

Vision Toolbox for MATLAB

Quick Start Guide

**Embedded Target for the S32V234 Family of Automotive Vision Processors
Version 1.1.0**

Target Based Automatic Code Generation Tools

For MATLAB™ working with Mathworks Image Processing, Computer Vision and Deep Learning Toolboxes



Summary

1	Introduction	1-3
1.1	Purpose	1-3
1.2	Audience.....	1-3
1.3	Definitions, Acronyms and Abbreviations	1-3
2	Installation	2-4
2.1	System Requirements	2-4
2.2	MATLAB Required and Recommended Products.....	2-4
2.3	Mandatory Software	2-6
2.3.1	NXP Support Package for S32V234	2-6
2.3.2	NXP Vision Toolbox for S32V234	2-11
2.3.3	License Generation and Activation	2-16
2.3.4	Vision SDK and Build Tools.....	2-21
2.3.5	Setting up the Environment	2-23
2.4	Optional Software.....	2-25
2.4.1	SD Card Bootable Linux Image	2-25
3	Vision Application	3-26
3.1	Examples Library & Help	3-26
3.2	Face Detection in Simulation Mode.....	3-27
3.2.1	Running the Algorithm for Images	3-27
3.2.2	Running the Algorithm using Video Frames from PC Webcam.....	3-28
3.2.3	Running the Algorithm for Videos.....	3-30
3.3	Face Detection on S32V234 Vision Processor.....	3-32
3.3.1	Configure the microSD Card.....	3-32
3.3.2	S32V234 Evaluation Board Configuration	3-35
3.3.2.1	S32V234 EVB2 HW Setup.....	3-35
3.3.2.2	S32V234 SBC HW Setup	3-37
3.3.3	USB to UART connection.....	3-38
3.3.4	Compile and Run on S32V234.....	3-41

1 Introduction

This Quick Start Guide is designed to get you up and running in a matter of minutes with the concepts used by the NXP Vision Toolbox for S32V234 automotive vision processors. This toolbox is designed to be used from MATLAB in conjunction with the NXP 32V234 Vision SDK that support the Linux OS runtime environment.

The first part of this document covers the toolbox installation and setup of required prerequisites.

The second part then shows how to run a simple vision application in simulation and then on the real hardware evaluation board.

1.1 Purpose

The purpose of this document is to demonstrate how to install all the required software and run a vision application on NXP S32V234 automotive vision processors.

1.2 Audience

This document is intended to:

- MATLAB Computer Vision System users that wish to evaluate the NXP HW&SW solutions;
- NXP S32V234 users that need to have a quick start-up into vision applications and ready to run examples;

1.3 Definitions, Acronyms and Abbreviations

Acronym	Description
ACF	APEX Core Framework
APEX	A parallel image processing accelerator HW block part of NXP S32V234 SoC.
APEX COMPILER	Set of tools (NXP APU compiler) that allow compilation of code for APEX subsystem
ARM	Family of RISC architectures
SDK	Software Development Kit
ISP	Image Signal Processor

2 Installation

Installing the NXP Vision Toolbox for S32V234 is the first step in setting up and running automatic code generation from MATLAB for NXP S32V234 automotive vision processors and development boards.

The next sections present all the steps required to have the toolbox installed successful and ready for running the first application.

2.1 System Requirements

For a flowless development experience the minimum recommended PC platform is:

- Windows® 7/10 64bit Operating System
- At least 2 GHz CPU Speed
- At least 4 GB of RAM
- At least 20 GB of free disk space.
- Internet connectivity for web downloads.

2.2 MATLAB Required and Recommended Products

The NXP Vision Toolbox for S32V234 requires the following MathWorks products to be installed. Make sure you have a valid license for the products marked as “Required”

Product	Version Compatibility	Required or Recommended
MATLAB	R2018a or newer	Required
MATLAB Coder	R2018a or newer	Required
Embedded Coder	R2018a or newer	Required
Image Processing Toolbox	R2018a or newer	Required
Computer Vision System Toolbox	R2018a or newer	Required
Embedded Coder Support Package for ARM Cortex-A Processors	R2018a or newer	Required
Computer Vision System Toolbox OpenCV Interface	R2018a or newer	Required
Deep Learning Toolbox	R2018a or newer	Required for CNN code generation
MATLAB Support Package for USB Webcams	R2018a or newer	Recommended for webcam usage in simulations

Image Acquisition Toolbox Support Package for OS Generic Video Interface	R2018a or newer	Recommended for webcam usage in simulations
Deep Learning Toolbox™ Model for SqueezeNet Network	R2018a or newer	Recommended for CNN code generation
Deep Learning Toolbox™ Model for AlexNet Network	R2018a or newer	Recommended for CNN code generation
Deep Learning Toolbox™ Model for GoogLeNet Network	R2018a or newer	Recommended for CNN code generation
MATLAB Coder Interface for Deep Learning Libraries Deep Learning Toolbox	R2018a or newer	Recommended for CNN code generation

Due to code generation performance issues the NXP Vision Toolbox uses a special feature `row-major` that has been introduced in [MATLAB Coder 2018a](#).

2.3 Mandatory Software

[NXP Vision Toolbox](#) is delivered as MATLAB Toolbox Package (MLTBX) that can be installed:

- Online from MathWorks File Exchange [website](#). For convenience, a NXP Support Package for S32V234 is available to assist throughout the installation process of the NXP Vision Toolbox and supplementary software;
- Offline from NXP [website](#) as a MATLAB Add-on;

This section shows how to install the NXP Vision Toolbox using online approach directly from MathWorks Add-ons File Exchange website. In case you have already downloaded the NXP Vision Toolbox for S32V234 MLTB file from NXP's official web page then jump directly to section 2.3.2 NXP Vision Toolbox for S32V234

To have the NXP Vision Toolbox installed and configured properly the following actions should be executed:

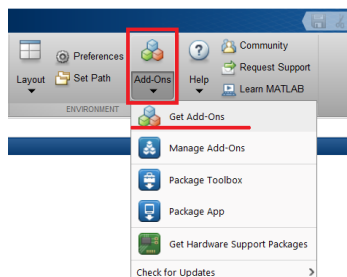
- Use Get Add-ons menu from MATLAB to search for “S32V Support Package” online and install the toolbox;
- Start the NXP Support Package for S32V234 and follow the steps indicated in the user interface;
- Download and install the NXP Vision Toolbox for S32V234 from NXP [website](#)
- Register and activate the NXP Vision Toolbox license
- Download and install the NXP Vision SDK package, including the cross-compilation tools for ARM and APEX cores
- Set the APU Compiler and Vision SDK Environment Variables

Each of these actions are explained in the following sub-chapters.

NOTE It is recommended to install all the software (MATLAB, NXP Toolboxes and NXP Vision SDK into system paths without spaces.

2.3.1 NXP Support Package for S32V234

For convenience a step-by-step installer guide is available on MathWorks's File Exchange [website](#). Open MATLAB and select Get Add-ons:

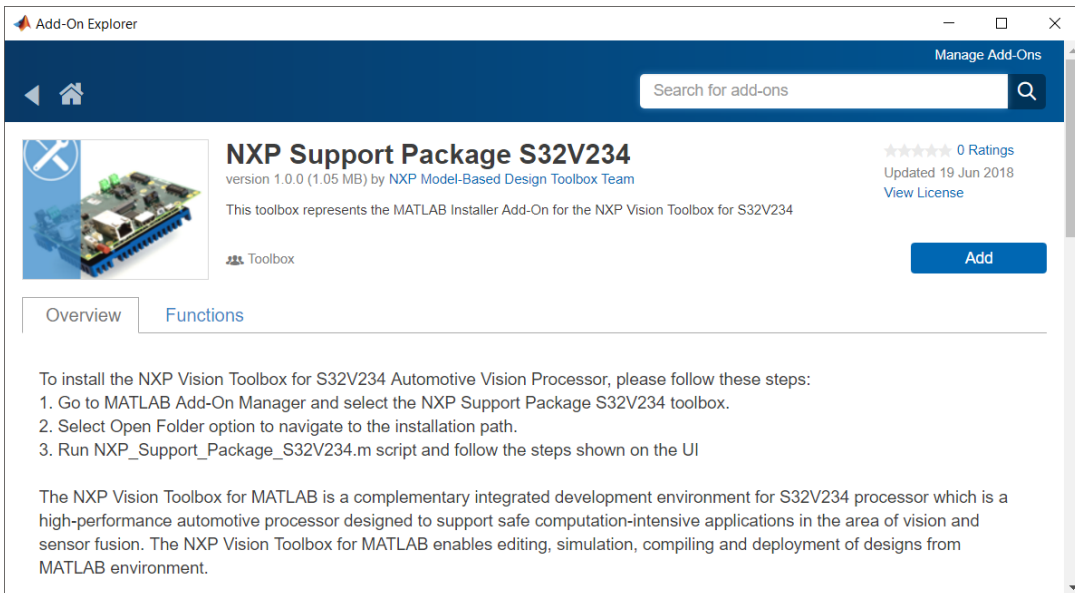


NOTE The screenshots below have been taken for an existing release of NXP Vision supporting S32V234 processor family since the final version was not yet published on the website at the time when this document was created. However, the entire process for Vision Toolbox revision 1.1.0 RFP will be identical with the one presented below.

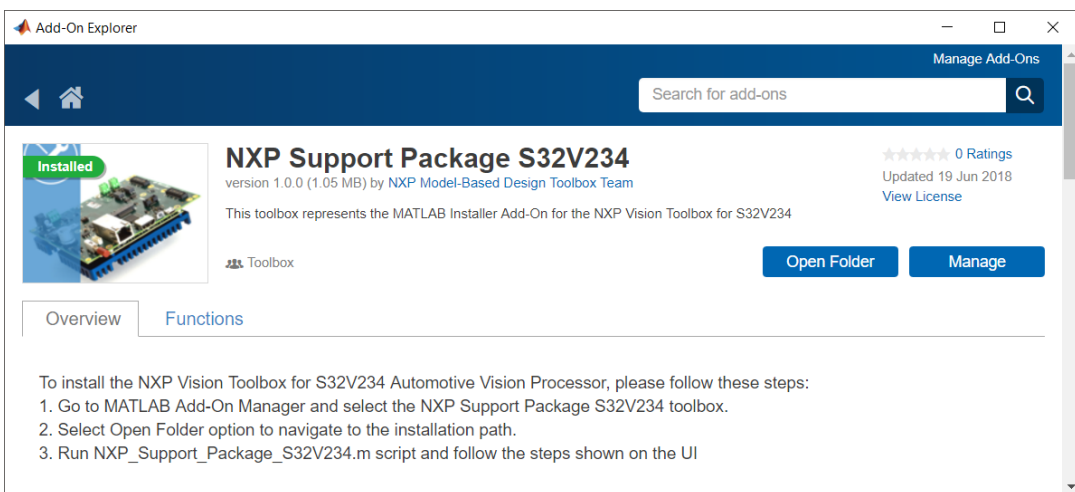
1. Once the Add-On Explorer window opens, search for “nxp vision toolbox s32v”



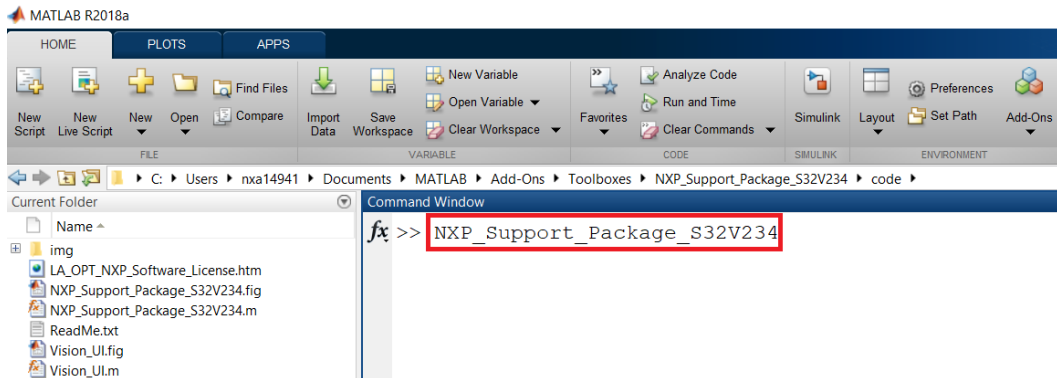
2. Select the NXP Support Package for S32V234 and click on Add button to start the installation of the installer guide into your MATLAB instance.



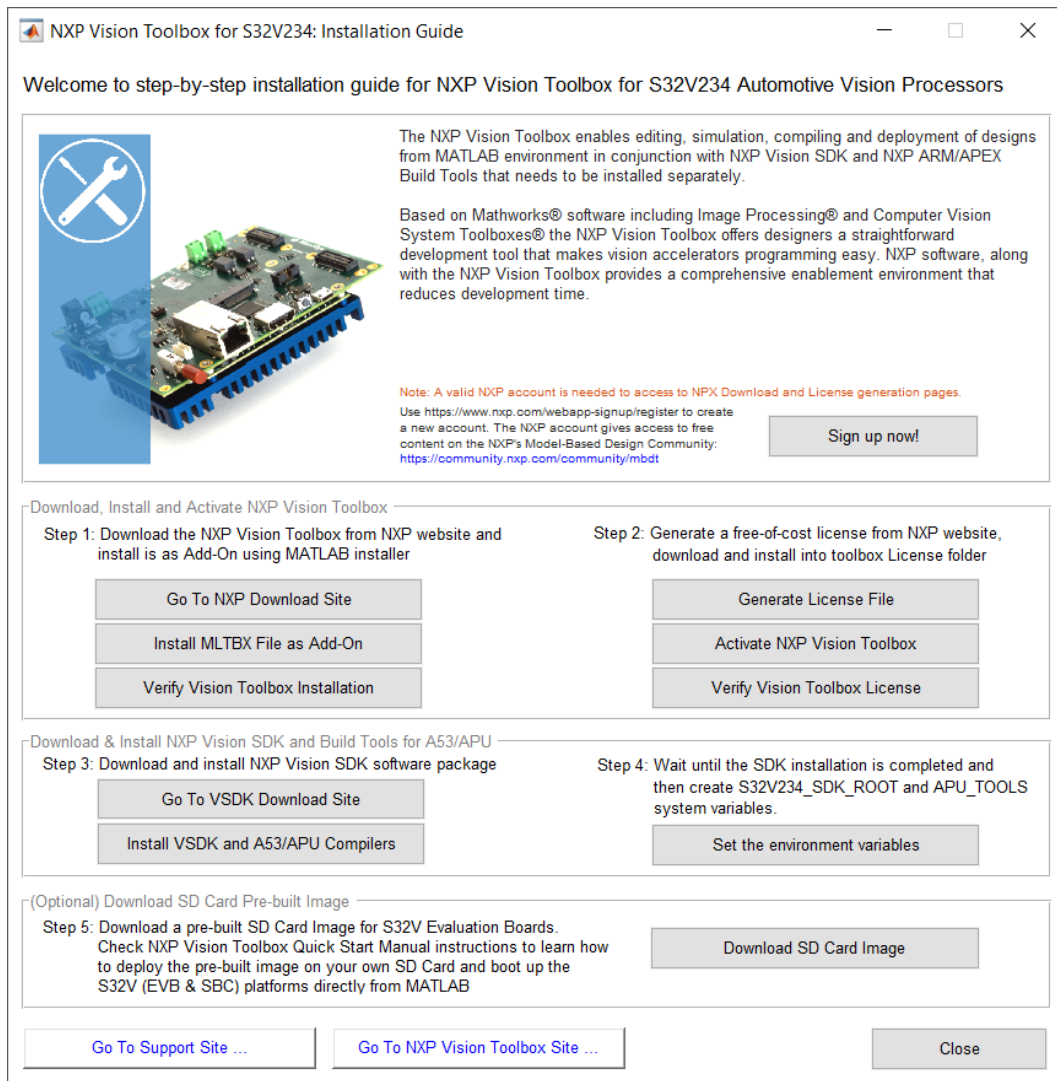
3. Wait until the toolbox is installed and then click on Open Folder button.



- Run the `NXP_Support_Package_S32V234` command in your MATLAB console to start the Installer Guide.



