S32 Design Studio 3.2

Contents

1. Release Description

NXP Semiconductors is pleased to announce the release of the S32 Design Studio 3.2 for NXP Arm® based devices and hardware accelerators. S32 Design Studio is based on the Eclipse open development platform and integrates the Eclipse IDE, GNU Compiler Collection (GCC), GNU Debugger (GDB), and other software to offer designers a straightforward development tool with no code-size limitations.

1.1. Release content

- Eclipse Neon 4.6 Framework
- GNU Bare-Metal Targeted Tools for Arm® 32-bit Embedded Processors (GCC version 6.3.1 20170509, build 1620 revision g01b30c3)
- GNU Bare-Metal Targeted Tools for Arm® 64-bit Embedded Processors (GCC version 6.3.1 20170509, build 1620 revision g01b30c3)
- GNU Linux Targeted Tools for Arm® 64-bit Embedded Processors (GCC version 6.3.1 20170509, build 1620 revision g01b30c3)
- Libraries: NewLib, NewLib Nano, EWL, and EWL Nano [1]
- Semihosting for Arm® 32-bit and 64-bit bare-metal target toolchains
- MSYS2 32bit version 1.0.0
- GDB 7.12.1 with Python support
- S32 Flash Tool
- S32 Configuration Tool framework with the Pin, Clock, Peripheral, DCD, IVT, and DDR Configuration tools
- The wizards for creating application, library projects and projects from project examples for the supported processor families[2]
- The S32DS Extensions and Updates tool
- S32 Trace Tool
- S32 Debugger support
- PEMicro® debugger support
- Lauterbach Trace32® support
- Green Hills compiler support
- EmbSys Registers view
- SDK management
- Support for importing MCAL configuration to a custom SDK
- The Getting Started page

2. What's New

This release comes with several new features, improvements and bug fixes.

[1]The availability of libraries depends on the toolchain version and the core type. Tools for Arm® 64-bit processors support NewLib libraries only. Tools for Arm® 32-bit processors do not support EWL libraries for the Cortex-R52 and Cortex-M33 cores.
[2]Support for wizards and the project examples are provided in the device specific software packages.

2.1. New features

- Multiple builds of the GNU tools can be installed.
- The SDK integration is provided with additional software packages. The SDK packages can be installed and updated with the S32DS Extensions and Updates tool.
- The GHS toolchain support is provided by the project wizard (available for particular devices).
- The CMSIS-DAP debugging is supported by particular devices.
- The UART communication speed of S32 Flash Tool is improved.
- The latest versions of P&E debugger plug-in and drivers are provided.
- S32 Debugger provides OS Awareness support for FreeRTOS and OSEK.
- S32 Debugger provides preliminary support for secure debugging with the Password and Challenge/Response authentication methods.
- New type of Launch Groups is defined for S32 Debugger, which allows you to create S32 Debugger specific configurations and use the new "Wait for stop on breakpoint" post launch action.
- The SDK migration support is provided to upgrade an SDK version attached to the project.
- The S32DS Extensions and Updates tool notifies about dependencies and incompatible packages.
- The progress bar is displayed for the S32 Debugger flash programmer.

2.2 Bug fixes

- An SDK can now be attached to an existing project without errors.
- The installer works properly after entering invalid activation code.
- Hyperlinks now work correctly on Windows 10 when Microsoft Edge is configured as the default browser.

3. System Requirements for Windows Host

3.1. Recommended configuration

- PC with 2.6 GHz Intel® Pentium® compatible processor or better
- 4 GB of RAM
- 4.5 GB of disk space
- 3 GB of temporary storage (required only during the product installation)
- USB port for communications with target hardware
- Ethernet port for communications with target hardware (optional)

3.2. Operational minimum configuration

- PC with 1.8 GHz Intel® Pentium® compatible processor
- 2 GB of RAM

- 4.5 GB of disk space
- 3 GB of temporary storage (required only during the product installation)
- USB port for communications with target hardware

3.3. Host operating system support

- Microsoft® Windows® 7 64-bit
- Microsoft® Windows® 8/8.1 64-bit
- Microsoft® Windows® 10 64-bit

S32 Design Studio 3.2 supports all editions of the operating systems listed above and is limited only by the requirements of the Java Runtime Environment.

4. System Requirements for Linux Host

4.1. Recommended configuration

- PC with 2.6 GHz Intel® Pentium® compatible processor or better
- 4 GB of RAM
- 4 GB of disk space for installation files (full product or updates)
- 2.5 GB of temporary storage (required only during the product installation)
- Java 1.8
- GCC 5.x
- USB port for communications with target hardware
- Ethernet port for communications with target hardware (optional)

S32 Design Studio 3.2 Installation Guide specifies the installation prerequisites for Linux platforms.

