that provides a virtual serial port and an easy and convenient wa program applications into the KEA MCU. It emulates a FAT16 file system, appearing as a removable drive in the host file syst volume label of EVB-S32K144. Raw binary and Motorola S-record files that are copied to the drive are programmed directly in flash of the KEA and executed automatically. The virtual serial port enumerates as a standard serial port device that can be o standard serial terminal applications.

Using the MSD Flash Programmer

1. Locate the .srec file of your project , file is under the Debug folder of the S32DS project.
2. Copy & paste or drag & drop one of the .srec files to the EVB-S32K144 drive

The new application should now be running on the S32K144 EVB. Starting with v1.03 of the MSD Flash Programmer, you can program repeatedly without the need to unplug and reattach the USB cable before reprogramming.

Drag one of the .srec code for the S32K144 the S32K144 EVB board over USB to reprogram the preloaded code example to another example.

NOTE: Flash programming with the MSD Flash Programmer is currently only supported on Windows operating systems. However, the virtual serial port has been successfully tested on Windows, Linux and Mac operating systems.

Using the Virtual Serial Port

1. Determine the symbolic name assigned to the EVB-S32K144 virtual serial port. In Windows open Device Manager and look for the COM port named "PEMicro/Freescale – CDC Serial Port".
2. Open the serial terminal emulation program of your choice. Examples for Windows include Tera Term, PuTTY, and HyperTerminal
3. Press and release the Reset button (SW0) at anytime to restart the example application. Resetting the embedded application will not affect the connection of the virtual serial port to the terminal program.
4. It is possible to debug and communicate with the serial port at the same time, no need to stop the debug.

NOTE: Refer to the OpenSDA User's Guide for a description of a known Windows issue when disconnecting a virtual serial port while the COM port is in use.