- The NXP Support Package for S32V234 - Installer Guide User Interface is started



The Installer Guide contains instructions for downloading, installing and verification of all software components required for being able to develop vision application with MATLAB for NXP S32V234 automotive vision processors:

- Steps to download, install and verification of the NXP Vision Toolbox for S32V234
- Steps to generate, activate and verification of the license for NXP Vision Toolbox for S32V234
- Steps to download and install NXP Vision SDK package
- Steps to configure the software environment for code generation
- Steps to download additional software

There are 2 main advantages of using this Installer Guide:

- Each step completion is automatically checked by the tool. If the action is completed successfully, then the tool is going to mark it as green. If a particular step cannot be verified, then the tool will issue a warning or error and is going to highlight in red that particular step that needs more attention for user side.
- Future updates will be made available via this online toolbox. In case you wish to keep your software up to date, then please install this into your MATLAB Add-ons and once a new update will be available your MATLAB instance will notify you.

The next screen capture shows how the Installer Guide notify user of successful or failed actions. At the end of installation all push buttons should be green.

NXP Vision Toolbox for S32V234: Installation Guide

Welcome to step-by-step installation guide for NXP Vision Toolbox for S32V234 Automotive Vision Processors

The NXP Vision Toolbox enables editing, simulation, compiling and deployment of designs from MATLAB environment in conjunction with NXP Vision SDK and NXP ARM/APEX Build Tools that needs to be installed separately.

Based on Mathworks® software including Image Processing® and Computer Vision System Toolboxes® the NXP Vision Toolbox offers designers a straightforward development tool that makes vision accelerators programming easy. NXP software, along with the NXP Vision Toolbox provides a comprehensive enablement environment that reduces development time.

Warning
NXP Vision Toolbox for S32V234 was not found!
Please check Add-Ons/Manage Add-Ons menu to verify if this toolbox exists

Step 1: Download, Install and Activate NXP Vision Toolbox

Step 1: Download the NXP Vision Toolbox from NXP website and install is as Add-On using MATLAB installer

Go To NXP Download Site
Install MLTBX File as Add-On
Verify Vision Toolbox Installation

Step 2: Generate a free-of-cost license from NXP website, download and install into toolbox License folder

Generate License File
Activate NXP Vision Toolbox
Verify Vision Toolbox License

Step 3: Download and install NXP Vision SDK software package

Go To VSDK Download Site
Install VSDK and A53/APU Compilers

Step 4: Wait until the SDK installation is completed and then create S32V234_SDK_ROOT and APU_TOOLS system variables.

Set the environment variables

(Optional) Download SD Card Pre-built Image

Step 5: Download a pre-built SD Card Image for S32V Evaluation Boards. Check NXP Vision Toolbox Quick Start Manual instructions to learn how to deploy the pre-built image on your own SD Card and boot up the S32V (EVB & SBC) platforms directly from MATLAB

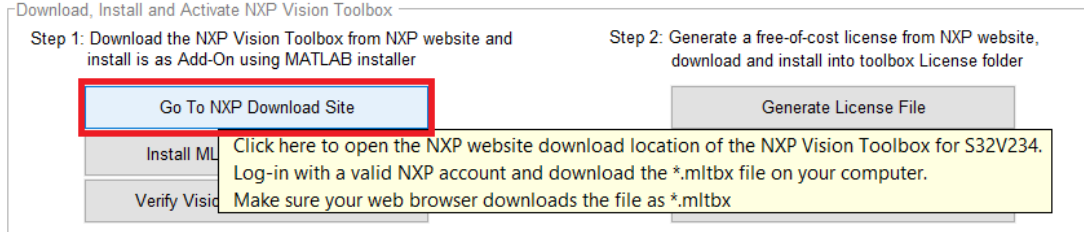
Download SD Card Image

Go To Support Site ... Go To NXP Vision Toolbox Site ... Close

2.3.2 NXP Vision Toolbox for S32V234

You can obtain the NXP Vision Toolbox for S32V234 by:

- Using the Installer guide “Go To NXP Download Site” button

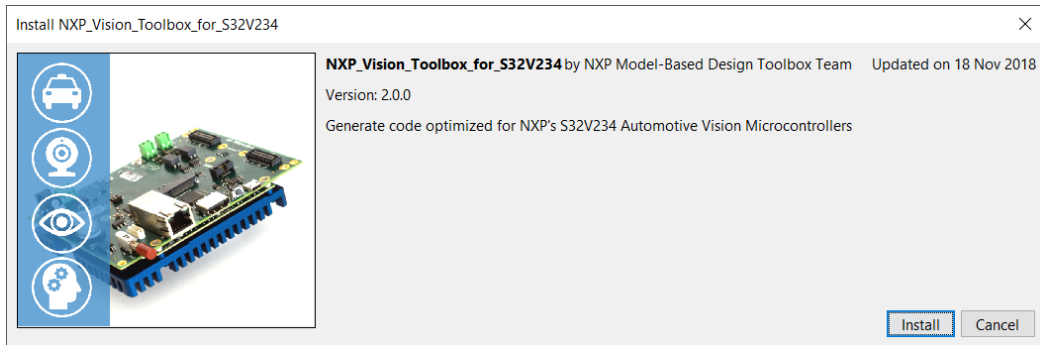


- Go directly into your NXP Software Account and download the toolbox using this [link](#)

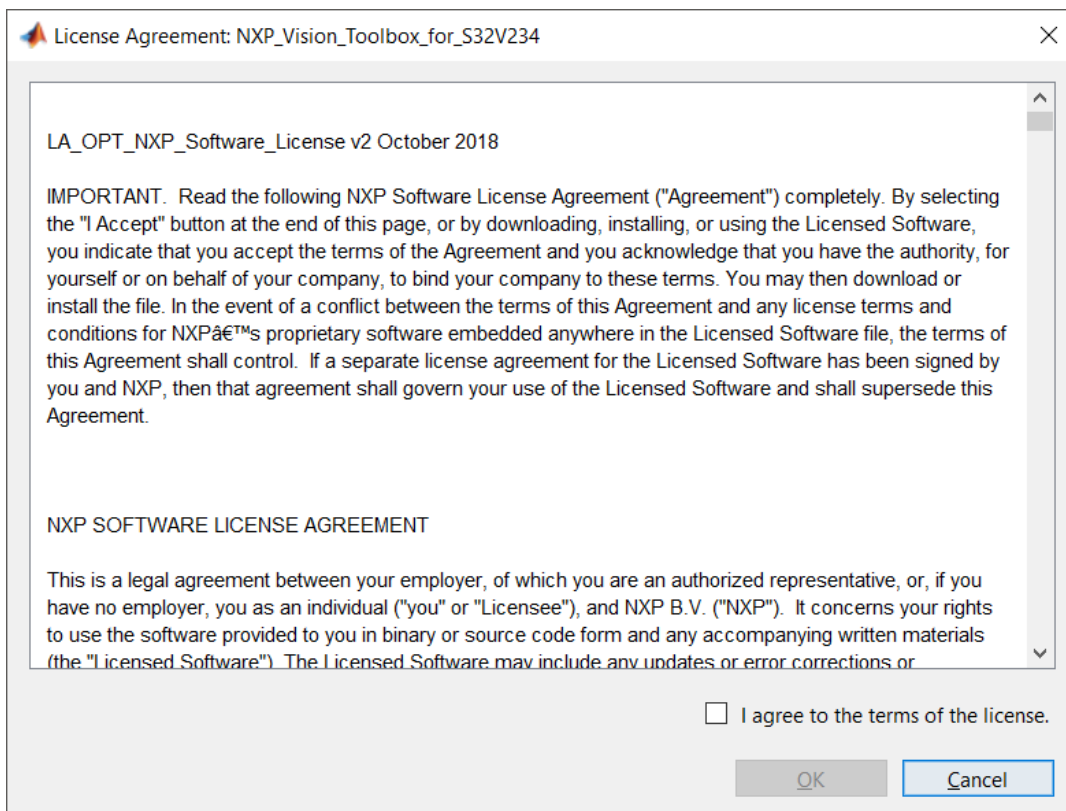
No matter which option is used, the NXP Vision Toolbox for S32V234 installation steps are similar: once you have the toolbox on your PC, double click on the *.mltbx file to start the MATLAB Add-ons installer that will automatically start the installation process.

You will be prompted with the following options:

1. The NXP’s Vision Toolbox Installation Wizard dialog will appear. Click “Install” to proceed.

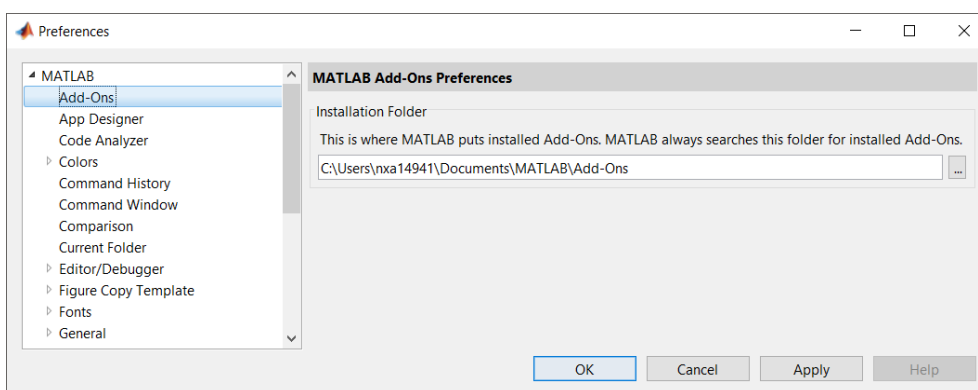


2. Indicate acceptance of the NXP Software License Agreement by selecting “I agree to the terms of the license” to proceed.

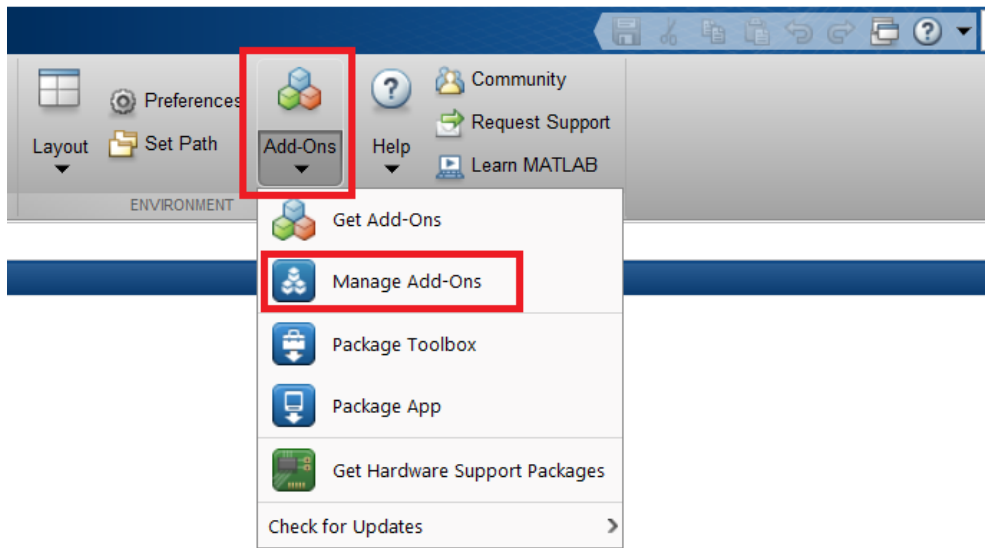




3. Click “OK” to start the MATLAB installation process. The rest of the process is silent and under MATLAB control. All the files will be automatically copied into default Add-Ons folder within the MATLAB

The default location can be changed prior to installation by changing the Add-Ons path from MATLAB Preferences

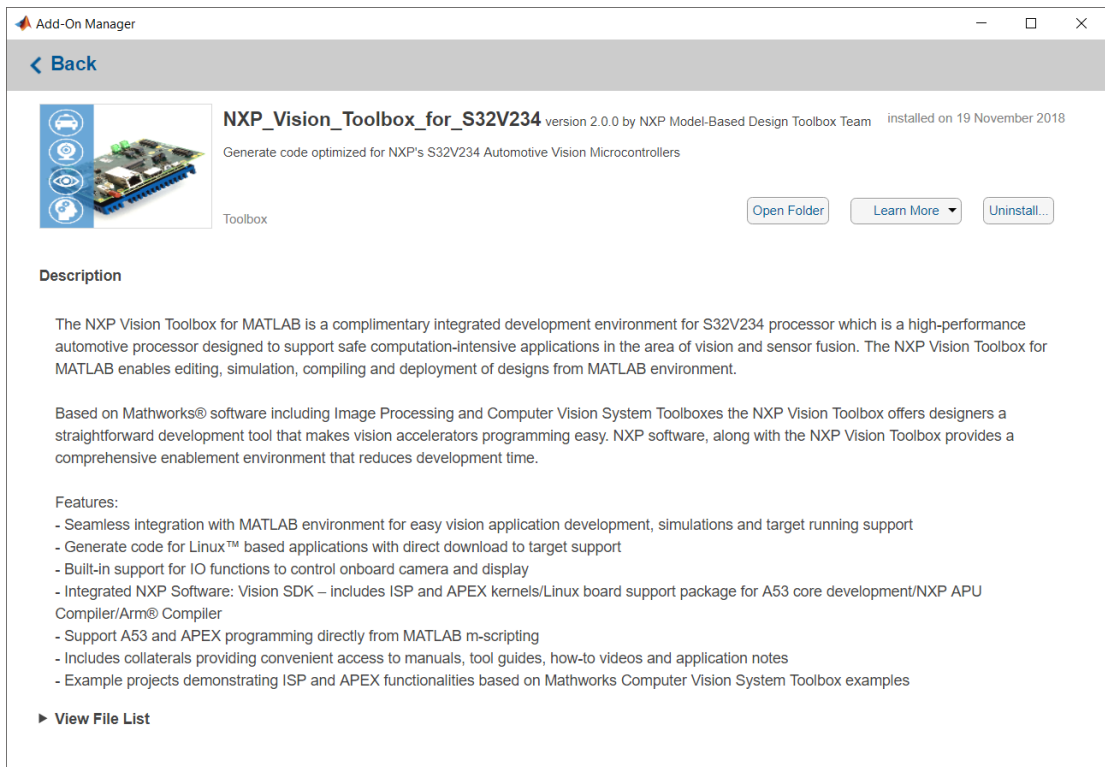


4. After a couple of seconds, the NXP's Vision Toolbox should be visible as a new Add-ons.

A screenshot of the 'Add-On Manager' window. It shows a list of installed add-ons. The first entry, 'NXP_Vision_Toolbox_for_S32V234 version 2.0.0', is highlighted with a red box. The second entry, 'NXP_Support_Package_S32V234 version 2.0.0', is also visible. The table has columns for Name, Type, Author, and Install Date.


Name	Type	Author	Install Date
 NXP_Vision_Toolbox_for_S32V234 version 2.0.0	Toolbox	NXP Model-Based Design Toolbox Team	19 November 2018
 NXP_Support_Package_S32V234 version 2.0.0	Toolbox	NXP Model-Based Design Toolbox Team	19 November 2018

5. More details about the NXP's Vision Toolbox can be found by clicking on View Details



Add-On Manager

[Back](#)

 **NXP_Vision_Toolbox_for_S32V234** version 2.0.0 by NXP Model-Based Design Toolbox Team installed on 19 November 2018

Generate code optimized for NXP's S32V234 Automotive Vision Microcontrollers

Toolbox [Open Folder](#) [Learn More](#) [Uninstall...](#)

Description

The NXP Vision Toolbox for MATLAB is a complimentary integrated development environment for S32V234 processor which is a high-performance automotive processor designed to support safe computation-intensive applications in the area of vision and sensor fusion. The NXP Vision Toolbox for MATLAB enables editing, simulation, compiling and deployment of designs from MATLAB environment.

Based on Mathworks® software including Image Processing and Computer Vision System Toolboxes the NXP Vision Toolbox offers designers a straightforward development tool that makes vision accelerators programming easy. NXP software, along with the NXP Vision Toolbox provides a comprehensive enablement environment that reduces development time.