4.2. Operational minimum configuration

- PC with 1.8 GHz Intel® Pentium® compatible processor
- 2 GB of RAM
- 4 GB of disk space for installation files (full product or updates)
- 2.5 GB of temporary storage (required only during the product installation)
- Java 1.8
- USB port for communications with target hardware

4.3. Host operating system support

- Ubuntu LTS 16.04 64-bit
- Debian 8 64-bit
- CentOS 7 64-bit

S32 Design Studio 3.2 supports all editions of the operating systems listed above and is limited only by the requirements of the Java Runtime Environment.

5. Product Web Page

S32 Design Studio 3.2 is available for download from the product Web page at www.nxp.com.

6. Installation and Licensing

The S32 Design Studio 3.2 installation package contains the base tools and the installer. Support for the NXP Arm® based processor families is provided with additional software packages. The packages can be installed on the product from the S32DS Extensions and Updates tool or downloaded from the product Web page.

Note: The plug-ins to support third-party compilers or debuggers such as Lauterbach Trace32® are not included in the installation package and have to be installed from the corresponding sites or installation packages.

Run the installation package. The wizard will guide you through the installation process.

Note: Installation of S32 Design Studio 3.2 from the command line interface in the console or silent mode is not supported.

During installation, license activation is requested. The following activation types are supported:

- Online activation requires access to the Internet. A license activation request is sent automatically.
- Offline activation requires no Internet access. A license activation request is generated and needs to be passed to the licensing site manually. The activation response is loaded from the site to the installer manually.

New functionality can be added to S32 Design Studio 3.2 with additional software packages, updates and patches. Software packages add support for specific NXP Arm® based processor families. Updates and patches extend general functionality of the product and correct software defects.

New functionality can be added directly from the Internet or from a downloaded archive. If your computer is connected to the Internet, select **S32DS Extensions and Updates** on the **Help** menu to find and install all updates and packages available. If your computer does not have access to the Internet, you can download software packages, updates, and patches from the product page and install them from the archive files using the S32DS Extensions and Updates tool.

7. Starting S32 Design Studio

To start S32 Design Studio and begin to work with it:

- In Windows, double-click the product icon on the desktop or go to the Start menu and click **NXP S32 Design Studio** > **S32 Design Studio 3.2**. To run the product from the command line, open the terminal, navigate to the directory with the installed product, type `s32ds.bat`, and press `Enter`.

S32 Design Studio 3.2, Release Notes, Rev. 1.0, 09/2019

- In Linux, to run the product from the command line, open the terminal, navigate to the directory with the installed product, type `./s32ds.sh`, and press `Enter`.

8. Technical Support

- S32 Design Studio general issues are tracked through the S32DS Public NXP Community space:

  https://community.nxp.com/community/s32/s32ds

- For confidential cases and cases which cannot be publicly shared on NXP Community please follow the steps described here:

  https://community.nxp.com/docs/DOC-329745

- Submit a support request for P&E Microcomputer Systems:

  https://www.pemicro.com/support/

9. Related Documentation

To learn more about the included tools, refer to the following Release Notes:

- S32 Debugger Release Notes located in `/S32DS/tools/S32Debugger/Debugger/`
- S32 Configuration Tool Release Notes located in `/Release_Notes/`
- S32 Flash Tool Release Notes located in `/S32DS/tools/S32FlashTool/doc/`
- GNU Bare-Metal Targeted Tools for Arm 32-bit Embedded Processors Release Notes located in `/S32DS/build_tools/gcc_b1620/gcc-6.3-arm32-eabi/`
- GNU Bare-Metal Targeted Tools for Arm 64-bit Embedded Processors Release Notes located in `/S32DS/build_tools/gcc_b1620/gcc-6.3-arm64-eabi/`
- GNU Linux Targeted Tools for Arm 64-bit Embedded Processors Release Notes located in `/S32DS/build_tools/gcc_b1620/gcc-6.3-arm64-linux/`

Appendix A. Known issues and Workarounds

- **Conditional watchpoints and breakpoints**: Conditional breakpoints and watchpoints, including those using ignore counts, may not work in some cases.

  Workaround: Avoid using conditions for breakpoints and watchpoints, instead check for condition in the code and set a normal breakpoint.

- **Disassembly view** content might be destroyed occasionally.

  Workaround: Close the Disassembly view and open it again by selecting **Window** > **Show View** > **Disassembly** on the menu.

- **Hyperlinks** may not work correctly when Microsoft Edge is configured as the default browser.

  Workaround: Set a different browser as the default one.

- **Views** are not updated instantly after executing a command in the GDB console.

Workaround: Use the `step` command to see refreshed views.

- The **Getting Started** page may be displayed blank in Linux.

  Workaround: Install Webkit1 for GTK2 using the following command: `sudo apt-get install libwebkitgtk-1.0-0`

- The **Getting Started** page may fail to load content at startup of S32 Design Studio 3.2. " The page can't be displayed " error message may be shown instead.

  Workaround: Refresh the **Getting Started** page by pressing the F5 key.

- **Duplicated error dialog**: When adding more watchpoints than supported by the device, the popup box with the " Not enough hardware resources for processing " error message is displayed twice.