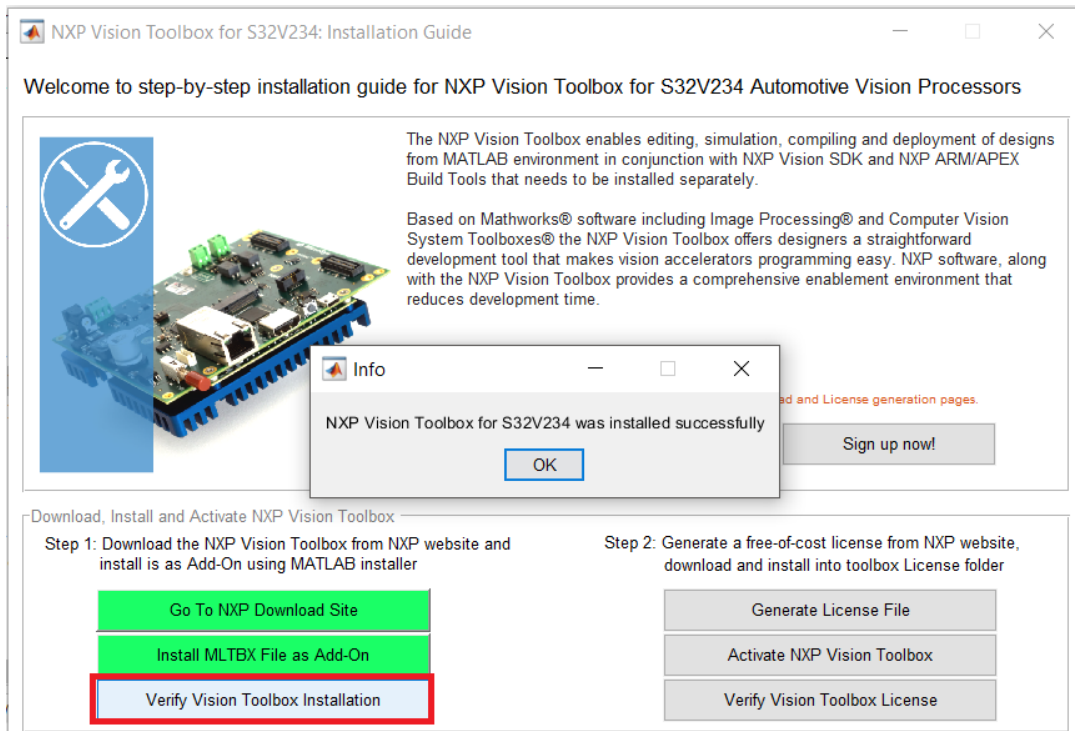
Features:

- Seamless integration with MATLAB environment for easy vision application development, simulations and target running support
- Generate code for Linux™ based applications with direct download to target support
- Built-in support for IO functions to control onboard camera and display
- Integrated NXP Software: Vision SDK – includes ISP and APEX kernels/Linux board support package for A53 core development/NXP APU Compiler/Arm® Compiler
- Support A53 and APEX programming directly from MATLAB m-scripting
- Includes collaterals providing convenient access to manuals, tool guides, how-to videos and application notes
- Example projects demonstrating ISP and APEX functionalities based on Mathworks Computer Vision System Toolbox examples

[View File List](#)

6. NXP Vision Toolbox documentation, help and examples are fully integrated with MATLAB development environment. Get more details by accessing the standard Help and Supplemental Software section

7. In case you are using the Installer Guide, then you have the option to check if the NXP Vision Toolbox is installed correctly on your MATLAB environment by simply clicking on “Verify Vision Toolbox Installation” button



After this step you should see all button related with Vision Toolbox Step 1, green

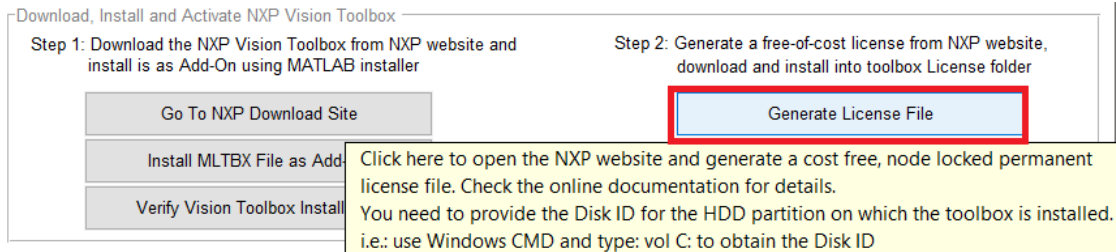


2.3.3 License Generation and Activation

The NXP Vision Toolbox for S32V234 is available free of charge, however, a valid license is required.

You can obtain the NXP Vision Toolbox for S32V234 license free of charge by:

- Using the Installer guide “Generate License File” button



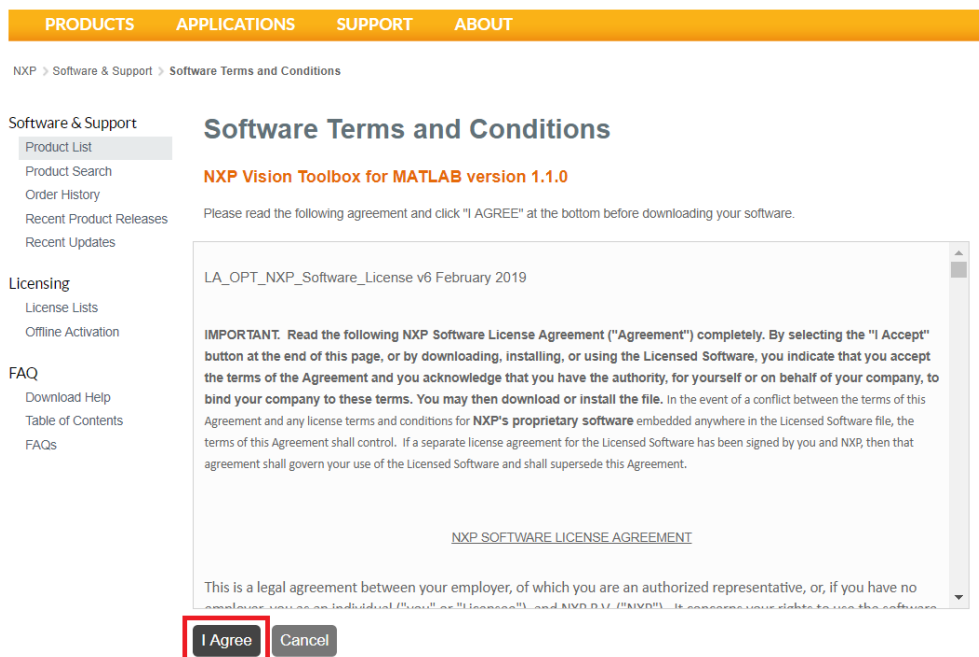
- Go directly into your NXP Software Account and Generate the license using this [link](#)

For more details about license generation please refer to online or offline manual: [Vision_Toolbox_License_Activation.pdf](#)

Perform the following steps to obtain the NXP Vision Toolbox for S32V234 license:

1. For the first-time log-in, the “Software Terms and Conditions” page will be displayed. Click on “I agree” button to consent to the software license agreement.

NOTE In this section we presume, you already logged into your NXP account to download the toolbox prior to license generation step.



2. Click on “License Keys” tab

The screenshot shows the 'Product Download' page for 'NXP Vision Toolbox for MATLAB version 1.1.0'. The navigation bar at the top includes 'PRODUCTS', 'APPLICATIONS', 'SUPPORT', and 'ABOUT'. The breadcrumb trail is 'NXP > Software & Support > Vision Toolbox > NXP Vision Toolbox for MATLAB version 1.1.0 : Files'. On the left, there are sections for 'Software & Support', 'Licensing', and 'FAQ'. The main content area has three tabs: 'Files', 'License Keys' (highlighted with a red box), and 'Notes'. Below the tabs, there is a 'Download Help' link and a paragraph explaining that the toolbox is delivered as a MATLAB MLTBX file. A 'Show All Files' button is present, and a table lists four files with columns for checkboxes, file descriptions, file sizes, and file names. A 'Download Selected Files' button is at the bottom.

<input type="checkbox"/>	File Description	File Size	File Name
<input type="checkbox"/>	+ NXP Support Package for S32V234	1.1 MB	NXP_Support_Package_S32V234_20190403.mltbx
<input type="checkbox"/>	+ NXP Vision Toolbox for S32V234 version 1.1.0	111.8 MB	NXP_Vision_Toolbox_S32V234_1.1.0.RFP_20190322.mltbx
<input type="checkbox"/>	+ Quick Start Guide	3.8 MB	Vision_Toolbox_Quick_Start.pdf
<input type="checkbox"/>	+ Release Notes	1 MB	Vision_Toolbox_Release_Notes.pdf

3. Verify if the correct tool and version are identified and then check the box and click on “Generate” button.

The screenshot shows the 'License Information' page for 'NXP Vision Toolbox for MATLAB version 1.1.0'. The navigation bar at the top includes 'PRODUCTS', 'APPLICATIONS', 'SUPPORT', and 'ABOUT'. The breadcrumb trail is 'NXP > Software & Support > License Information'. On the left, there are sections for 'Software & Support', 'Licensing', and 'FAQ'. The main content area has a 'Generate' button at the top. Below it, there is a table with columns for 'Item Description' and 'NXP Vision Toolbox for MATLAB'. The table lists 'Order Number', 'Purchase Order Number', and 'Total Number of Licenses'. Below the table, there is a checkbox (highlighted with a red box) and a section titled 'License Applicable to Product(s)'. This section has two columns: 'Version' and 'Description'. It lists two versions: '2018' and '1.1.0'. At the bottom, there is a 'Generate' button (highlighted with a red box).

Item Description	NXP Vision Toolbox for MATLAB
Order Number	VISION-MATLAB_v2018.R1_64522011
Purchase Order Number	
Total Number of Licenses:	101

Version	Description
2018	NXP Vision Toolbox for MATLAB version 2018.R1 (View EULA)
R1	
1.1.0	NXP Vision Toolbox for MATLAB version 1.1.0 (View EULA)
	101 Available

- Select Disk Serial Number or Ethernet address as the “Node Host ID”. If you do not know your Disk Serial Number nor the Ethernet address then check the link available on this page with details about License Generation. Enter a name for license to help managing them in case you need to use the Vision Toolbox on multiple computers. (Optional)

PRODUCTS APPLICATIONS SUPPORT ABOUT

NXP > Software & Support > Generate Licenses

Software & Support
 Product List
 Product Search
 Order History
 Recent Product Releases
 Recent Updates

Licensing
 License Lists
 Offline Activation

FAQ
 Download Help
 Table of Contents
 FAQs

Generate Licenses

Instructions for finding your host ID details are available [here](#).

Please do not use spaces in the **Name** field (for node-locked licenses) or **Host Description** field (for floating licenses). These fields are available to add brief text notes to your license.

License Applicable to Product(s):		Number of Licenses Available
<u>Version</u>	<u>Description</u>	
2018 R1	NXP Vision Toolbox for MATLAB version 2018.R1	101
1.1.0	NXP Vision Toolbox for MATLAB version 1.1.0	

Node Host ID:
 Name:

Node Host ID:
 Name:

Node Host ID:
 Name:

Node Host ID:
 Name:

Node Host ID:
 Name:

- Click on “Generate” button to get the license. Verify if the information is correct: Toolbox version, expiration date, Node Host ID

PRODUCTS APPLICATIONS SUPPORT ABOUT

NXP > Software & Support > View Licenses

Software & Support
 Product List
 Product Search
 Order History
 Recent Product Releases
 Recent Updates

Licensing
 License Lists
 Offline Activation

FAQ
 Download Help
 Table of Contents
 FAQs

View Licenses

Below are the licenses you just generated.

License Applicable to Product(s):

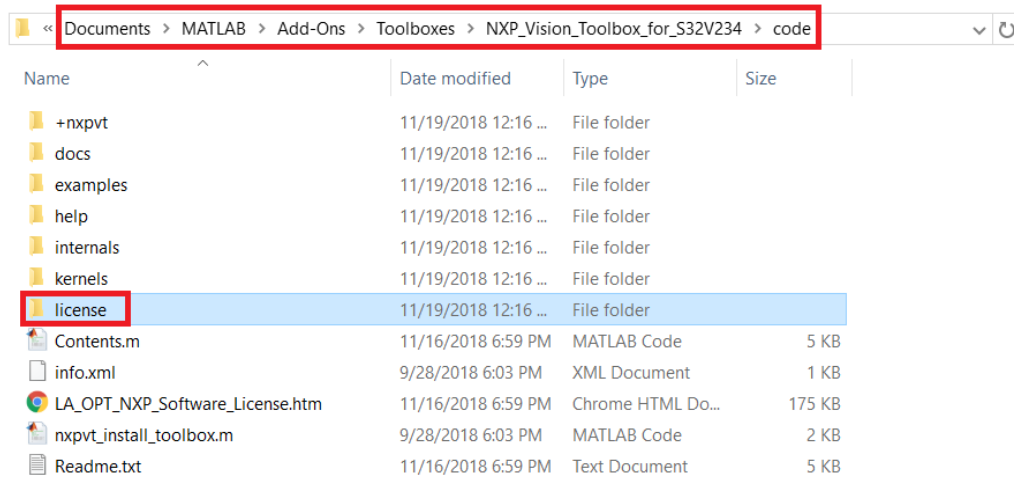
<u>Version</u>	<u>Description</u>	
2018 R1	NXP Vision Toolbox for MATLAB version 2018.R1	
1.1.0	NXP Vision Toolbox for MATLAB version 1.1.0	

License Quantity: 1 Expiration Date: May 28, 2023

Disk Serial Number: 66B72EBD (dp-laptop)
 Generated By: Dumitru-Daniel Popa on Apr 3, 2019

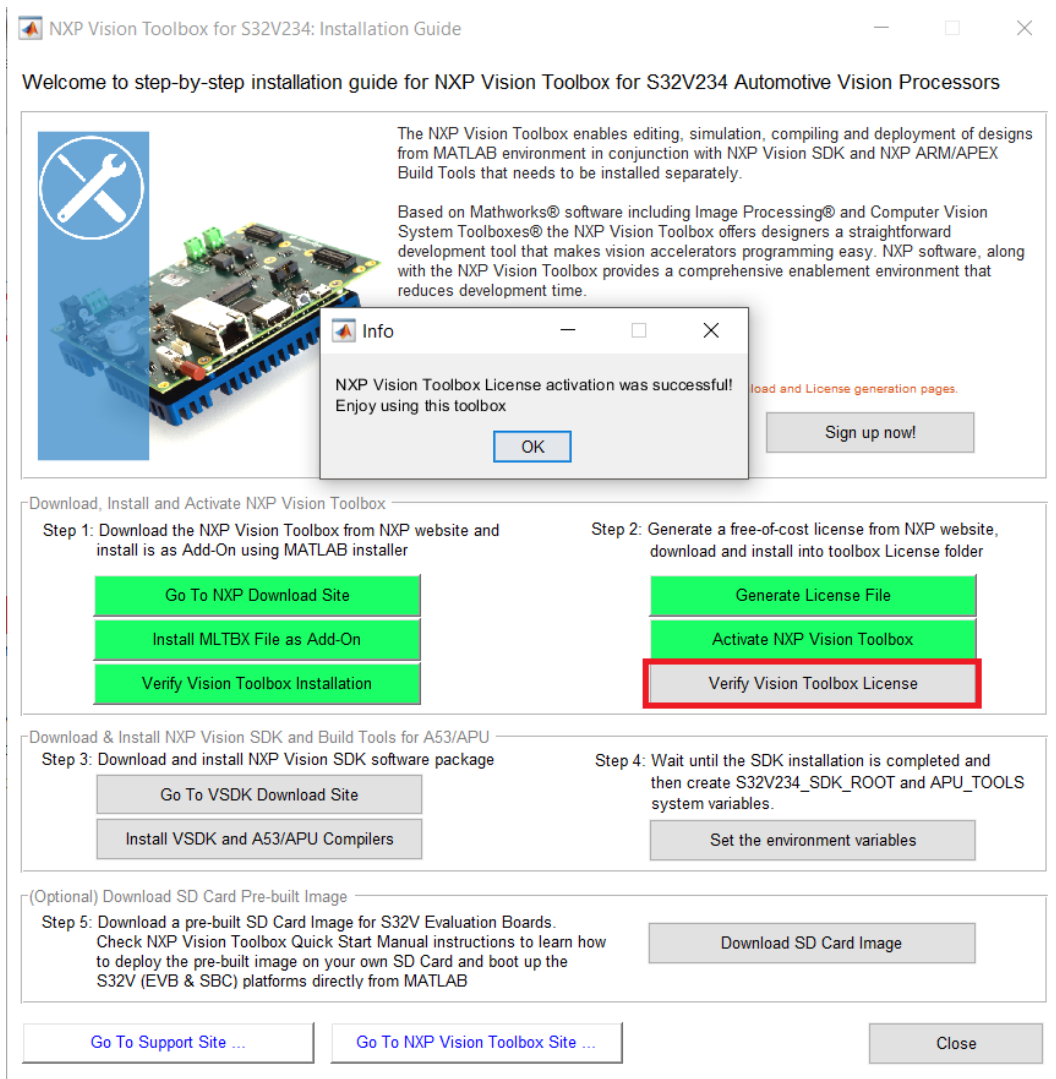
```
#NXP Vision Toolbox for MATLAB - NXP Vision Toolbox for MATLAB version 2018.R1
#for Dumitru-Daniel Popa Software Account
# License for DISK_SERIAL_NUM=66B72EBD dp-laptop
INCREMENT Vision_Toolbox freescale 1.0 28-may-2023 uncounted \
  VENDOR_STRING="NXP Vision Toolbox for MATLAB version 2018.R1" \
  HOSTID=DISK_SERIAL_NUM=66b72ebd ISSUER="Freescale \
  Semiconductor" ISSUED=03-apr-2019 ck=175 SN="FSL - 26360877" \
  TS_OK SIGN="014D 4A15 5613 ECA4 0486 3564 E9D0 163F C645 B54A \
  42C4 32D2 79DD 6063 AB7A 0B68 FF26 94EB A97E 4070 3FC3 8F0F \
  FE39 566D F61A 8B0B 0C54 A144 B7AF 1BF0"
```

6. Either click on “Save All” or copy and paste the file into a text editor, and save the file as “license.dat” into the “Vision Toolbox installed directory\license” folder.

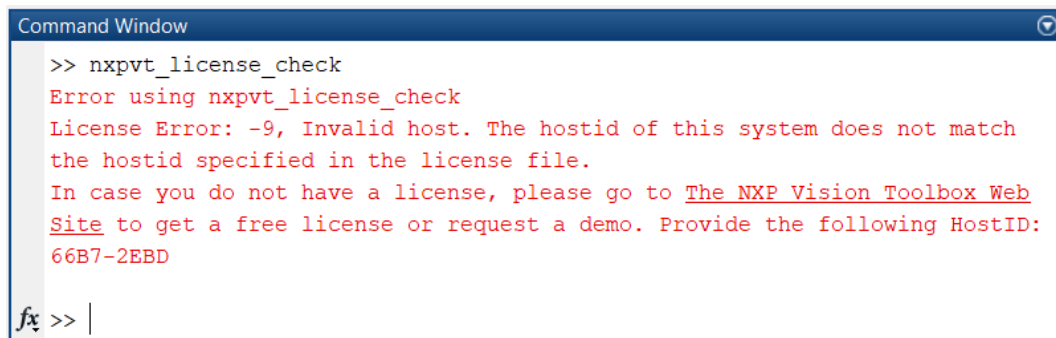


In case you are using the Installer Guide, then you can save the license file anywhere and use the “Activate NXP Vision Toolbox” option to make sure the license is copied correctly in the appropriate toolbox location

7. Check if the license file is installed correctly by using the “Verify Vision Toolbox License” button. If everything is ok, then the Installer Guide will confirm the action



Alternatively, you can check from command line if the license for NXP Vision Toolbox is activated. Run the command `nxpvt_license_check`. If there are issues with the license, this command will return the root-cause.

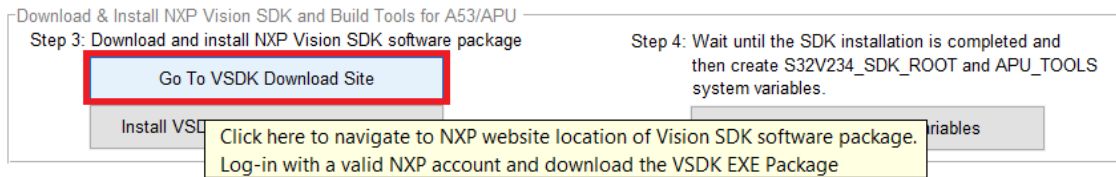


2.3.4 Vision SDK and Build Tools

All the code generated by NXP Vision Toolbox is based on S32V234 Vision SDK package. This software package is also free of charge and apart of optimized kernels and libraries for the S32V automotive vision processors, it also contains the build tools to cross-compile the MATLAB generated code to ARM A53 and APEX cores.

You can obtain the S32V234 Vision SDK free of charge by:

- Using the Installer guide “Go To VSDK Download Site” button



- Go directly to NXP [website](#)

Perform the following steps to obtain and install the S32V234 Vision SDK and NXP Build Tools:

1. Download the Vision SDK RTM v1.3.0 on your PC. Due to the size of the package this might take a while.

NOTE You may need to install additional Hot Fixes that are applicable for the Vision SDK.

PRODUCTS APPLICATIONS SUPPORT ABOUT

NXP > Software & Support > Product Information : Automotive SW - Vision Software

Software & Support

- Product List
- Product Search
- Order History
- Recent Product Releases
- Recent Updates

Licensing

- License Lists
- Offline Activation

FAQ

- Download Help
- Table of Contents
- FAQs

Product Information

Automotive SW - Vision Software

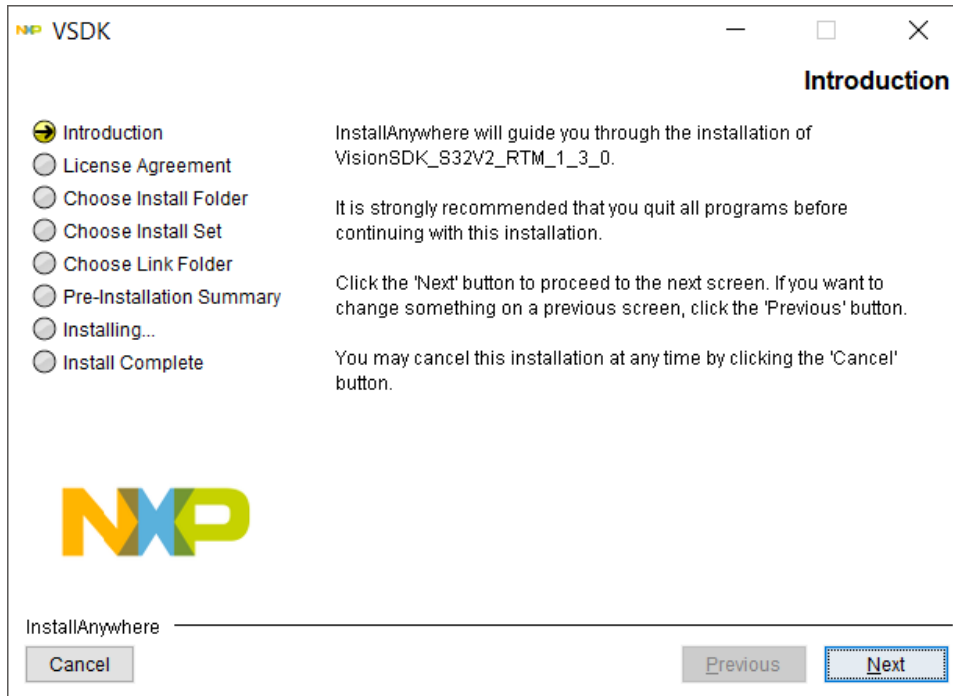
To register a New Product please click on the button below

Register

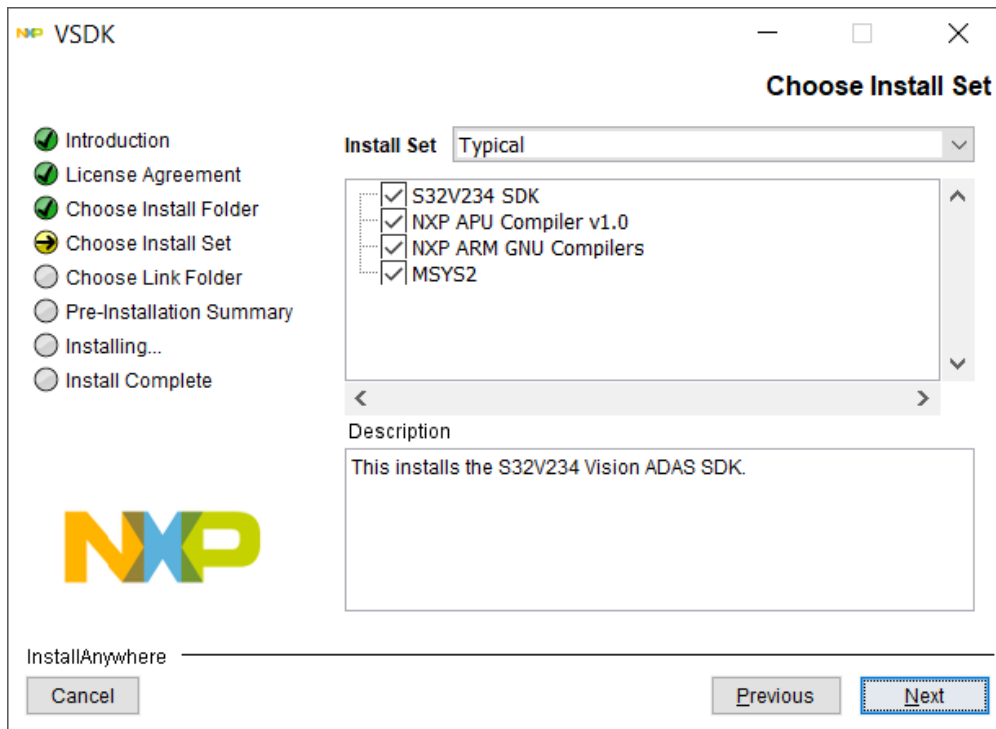
Current Previous

Version	Description	
1.0.0	SW32V23-VSDKQNX-RTM-1.0.0	Download Log
S32V2 Vision Software Development Kit for QNX BSP		
1.3.0	SW32V23-VSDK001-RTM-1.3.0	Download Log
S32V2 Vision Software Development Kit for Linux BSP		
The Vision Software Development Kit (VisionSDK) for S32V2 provides a comprehensive SW enablement environment for NXP/AMPs 2nd generation of vision processors, S32V2xx. The S32V2xx family of devices is designed to support computation intensive applications for image processing.		
The VisionSDK provides comprehensive abstraction of the powerful accelerators for image signal processing (ISP) and massive parallel computing (APEX). Well documented APIs help to fully exploit the HW capabilities and create applications easily.		
Users of the VisionSDK can be inspired by the broad offering of demo applications bundled with the package coming with full source code. The VisionSDK is topped with its sophisticated build system which makes it easy to create new projects. For convenient code development NXP offers the eclipse based Design Studio for Vision.		

2. Once the VisionSDK_S32V2_RTM_1_3_0.exe download is finished, select “Install VSDK and A53/APU Compilers” option in the Installer Guide UI.
3. Select the exe file and wait for the Vision SDK Install Anywhere to start.



4. Make sure you follow all the steps and install the:
 - NXP APU Compiler v1.0 – used to compile the generated code for APEX Vision Accelerator
 - NXP ARM GNU Compilers – used to compile the generated code for ARM A53
 - MSYS2 – used to configure the bootable Linux image and to download the actual vision application to the S32V234 Evaluation Board



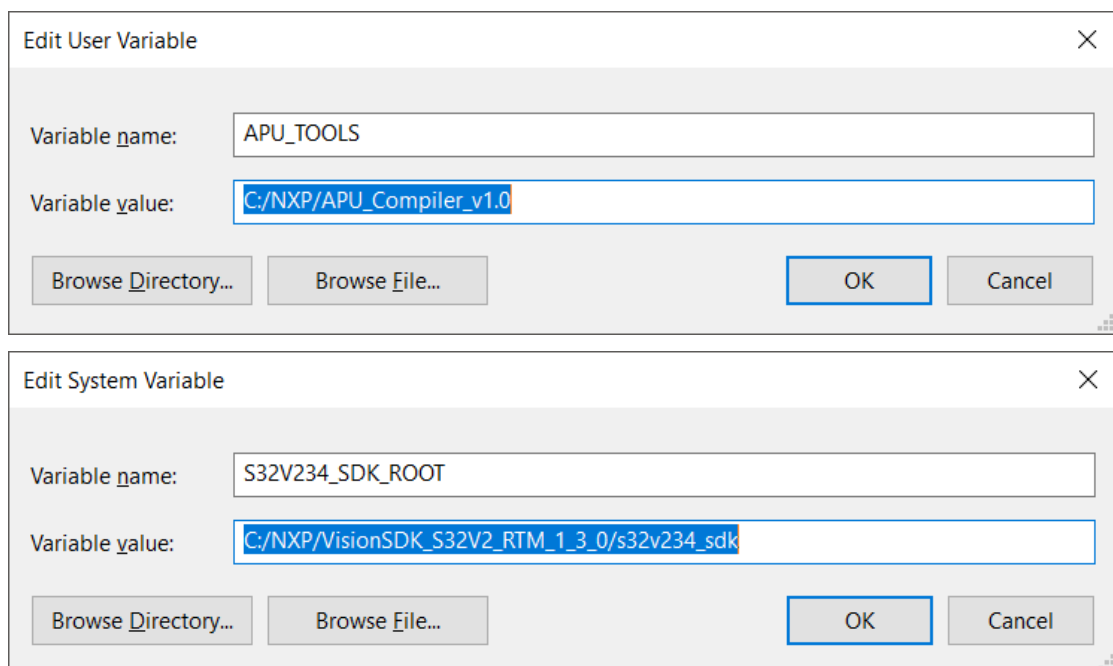
2.3.5 Setting up the Environment

The last step required for software configuration is to set two system or user environmental variables `APU_TOOLS` and `S32V234_SDK_ROOT` that points to:

APU_TOOLS= C:/NXP/APU_Compiler_v1.0

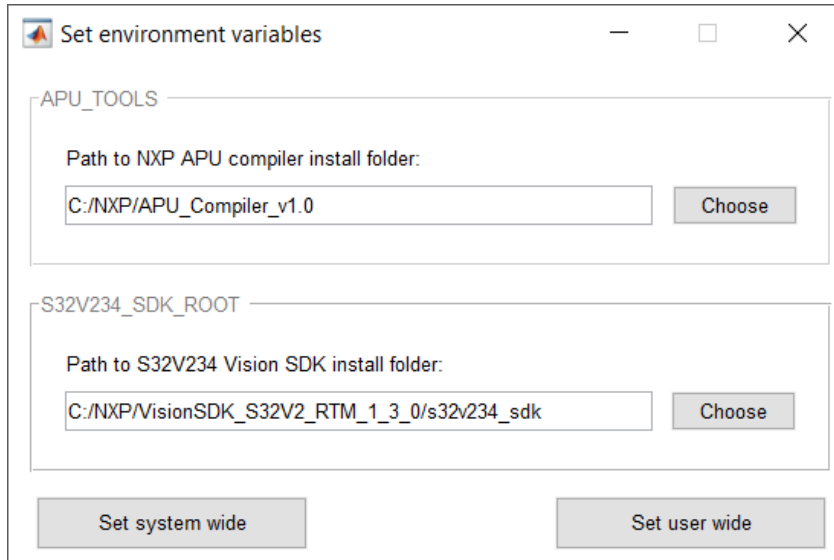
S32V234_SDK_ROOT = C:/NXP/VisionSDK_S32V2_RTM_1_3_0/s32v234_sdk

Ensure system or user environment variables, corresponding to the compiler(s) you have installed, are defined to compiler path value as shown below:



Note: Paths shown are for illustration, your installation path may be different. Once environmental variables are setup you will need to restart MATLAB to use these variables.

An alternative for setting the system paths manually is the “Set the environment variables” option from the NXP Vision Toolbox support package installer:



NOTE If the MATLAB is open with Administrator rights, then the “Set system wide” can be used to set the system variables. Otherwise (most of the cases) use “Set user wide” to setup the environment variables.

2.4 Optional Software

This section describes the additional software that may be needed to have the full setup working and to be able to download and run vision application directly from MATLAB on S32V234 Evaluation Boards.

2.4.1 SD Card Bootable Linux Image

The S32V234 Vision SDK is delivered with pre-built images that can be used to configure the S32V234 evaluation board to have it up and running for vision application. In case you are familiar with Linux OS, then please follow the procedures shown in the Vision SDK Manuals for building and configuration of the bootable SD Card.