  Workaround: Close the popup box, then close it again.

- **Stepping over the try-catch block fails on a VDK**: When debugging a project on a VDK, an attempt to step over the try-catch block fails.

  Workaround: The issue is specific for projects compiled with NewLib. Recreate a project and select NewLib Nano as the library.

- **Watchpoints set on complex data types are not hit**: When debugging on a target connected with S32 Debug Probe, a debug session ignores a watchpoint set on a variable of a complex data type (such as a structure or other). A watchpoint set on an item of a basic data type inside a complex variable works correctly.

  Workaround: Avoid setting watchpoints at complex data types.

- **USB connection failure when debugging with S32 Debug Probe**: A debug session may fail when S32 Debug Probe is connected through USB. Debugging with S32 Debug Probe connected through Ethernet can be done successfully.

  Workaround: Connect the probe to USB. Set up the debug configuration to use the Ethernet connection rather than USB. Specify the virtual IP address of the probe in the connection settings of the debug configuration. To obtain the required value, depending on the platform run the following command:

```
sudo ifconfig -a
```

```
ipconfig -all
```

The displayed output includes a section for the virtual Ethernet interface created for communication with the probe over the physical USB link. For example, the *inet addr* (or *IPv4 Address*) value in this section is a local IP address starting with " 169.254 " . Take the *HWAddr* (or *Physical Address*) value in the section and convert the last two bytes into decimal notation. Append the resulting numbers to " 169.254 " to obtain the required virtual IP address.

- **Unable to install the VP Explorer plug-in**: When launched, the S32 Design Studio installer looks for the existing Virtualizer Runtime installation to install the VP Explorer plug-in. The latest version of Virtualizer Runtime is not detected.

  Workaround: Install the VP Explorer plug-in manually. On the main menu, choose **Help > Install New Software** from the menu bar, click **Add**, then click **Archive**. Navigate to the `<synopsys_version>/VirtualizerRuntime/SLS/windows/vpexplorer_feature` directory and choose the `vpexplorer-feature.zip` file. The required plug-in will appear in the list, select it and click **Next** to proceed the installation.

- **Debugging on a VDK may fail**: Some errors may occur due to unexpected termination of a debug session or unsupported registers and watchpoints, etc.

  Workaround: Run a debug session in Virtualizer Studio.

- **Creating new project in the root of drive C:\ fails**: When creating a new project in the root directory, the project wizard shows the "Can't create directory: C:\project_name" error message. Windows does not allow user to create files in this location.

  Workaround: Create a subdirectory in the root directory (for example, C:\projects) and specify this location in the project creation wizard.

Note: There are some issues which are introduced by Eclipse CDT and are therefore reproduced in the S32 Design Studio. These issues might be fixed when the fix is available in the future version of Eclipse CDT and the S32 Design Studio migrates to the updated CDT.

Appendix B. Performance Considerations

The following suggestions will help you keep the S32 Design Studio tools running at a respectable performance level.

1. To maximize the performance, the S32 Design Studio tools should be installed on a computer with the recommended configuration. While the tools will operate on a computer with the operational minimum configuration, limited hardware will restrict its ability to function at desired performance levels.
2. Close unused projects. Eclipse caches files for all open projects in the workspace. If you need multiple projects open, try to limit the number of projects to no more than 10.
3. The Eclipse IDE provides several options that provide user assistance tools. These options, however, use memory and CPU resources. If the performance is slow, and you do not need these options, turn them off.
4. Scalability mode options configure how Eclipse deals with large source files and may affect the overall performance of the IDE. Turn off these options to maximize the performance. The following scalability mode options are available: editor live parsing, semantic highlighting, syntax coloring, parsing–based content assist proposals, and content assist auto activation.

   To disable:

   a. Choose **Window > Preferences**
   b. Expand **C/C++ > Editor > Scalability**

   c. Uncheck **Enable all scalability mode options**

5. Content Assist Auto Activation can reduce the number of keystrokes one must type to create the code. The Content Assist plug-in consists of components that predict what keystroke you may type next, based on the current context, scope and prefix. Turn off this predictor to maximize the performance.

   To disable:

   a. Choose **Window** > **Preferences**
   b. Expand **C/C++** > **Editor** > **Content Assist**
   c. Uncheck all the options for **Auto-Activation**

6. For faster serial communication response (for example, UART connection), you can change the OS latency timer settings. The latency timer is a form of time-out mechanism for the read buffer of FTDI devices. The default value for the latency timer is 16 ms. You can decrease the latency value to maximize the performance:

   - On Windows, go to the port properties, open advanced port settings and select the required latency value from the drop-down menu.
   - On Linux, open the terminal and set the required value using the `echo` or `setserial` commands. For example:

   ```
   echo 2 > /sys/bus/usb-serial/devices/ttyUSB0/latency_timer
   ```

S32 Design Studio 3.2, Release Notes, Rev. 1.0, 09/2019