For users that are not familiar with Linux OS or simply do not have a Linux Machine available to configure the SD Card for the S32V234 EVB there is a simple alternative.

Using the Installer Guide you can download a pre-built Linux bootable images for S32V234 EVB and S32V234 SBC boards that can be configured by MATLAB directly from Windows OS.

(Optional) Download SD Card Pre-built Image

Step 5: Download a pre-built SD Card Image for S32V Evaluation Boards. Check NXP Vision Toolbox Quick Start Manual instructions to learn how to deploy the pre-built image on your own SD Card and boot up the S32V (EVB & SBC) platform.

[Download SD Card Image](#)

Click here to navigate to NXP website location of Pre-built SD-Card Image.
The SD-Card Image can be copied directly from Windows OS to an empty SD-CARD to create a bootable Linux card needed for running vision applications

[Go To Support Site ...](#)

PRODUCTS APPLICATIONS SUPPORT ABOUT

NXP > [Software & Support](#) > Automotive SW - Vision Software > SW32V23-VSDK001-RTM-1.3.0 : Files

Software & Support

- Product List
- Product Search
- Order History
- Recent Product Releases
- Recent Updates

Licensing

- License Lists
- Offline Activation

FAQ

- Download Help
- Table of Contents
- FAQs

Product Download

SW32V23-VSDK001-RTM-1.3.0

Files License Keys Notes [Download Help](#)

Show All Files 10 Files

	File Description	File Size	File Name
<input type="checkbox"/>	+ README.md	962 bytes	↓ README.md
<input type="checkbox"/>	+ VisionSDK RTM 1.3.0 installer for Linux host.	517.3 MB	↓ VisionSDK_S32V2_RTM_1_3_0.bin
<input type="checkbox"/>	+ VisionSDK RTM 1.3.0 installer for Windows host.	891.2 MB	↓ VisionSDK_S32V2_RTM_1_3_0.exe
<input type="checkbox"/>	+ VisionSDK RTM 1.3.0 list of known bugs.	778.7 KB	↓ VisionSDK_S32V2_RTM_1_3_0_NLKOBS.xlsx
<input type="checkbox"/>	+ VisionSDK RTM 1.3.0 pre-built SD Card image based on Debian roots.	1.8 GB	↓ VisionSDK_S32V2_RTM_1_3_0_img_debian.tar.gz
<input type="checkbox"/>	+ VisionSDK RTM 1.3.0 pre-built SD Card image based on Yocto roots.	1.4 GB	↓ VisionSDK_S32V2_RTM_1_3_0_img_yocto.tar.gz
<input type="checkbox"/>	+ VisionSDK RTM 1.3.0 quality package.	44.4 MB	↓ VisionSDK_S32V2_RTM_1_3_0_QualityPackage.zip
<input type="checkbox"/>	+ VisionSDK RTM 1.3.0 release notes.	37 KB	↓ VisionSDK_S32V2_RTM_1_3_0_ReleaseNotes.txt
<input type="checkbox"/>	+ VisionSDK RTM 1.3.0 software content register.	1 MB	↓ VisionSDK_S32V2_RTM_1_3_0_SCR.xlsx
<input type="checkbox"/>	+ VisionSDK RTM 1.3.0 user guide.	1.5 MB	↓ VisionSDK_S32V2_RTM_1_3_0_UserGuide.pdf

[Download Selected Files](#)

In this case you need to provide at least 4GB microSD card, preferably SDHC or SDXC class 10.

3 Vision Application

This section shows how to simulate, generate the code, configure the S32V234 evaluation and run a face detection application in real time on the NXP Hardware.

3.1 Examples Library & Help

NXP's Vision Toolbox comes with an Examples Library that let you test and run multiple applications. To open the library, go to `MATLAB Help` (or simply press F1) and select the `NXP Vision Toolbox for S32V234 Supplemental Software`

There are four groups of examples that highlights four different types of functionalities supported by NXP Vision Toolbox for S32V234:

- Vision Applications – contains complex application like face, pedestrian and lane detection demos that can be run in both simulation and hardware;
- APEX Kernels – contains examples like Sobel and Gauss filters;
- APEX Computer Vision – shows how to use the APEXCV classes to build optimized examples on ARM and APEX cores;
- Convolutional Neural Network examples – shows how to use the ARM cores to run SqueezeNet, AlexNet and GoogLeNet pretrained networks
- S32V234 EVB IO Examples;

For the Quick Start we are going to choose the Face Detection application demo and go thru all steps to simulate and run on the target.

3.2 Face Detection in Simulation Mode

3.2.1 Running the Algorithm for Images

Go to `...examples/apps/face` detection folder and open the m-script file `face_detection_image_main.m`

Alternatively, you can open the example from the MATLAB Help. The script should look like the one shown below. This script is using as input an image and will run the face detection algorithm that is implemented using Local Binary Patterns and Cascading Classifiers. At the end, if any faces are found, the script will display a red rectangle on top of the original image.

```
function face_detection_image_main() %#codegen
    inImgPath = 'data/face_detection.png';

    inImgUMat = nxpvt.imread(inImgPath);
    if isempty(inImgUMat)
        fprintf('Failed to open input image: %s.', inImgPath);
        return;
    end

    height = uint32(inImgUMat.height);
    width = uint32(inImgUMat.width);

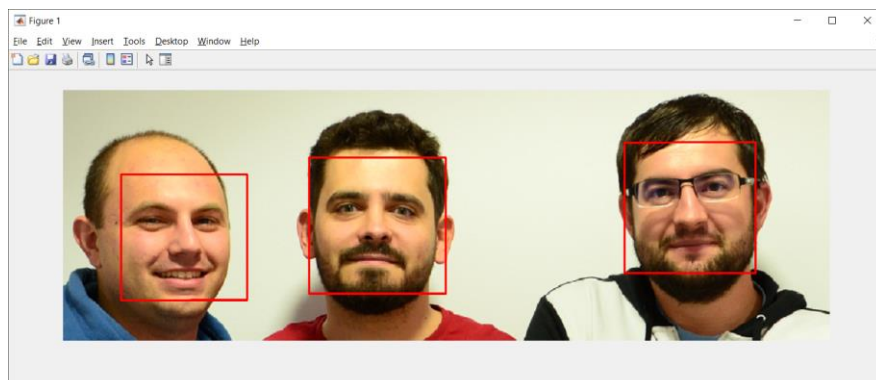
    fdetector = nxpvt.CascadeObjectDetector('data/lbpcascade_frontalface.xml',
        'ScaleFactor',1.1, 'MinSize',[110 110], 'MaxSize',[250 250],
        'SkipOdd',1,'MergeThreshold',4);
    resizeObj = nxpvt.apexcv.Resize();
    fNum = int32(0);
    nxpvt.tic;

    % Get faces.
    [bbox, l] = step(fdetector, inImgUMat);
    nxpvt.cv.rectangle(inImgUMat, bbox, [255, 0 ,0], 5);

    f = min(720 / single(height), 1280 / single(width));
    inImgUMat = Process(resizeObj, inImgUMat, f);
    nxpvt.imshow(inImgUMat);

    fps = int32(fix(1/nxpvt.toc));
    fprintf('%d] FPS: %d, Faces detected: %d, \n', fNum, fps, int32(l));
end
```

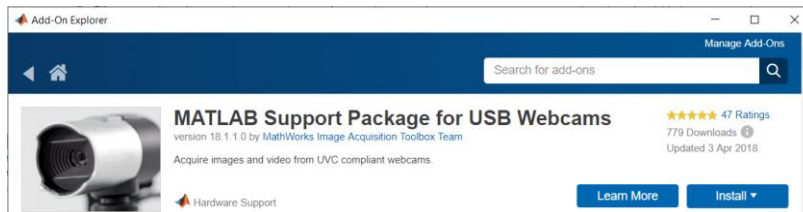
All you should do is to run this script from MATLAB to produce the results. Press F5 to start. The result should be identical with the one shown in the next figure.



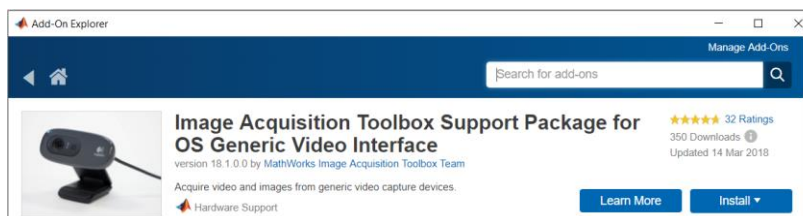
3.2.2 Running the Algorithm using Video Frames from PC Webcam

The next step is to try the algorithm on some real footage capture from a webcam. Before running this test, you need to install two additional toolboxes that allows you to capture frames from the webcam. Use the MATLAB Get Add-On menu to find and install the following toolboxes:

- [MATLAB Support Package for USB Webcams](#)



- [Image Acquisition Toolbox Support Package for OS Generic Video Interface](#)



Once both these toolboxes are installed, go to ...examples/apps/face detection folder and open the m-script file face_detection_camera_main.m

```
function face_detection_camera_main() %#codegen
    width = uint32(1280);
    height = uint32(720);

    if coder.target('MATLAB')
        input = nxpvt.videoinput('winvideo', 1, width, height);
    else
        input = nxpvt.videoinput('sony', 1, width, height, true, false);
    end

    fdetector = nxpvt.CascadeObjectDetector('data/lbpcascade_frontalface.xml',
        'ScaleFactor',1.1, 'MinSize',[110 -1], 'MaxSize',[250 -1], 'SkipOdd',1,
        'MergeThreshold',2);

    fNum = int32(0);
    fps = int32(0);
    while true
        nxpvt.tic;
        fNum = fNum + 1;
        frame = input.getsnapshot();

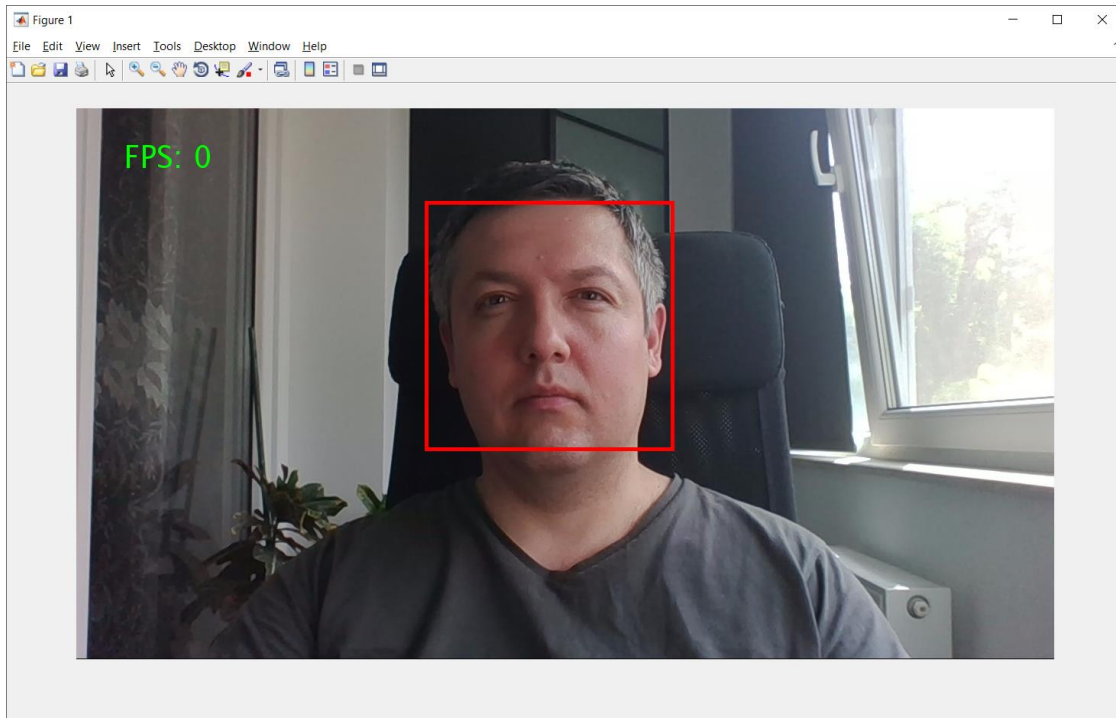
        % Get faces.
        [bbox, 1] = step(fdetector, frame);

        nxpvt.cv.rectangle(frame, bbox, [255, 0, 0], 5);
        nxpvt.cv.putText(frame, sprintf('FPS: %d', fps), [10, 40],...
            'FONT_HERSHEY_SIMPLEX', 1, [0, 255, 0], 2);
        nxpvt.imshow(frame);

        fps = int32(fix(1/nxpvt.toc));
        fprintf(['%d] FPS: %d, Faces detected: %d, \n', fNum, fps, int32(1)];
    end
end
```

In this example the source of the video frame is different between MATLAB Simulation and S32V234 Hardware Test. Using `coder.target()` we can choose the source of the inputs frames during code generation stage.

If the setup was performed correctly, the algorithm should detect your face.



3.2.3 Running the Algorithm for Videos

The vision algorithm can be then tested on pre-recorded video to emulate and test the real use-case on the hardware. Go to ...examples/apps/face_detection folder and open the m-script file `face_detection_video_main.m`

```
function face_detection_video_main() %#codegen
    if coder.target('MATLAB')
        videoFile = fullfile(matlabroot, 'toolbox', 'vision', 'visiondata',...
            'visionface.avi');
    else
        videoFile = 'data/visionface.avi';
    end

    videoReader = nxpvt.VideoReader(videoFile);

    if coder.target('MATLAB')
        % Generate a video player, only for simulation.
        videoPlayer = vision.VideoPlayer();
    end

    fdetector = nxpvt.CascadeObjectDetector('data/lbpcascade_frontalface.xml',
        'ScaleFactor',1.1, 'MinSize',[110 -1], 'MaxSize',[250 -1], 'SkipOdd',1,
        'MergeThreshold',2);

    fNum = int32(0);
    fps = int32(0);

    while hasFrame(videoReader)
        nxpvt.tic;
        fNum = fNum + 1;
        frame = readFrame(videoReader);

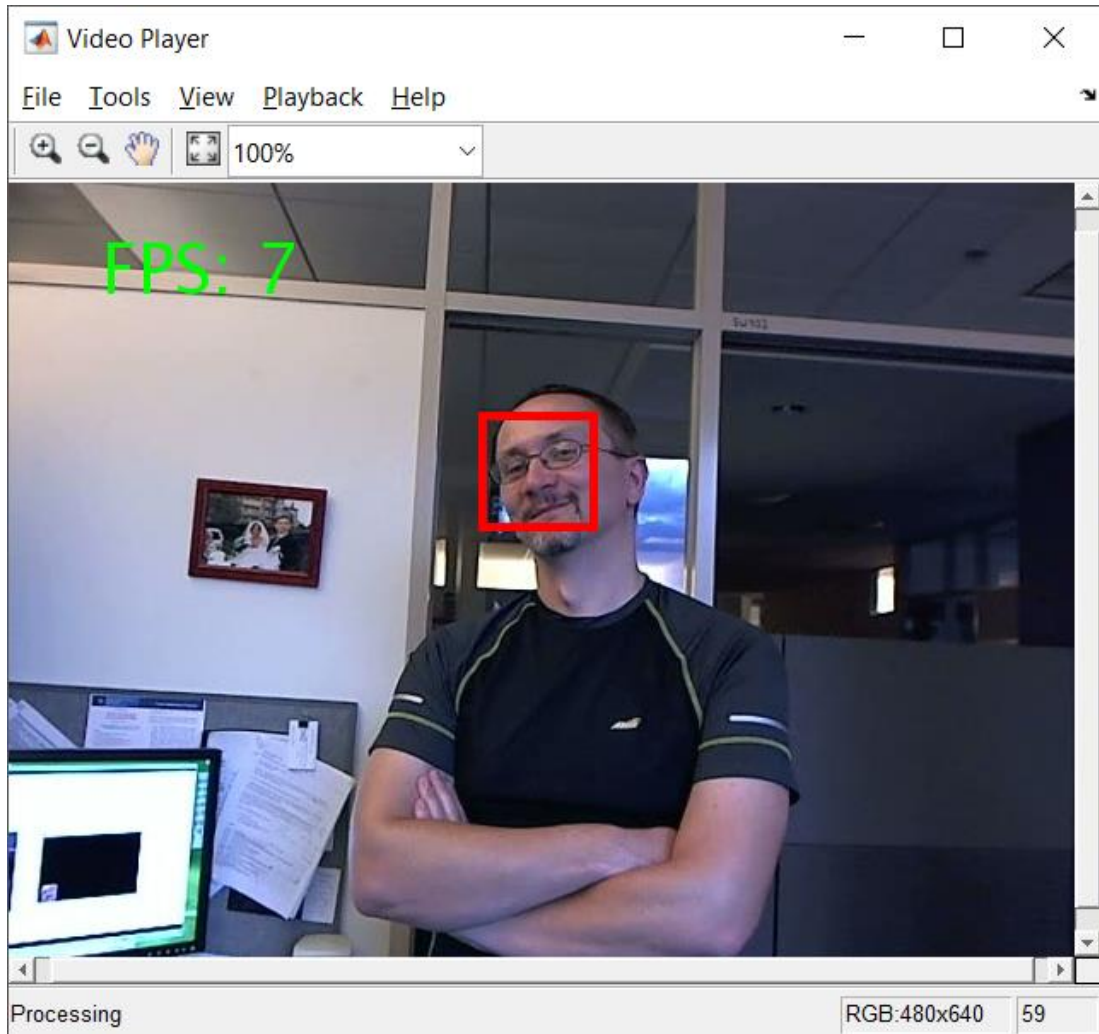
        % Get faces.
        [bbox, l] = step(fdetector, frame);
        nxpvt.cv.rectangle(frame, bbox, [255, 0, 0], 5);
        nxpvt.cv.putText(frame, sprintf('FPS: %d', fps), [10, 40],...
            'FONT_HERSHEY_SIMPLEX', 1, [0, 255, 0], 2);

        if coder.target('MATLAB')
            step(videoPlayer, frame.data);
            %Exit the loop if the video player figure is closed.
            if ~isOpen(videoPlayer)
                break;
            end
        else
            nxpvt.imshow(frame);
        end

        fps = int32(fix(1/nxpvt.toc));
        fprintf(['%d] FPS: %d, Faces detected: %d, \n', fNum, fps, int32(l));
    end
end
```

The same m-script can be executed in both Simulation or on the S32V234 EVB/SBC.

In this case we are using a video source from one of the standard MATLAB [examples](#). If the test is successful you should see this:



3.3 Face Detection on S32V234 Vision Processor

In this section the focus is on the steps required to generate the code and running the application on the S32V234 microprocessor.

3.3.1 Configure the microSD Card

As it was mentioned in the paragraph 2.4.1, you need to have a Linux OS configured to boot up the S32V234 platform to enable the Eth, ISP, ACF and other various drivers and modules needed for running embedded vision algorithms.

The entire procedure for configuration and booting up the platform is described in the Vision SDK manuals. Unfortunately, not everyone has access to a Host PC with Linux OS to configure a SD card (*formatting, uboot, filesystem, linux image copy*). For this reason, a complete microSD card bootable image for S32V234EVB, S32V234SBC or S32V234PCIE that can be configured from Windows OS is distributed alongside NXP Vision SDK.

Follow the next steps to create a bootable SD card for S32V234 SBC evaluation board:

1. Begin by inserting a microSD card with at least 4GB capacity in your Host PC running Windows OS. The Windows OS should be able to recognize the SD card and assign a drive letter (e.g.: “D:”)



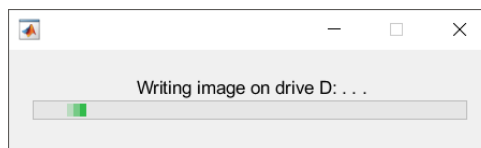
2. From MATLAB command window run the command:

```
nxpvt_create_target('sdcard-sbc.tar.bz2', 'D:');
```

This command will format the card and then is going to copy all the required files from the *.bz2 image to the SD Card for booting up the Linux on S32V234 SBC.

NOTE This example assumes you have untar the SD Card archive downloaded from the NXP website and you run the `nxpvt_create_target` command from the same directory as `sdcard-sbc.tar.bz2` image

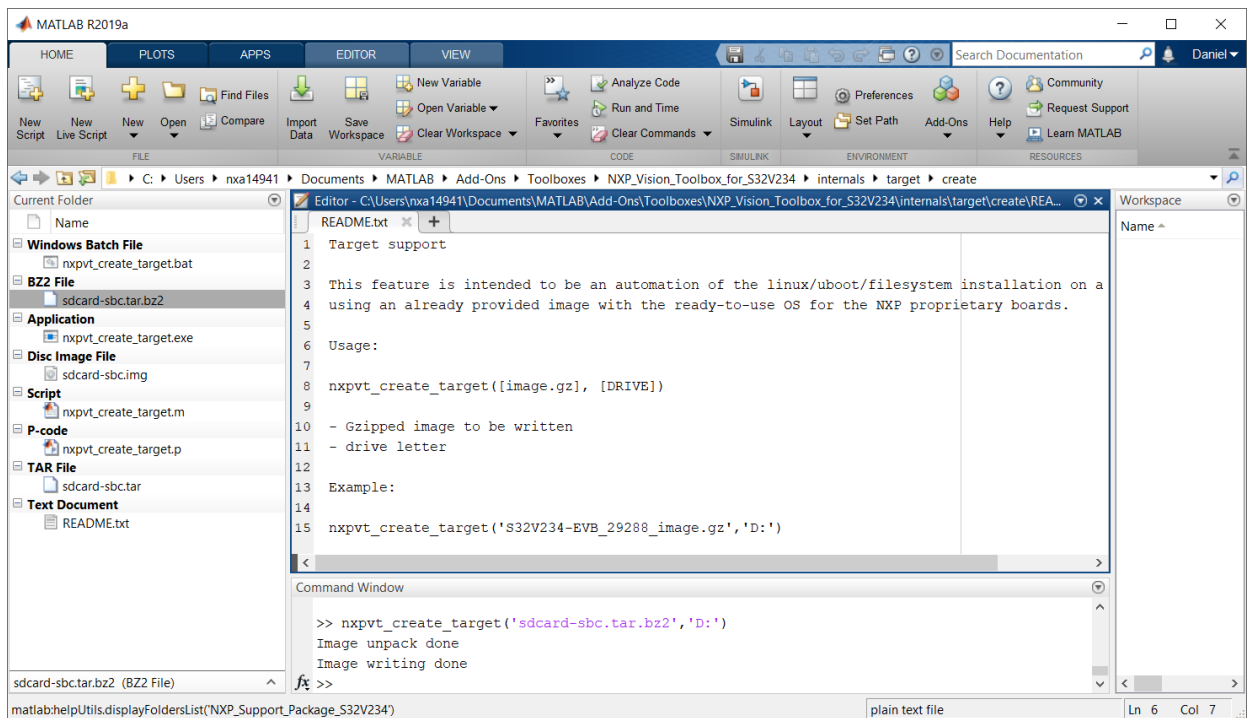
3. The copying process might take a while depending on the SD Card class type. During the process the following message will be shown on the screen. Wait until the copying process is finalized and the “Image writing done” message is displayed on the MATLAB command prompt.



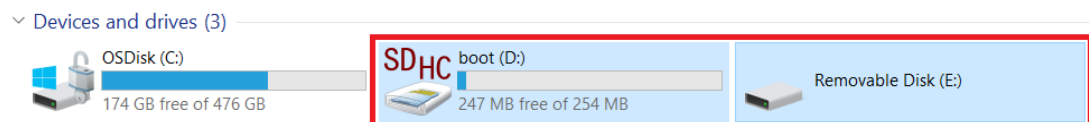

```

0000/41943041945600/41943041971200/41943041996800/41943042022400/41943042048000/4194304207
3600/41943042099200/41943042124800/41943042150400/41943042176000/41943042201600/4194304222
7200/41943042252800/41943042278400/41943042304000/41943042329600/41943042355200/4194304238
0800/41943042406400/41943042432000/41943042457600/41943042483200/41943042508800/4194304253
4400/41943042560000/41943042585600/41943042611200/41943042636800/41943042662400/4194304268
8000/41943042713600/41943042739200/41943042764800/41943042790400/41943042816000/4194304284
1600/41943042867200/41943042892800/41943042918400/41943042944000/41943042969600/4194304299
5200/41943043020800/41943043046400/41943043072000/41943043097600/41943043123200/4194304314
8800/41943043174400/41943043200000/41943043225600/41943043251200/41943043276800/4194304330
2400/41943043328000/41943043353600/41943043379200/41943043404800/41943043430400/4194304345
6000/41943043481600/41943043507200/41943043532800/41943043558400/41943043584000/4194304360
9600/41943043635200/41943043660800/41943043686400/41943043712000/41943043737600/4194304376
3200/41943043788800/41943043814400/41943043840000/41943043865600/41943043891200/4194304391
6800/41943043942400/41943043968000/41943043993600/41943044019200/41943044044800/4194304407
0400/41943044096000/41943044121600/41943044147200/41943044172800/41943044198400/4194304422400
Press any key to continue . . .

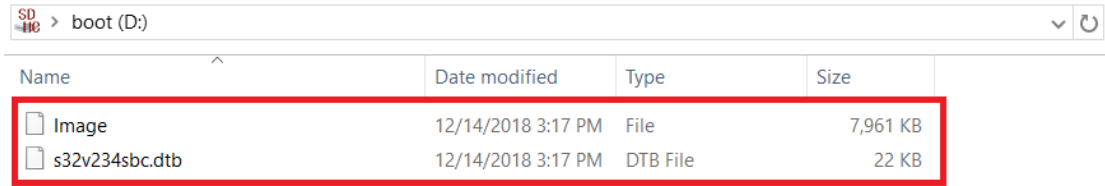
```



4. After the copying process is completed, you should be able to see an additional drive mapped on your system (e.g. E) that cannot be accessed since it is a ext3 file system type.



5. Check that the initial mapped drive (e.g. D) contains: Image and s32v234-sbc files



Name	Date modified	Type	Size
Image	12/14/2018 3:17 PM	File	7,961 KB
s32v234sbc.dtb	12/14/2018 3:17 PM	DTB File	22 KB

6. Remove the SD card from the Host PC and check the next section for details on how to bootup the S32V234 SBC Evaluation Board

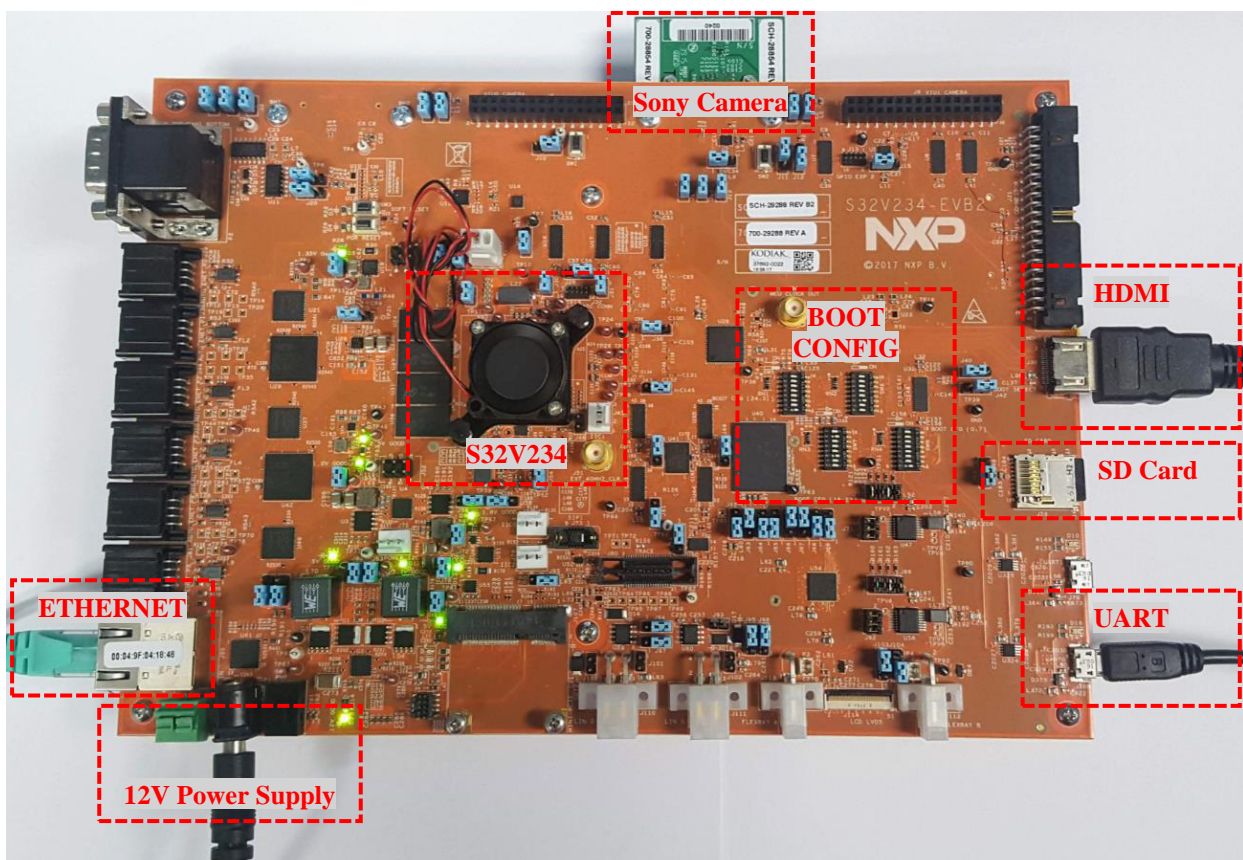
3.3.2 S32V234 Evaluation Board Configuration

All the examples provided with the Vision Toolbox were developed on [S32V234EVB](#) and [S32V234SBC](#). Additional information about these development kits can be found on NXP official web pages.

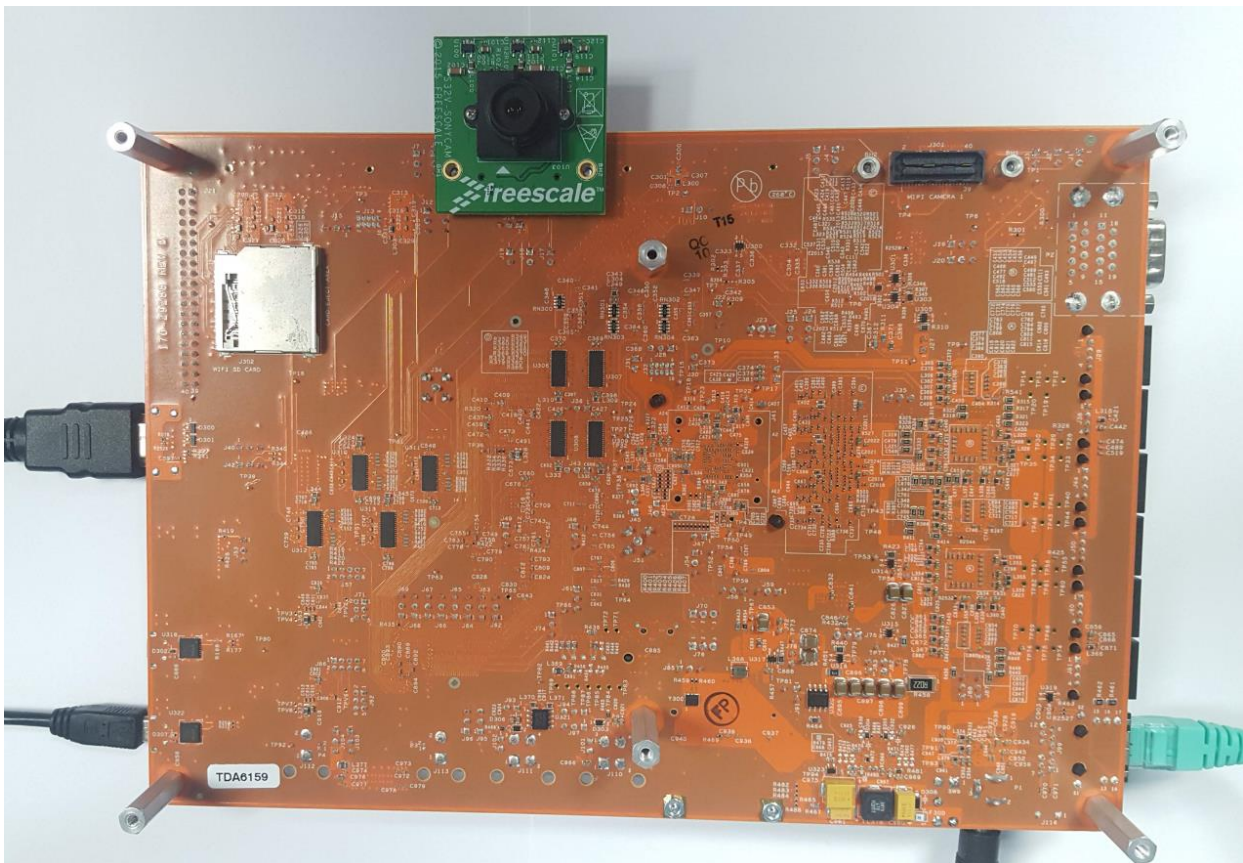
3.3.2.1 S32V234 EVB2 HW Setup

Before running any example on the S32V234 EVB2 you need to perform the following steps:

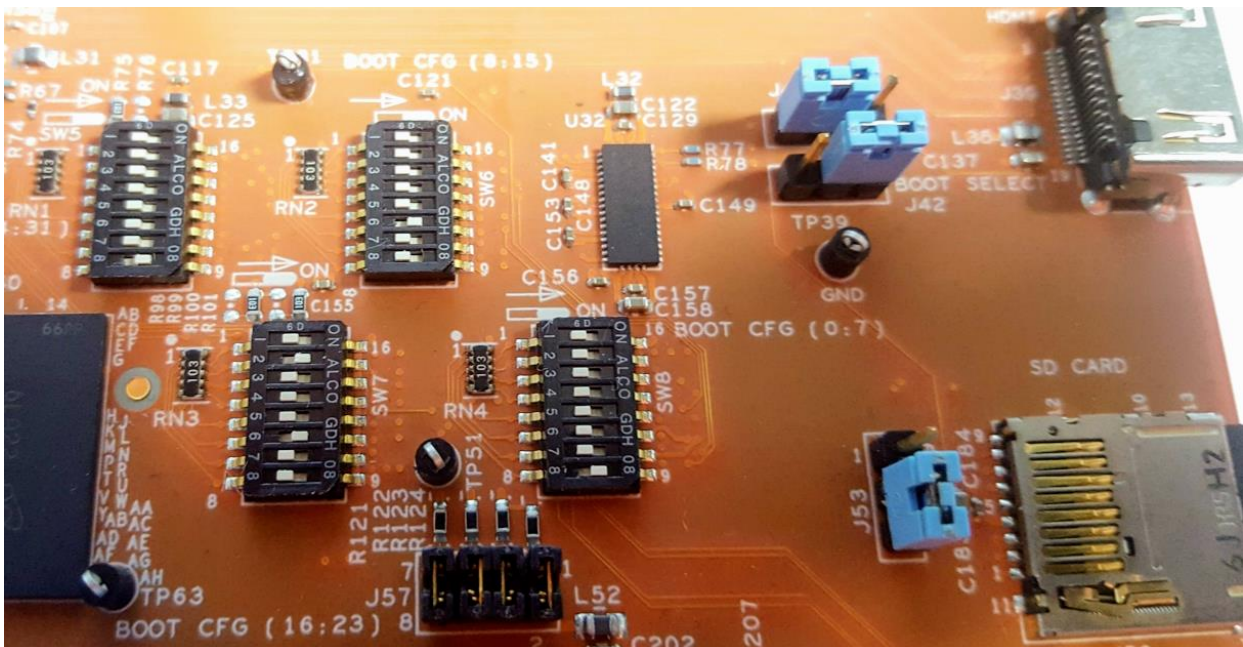
1. Insert the micro SD-card that has been configured in the previous section into the micro SD card slot
2. Insert the Sony camera into the MIPI Camera 0 port. The Sony camera is used for capturing the video frames used for computer vision processing
3. Insert an Ethernet cable in the ETH port. This will be used for downloading the application via TCP/IP
4. Connect the S32V234EVB via a microUSB cable with your Host PC. This is used for finding the IP of the board.
5. Set the Jumper J36 into position 1-2 to allow data to be displayed on a LCD monitor via HDMI
6. Connect a LCD monitor via HDMI cable with S32V234EVB
7. Configure the S32V234EVB Boot Configuration switches as shown below
8. Power on the board



The back side of the S32V234 EVB2 Evaluation board is shown below:



The boot configuration switches position is shown below:

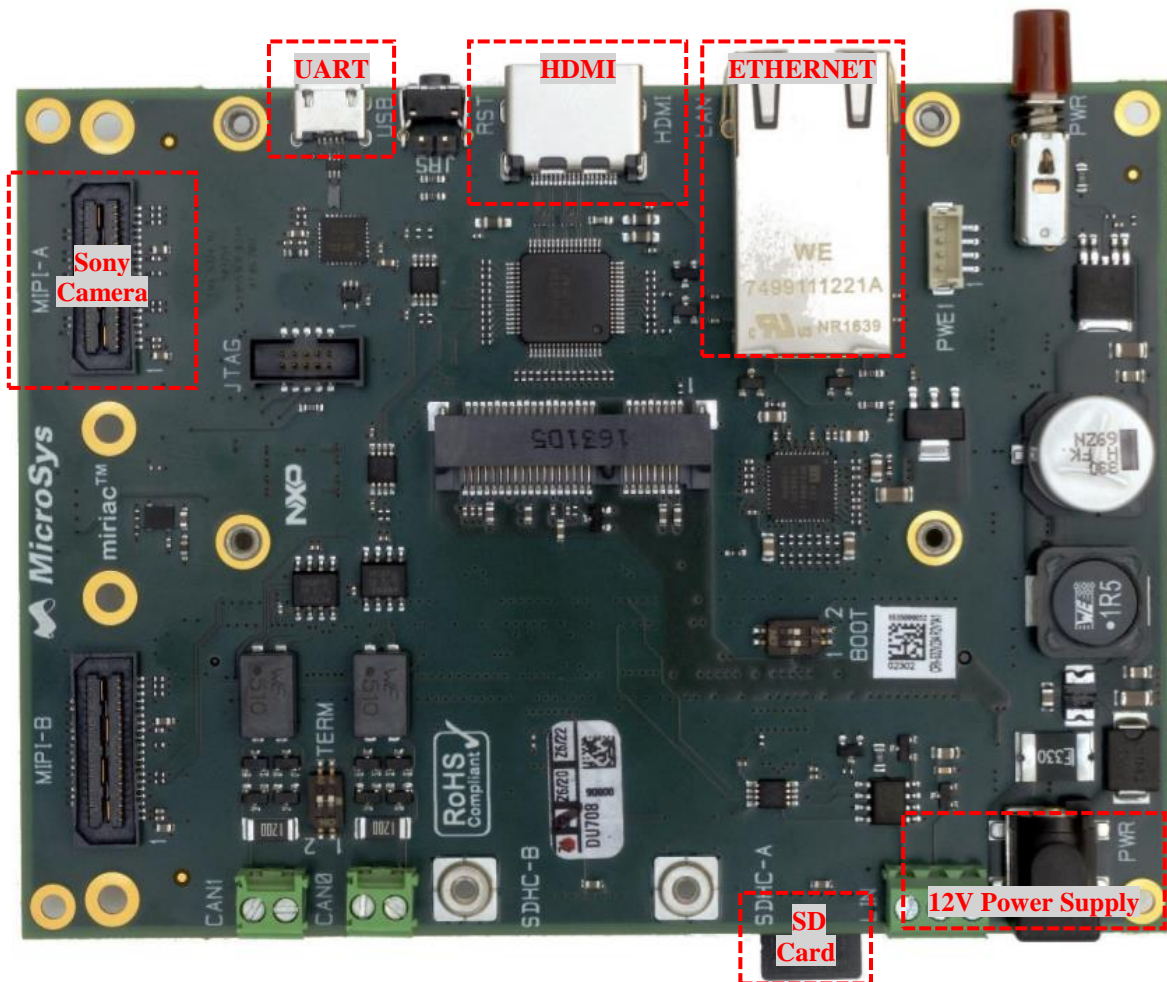


3.3.2.2 S32V234 SBC HW Setup

Before running any example on the S32V234 SBC you need to perform the following steps:

1. Insert the micro SD-card that has been configured in the previous section into the micro SD card slot
2. Insert the Sony camera into the MIPI-A port. The Sony camera is used for capturing the video frames used for computer vision processing
3. Insert an Ethernet cable in the ETH port. This will be used for downloading the application via TCP/IP
4. Connect the S32V234 SBC via a microUSB cable with your Host PC. This is used for finding the IP of the board.
5. Connect a LCD monitor via HDMI cable with S32V234 SBC
6. Power on the board

For more details please review the [SBC-S32V User Manual](#).

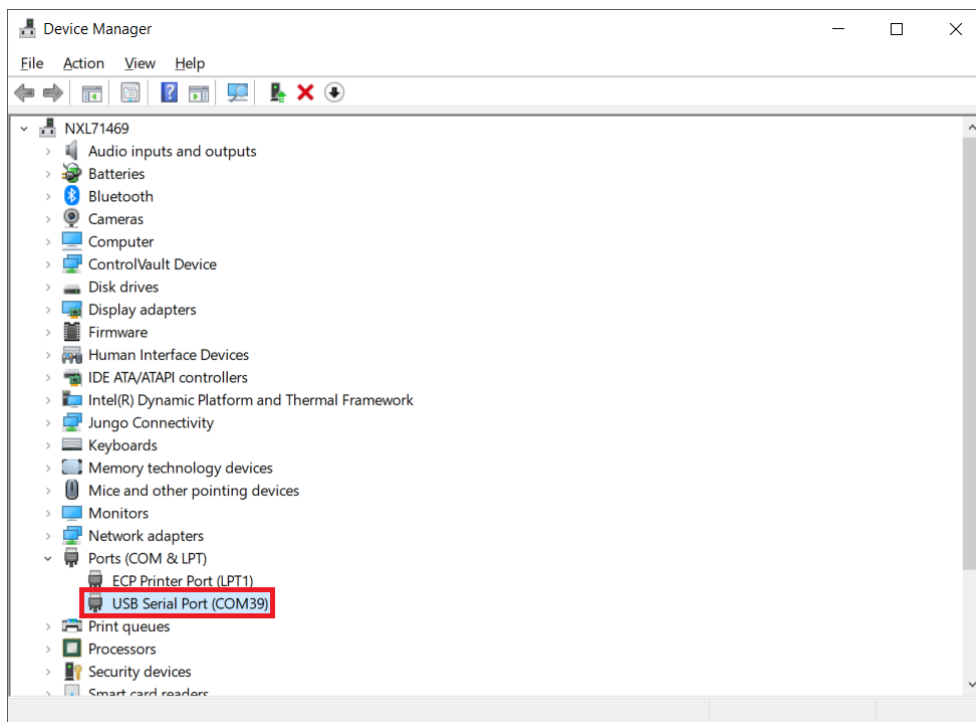


3.3.3 USB to UART connection

The UART enables the users to configure the boot up process and to find out the board IP address after reset.

To be able to connect the S32V234 board to the host PC running on Windows OS, the USB to UART bridge FDT Driver needs to be installed. Follow the next steps to configure the UART communication:

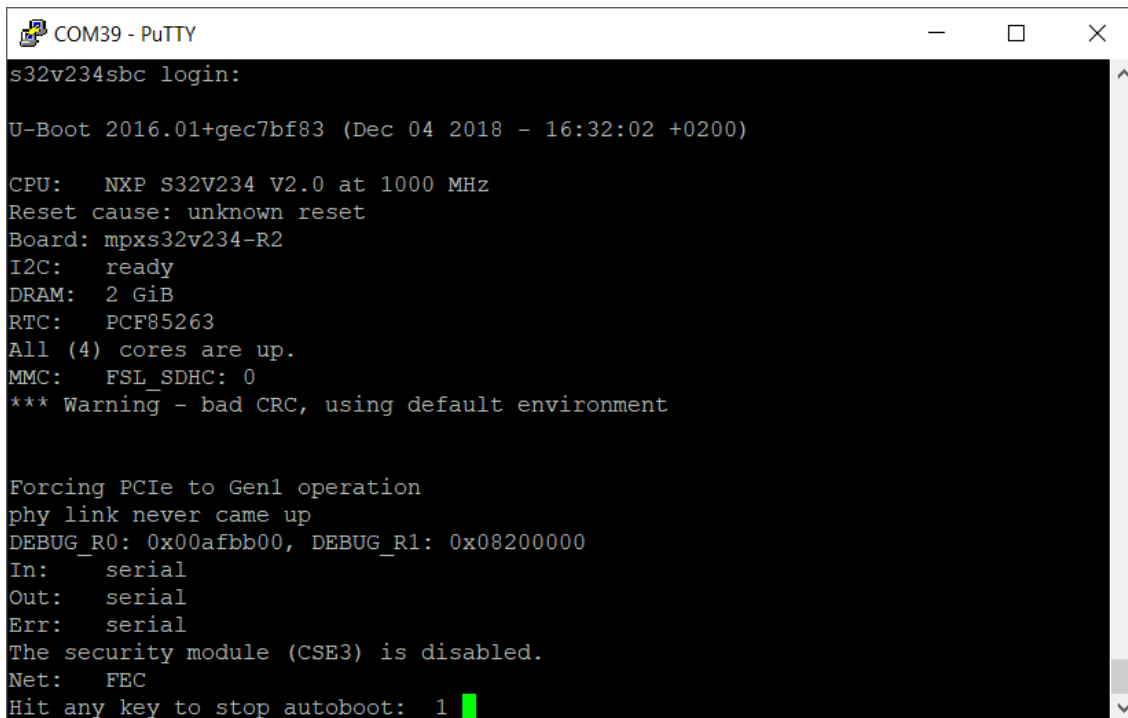
1. Download the USB to UART driver from <http://www.ftdichip.com/Drivers/VCP.htm>
2. Turn on the boards.
3. Install the Driver
4. Open Device Manager and find the COM port assigned



5. In Port Settings in the Driver's properties, set following settings: bits per second: 115200, data bits: 8, parity: None, stop bits: 1, flow control: None

After successful setup of the driver, it is possible to connect turned-on boards with console client application (e.g. Putty) on COM<number> and 115200 bps.

After resetting the S32V234 evaluation board you should be able to see the bootup process.

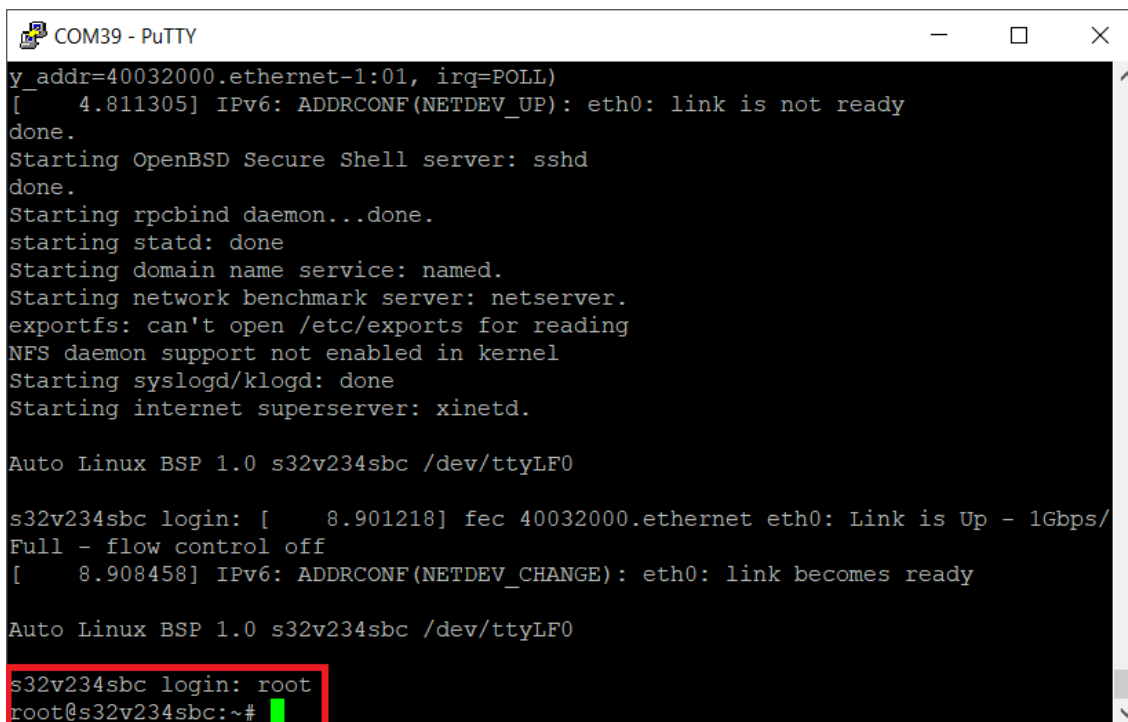


```
COM39 - PuTTY
s32v234sbc login:
U-Boot 2016.01+gec7bf83 (Dec 04 2018 - 16:32:02 +0200)

CPU:   NXP S32V234 V2.0 at 1000 MHz
Reset cause: unknown reset
Board: mpxs32v234-R2
I2C:   ready
DRAM:  2 GiB
RTC:   PCF85263
All (4) cores are up.
MMC:   FSL_SDHC: 0
*** Warning - bad CRC, using default environment

Forcing PCIe to Gen1 operation
phy link never came up
DEBUG_R0: 0x00afbb00, DEBUG_R1: 0x08200000
In:    serial
Out:   serial
Err:   serial
The security module (CSE3) is disabled.
Net:   FEC
Hit any key to stop autoboot: 1 █
```

Wait until the Linux is loaded and type root to log in.



```
COM39 - PuTTY
y_addr=40032000.ethernet-1:01, irq=POLL)
[ 4.811305] IPv6: ADDRCONF(NETDEV_UP): eth0: link is not ready
done.
Starting OpenBSD Secure Shell server: sshd
done.
Starting rpcbind daemon...done.
starting statd: done
Starting domain name service: named.
Starting network benchmark server: netserver.
exportfs: can't open /etc/exports for reading
NFS daemon support not enabled in kernel
Starting syslogd/klogd: done
Starting internet superserver: xinetd.

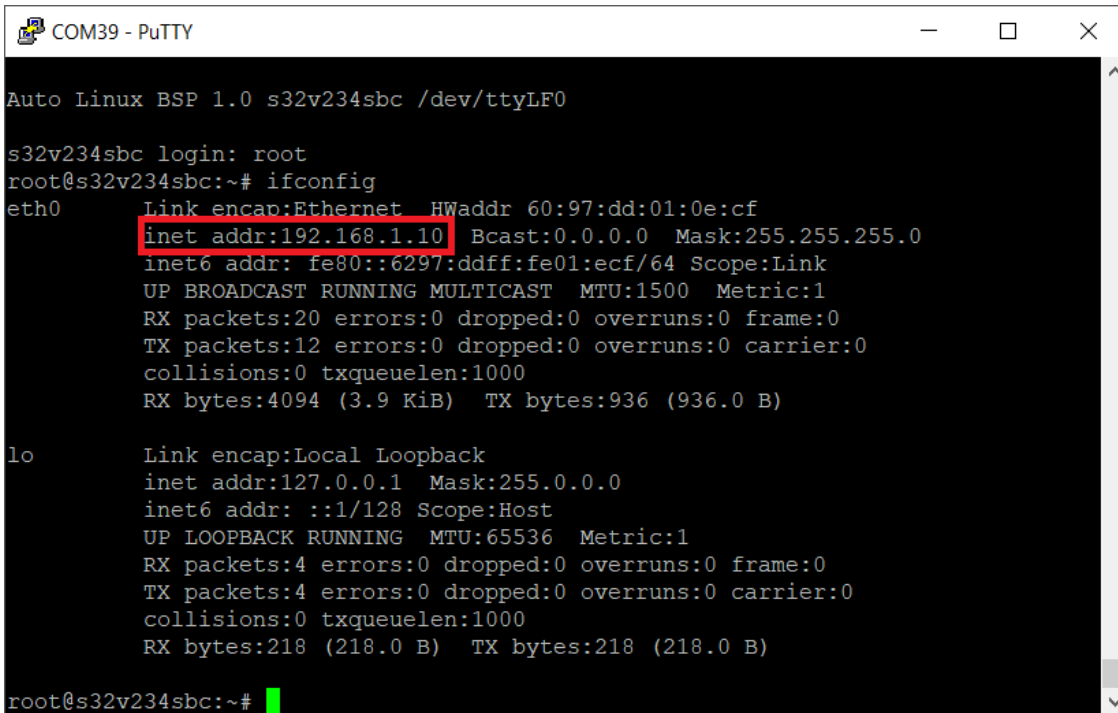
Auto Linux BSP 1.0 s32v234sbc /dev/ttyLF0

s32v234sbc login: [ 8.901218] fec 40032000.ethernet eth0: Link is Up - 1Gbps/
Full - flow control off
[ 8.908458] IPv6: ADDRCONF(NETDEV_CHANGE): eth0: link becomes ready

Auto Linux BSP 1.0 s32v234sbc /dev/ttyLF0

s32v234sbc login: root
root@s32v234sbc:~# █
```

Type `ifconfig eth0` to find the IP address of the board.



```
COM39 - PuTTY
Auto Linux BSP 1.0 s32v234sbc /dev/ttyLF0
s32v234sbc login: root
root@s32v234sbc:~# ifconfig
eth0      Link encap:Ethernet  HWaddr 60:97:dd:01:0e:cf
          inet addr:192.168.1.10  Bcast:0.0.0.0  Mask:255.255.255.0
          inet6 addr: fe80::6297:ddff:fe01:ecf/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:20 errors:0 dropped:0 overruns:0 frame:0
          TX packets:12 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:4094 (3.9 KiB)  TX bytes:936 (936.0 B)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:4 errors:0 dropped:0 overruns:0 frame:0
          TX packets:4 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:218 (218.0 B)  TX bytes:218 (218.0 B)

root@s32v234sbc:~#
```

This is the IP address needed to download the application from MATLAB to the S32V234 after code generation and build is completed.

NOTE For a step-by-step guide how to set up the S32V234 static IP, please check this [thread](#)

3.3.4 Compile and Run on S32V234

To generate the code and a face detection application on the S32V234 SBC begin by opening the `face_detection_camera_main.m` MATLAB script file from the `...examples/apps/face detection` folder:

```
function face_detection_camera_main() %#codegen
    width = uint32(1280);
    height = uint32(720);

    if coder.target('MATLAB')
        input = nxpvt.videoinput('winvideo', 1, width, height);
    else
        input = nxpvt.videoinput('sony', 1, width, height, true, false);
    end

    fdetector = nxpvt.CascadeObjectDetector('data/lbpcascade_frontalface.xml',
        'ScaleFactor',1.1, 'MinSize',[110 -1], 'MaxSize',[250 -1], 'SkipOdd',1,
        'MergeThreshold',2);

    fNum = int32(0);
    fps = int32(0);
    while true
        nxpvt.tic;
        fNum = fNum + 1;
        frame = input.getsnapshot();

        % Get faces.
        [bbox, l] = step(fdetector, frame);

        nxpvt.cv.rectangle(frame, bbox, [255, 0 ,0], 5);
        nxpvt.cv.putText(frame, sprintf('FPS: %d', fps), [10, 40],...
            'FONT_HERSHEY_SIMPLEX', 1, [0, 255, 0], 2);
        nxpvt.imshow(frame);

        fps = int32(fix(1/nxpvt.toc));
        fprintf(['%d] FPS: %d, Faces detected: %d, \n', fNum, fps, int32(l));
    end
end
```

Define a structure `config` that controls the behavior of the code generation and target configuration options. You can find all configuration fields explained in the next table.

Config fields	Description
MakeJobs	Number of CPU jobs used for parallel compilation
Optimize	When set to true, the cross compilers will use O3 optimization level. This ensure the best performance
Deploy	When set to true, the NXP Vision Toolbox will deploy the application on to the target after code generation stage. For the deployment to work the user needs to configure <code>config.TargetIpAddress</code> as well. If this is not set explicitly to true it will default to false

DeployPath	The path where the executable will be copied on the target. If left empty the '/examples/' folders will be used. This path should be an absolute path
TargetIpAddress	The IP address for the target
RemoteFilename	The name of the executable on the target. If left empty it will default to the entryFunc name with the elf extension instead of the .m extension
ExtraFiles	Files that are used by the elf (e.g.: videos, images). If left empty no extra files will be copied on the target. The paths for this file should be relative to the DeployPath

Here is an example for config structure configuration:

```
% Clear config structure
clear config

% Enables -O3 when you build the application. The application should run faster.
config.Optimize = true;

% Uses 8 make jobs when building the application. The build is faster.
config.MakeJobs = 8;

global TARGET_IP_ADDRESS;

if isempty(TARGET_IP_ADDRESS)
    warning("Target IP Address is not set. Please set the global TARGET_IP_ADDRESS
to the IP address of the board");
    config.Deploy = false;
else
    % Enables the deployment of the elf on the board.
    config.Deploy = true;

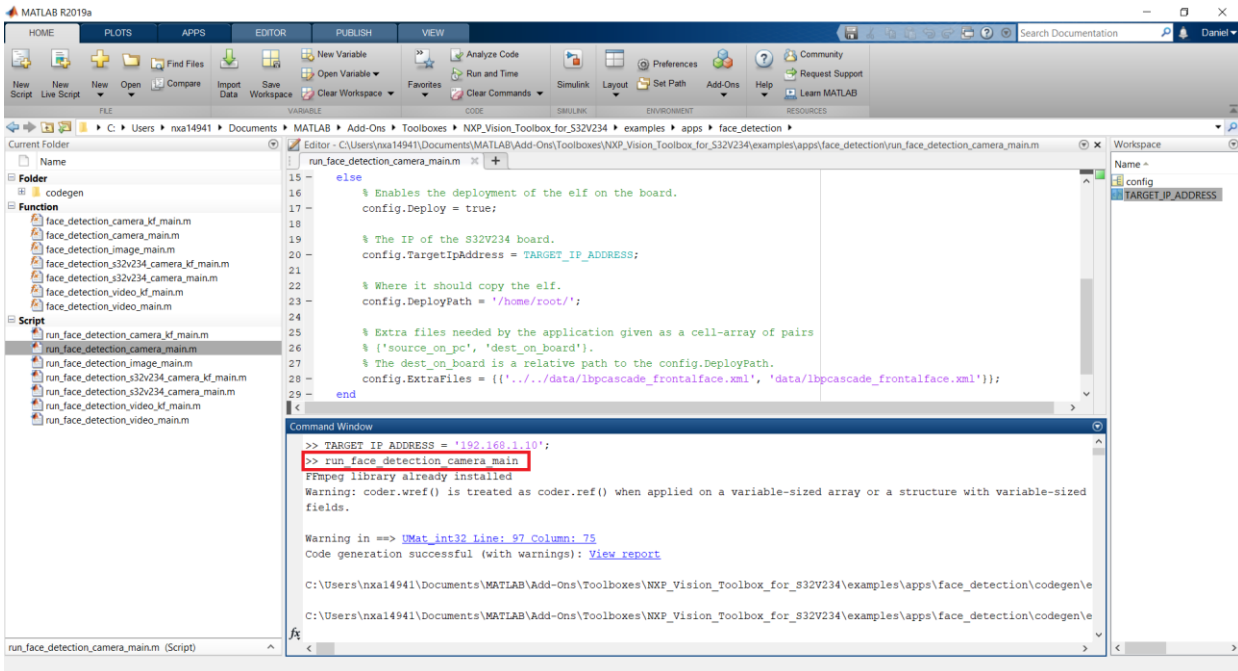
    % The IP of the S32V234 board.
    config.TargetIpAddress = TARGET_IP_ADDRESS;

    % Where it should copy the elf.
    config.DeployPath = '/home/root/';

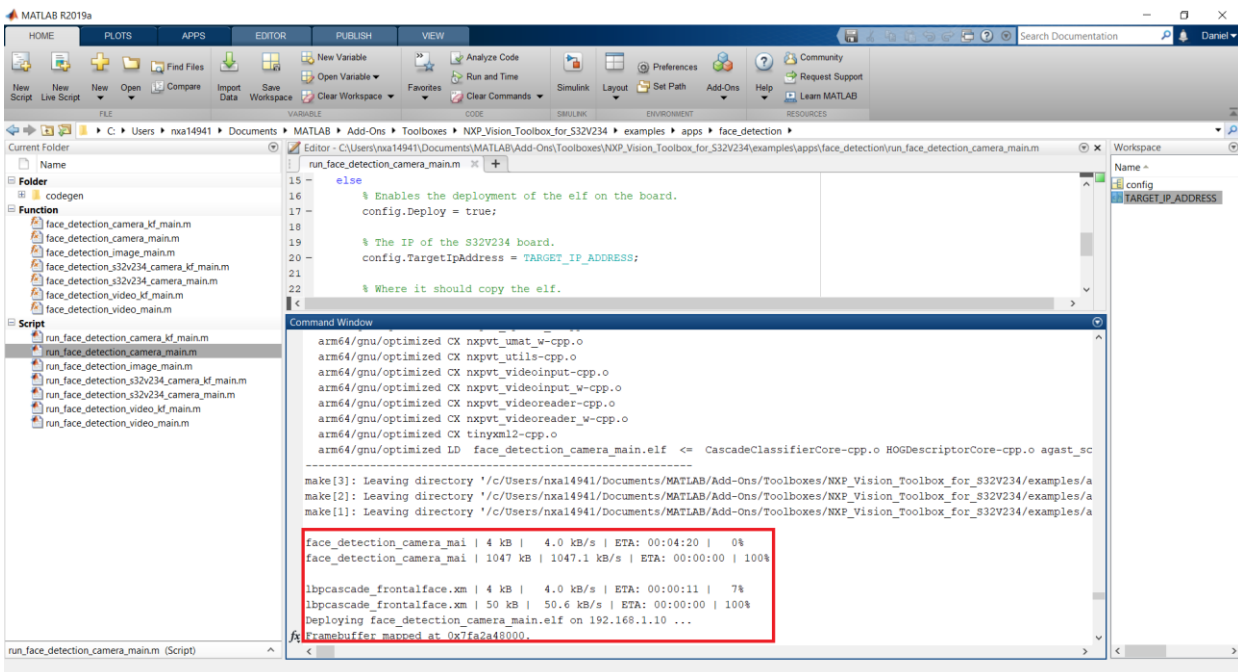
    % Extra files needed by the application given as a cell-array of pairs
    % {'source_on_pc', 'dest_on_board'}.
    % The dest_on_board is a relative path to the config.DeployPath.
    config.ExtraFiles = {'../data/lbpcascade_frontalface.xml',
'data/lbpcascade_frontalface.xml'};
end

nxpvt_codegen('face_detection_camera_main.m', config);
```

To start the code generation process, invoke `run_face_detection_camera_main.m` in MATLAB console.



After the application code generation and build is completed you should see this in your MATLAB Command Window.



The Vision Toolbox will automatically download the application to the target and you should be able to detect people's faces with NXP S32V234 on board camera



Congratulations! You succeeded with running your first example created with Vision Toolbox for S32V234

How to Reach Us:

Home Page:
www.nxp.com

Web Support:
www.nxp.com/support

Information in this document is provided solely to enable system and software implementers to use NXP Semiconductor products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.

NXP Semiconductor reserves the right to make changes without further notice to any products herein. NXP Semiconductor makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale Semiconductor assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters that may be provided in NXP Semiconductor data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including "Typicals", must be validated for each customer application by customer's technical experts. NXP Semiconductor does not convey any license under its patent rights nor the rights of others. NXP Semiconductor products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the NXP Semiconductor product could create a situation where personal injury or death may occur. Should Buyer purchase or use NXP Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify and hold NXP Semiconductor and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that NXP Semiconductor was negligent regarding the design or manufacture of the part.

MATLAB, Simulink, Stateflow, Handle Graphics, and Real-Time Workshop are registered trademarks, and TargetBox is a trademark of The MathWorks, Inc. Microsoft and .NET Framework are trademarks of Microsoft Corporation. Flexera Software, FlexIm, and FlexNet Publisher are registered trademarks or trademarks of Flexera Software, Inc. and/or InstallShield Co. Inc. in the United States of America and/or other countries.

NXP, the NXP logo, CodeWarrior and ColdFire are trademarks of NXP Semiconductor, Inc., Reg. U.S. Pat. & Tm. Off. Flexis and Processor Expert are trademarks of NXP Semiconductor, Inc. All other product or service names are the property of their respective owners

©2019 NXP Semiconductors. All rights reserved.