New Release of DAVE™ version 4 and DAVE™ SDK version 4

This document is mainly intended for existing users of DAVE™ version 3. It provides a brief overview of the main user relevant changes from DAVE™ version 3 to DAVE™ version 4 and a very brief introduction to the DAVE™ SDK. It also provides an overview of the concept of the DAVE™ APPs for DAVE™ version 4 and covers the newly introduced low level peripheral drivers, the XMC lib.

General release information
DAVE™ v4.0.0, DAVE™ SDK v4.0.0 and the all DAVE™ APPs 4.0.x are still under system test and code review, therefore released as Beta version to be used for evaluation and test purposes or adaption in development flow.

Feedback is appreciated, please send feedback to: dave@infineon.com

Sections in this Document

1. Overview of new features in DAVE™ v4.0.0
   1.1. Nano lib
   1.2. New project outline
   1.3. New APP selection view
   1.4. New DAVE APP Dependency View
   1.5. New DAVE APP HW Signal Connectivity View
   1.6. New graphical pin mapping functionality
   1.7. New Global Interrupt Editor
   1.8. New widgets in the Configuration UI
   1.9. Copy and paste of DAVE APP Configurations
   1.10. New instance label of DAVE APPs
   1.11. Better management of resource conflicts
   1.12. APP rating and feedback
   1.13. New Debugger

2. Concept and guidelines of DAVE™ APPs for DAVE™ v4

3. Compatibility between DAVE™ v3 and DAVE™ v4

4. Quick Start tutorial for DAVE™ v.4.0.0

5. Information about DAVE™ SDK, Quick Start tutorial for DAVE™ SDK

6. Overview of XMC Peripheral Drivers, XMC Lib

German speaking people may [download] a recent article of DAVE™ version 4.
1. Overview of improvements and new features in DAVE™ v4.0.0

In a nutshell DAVE™ v4 provides the following main improvements:

- Faster and more robust design: In particular the time for code generation, responses on user interactions in the graphical user interfaces and build time could be improved significantly.
- DAVE APPs are now based on low level peripheral drivers (XMC Lib) which makes generated code more efficient and more readable.
- A DAVE SDK is provided as separate tool: Now everyone can develop own DAVE APPs or modify existing APPs, use cases are not restricted and DAVE SDK can be used to develop a configuration APPs for any library.
- A set of low level peripheral drivers (XMC Lib) is now available and is used by DAVE APPs. The XMC Lib can also be used independent of DAVE or DAVE APPs with any other tool chain that supports XMC (Atollic, Altium, Keil, IAR, Rowley)

Following a more detailed list of valuable improvements:

1.1 Nano Lib is now included in the gcc toolchain which offers higher code density for microcontroller type of software.

This is reflected in the project settings: DAVE v4 now provides the option to include ‘nano lib’ and to improve default settings to accelerate compilation time.

1.2 New project outline

The project in DAVE v4 are now as much self-contained as possible. This means all CMSIS header files, all device specific low level drivers, start-up files and linker script are copied into the project. Also, all DAVE APPs will be fully included. The only remaining references are the standard header files of the compiler tool chain and the device model (Device Descriptions) that is used by the resource solver. With this approach a qualified project can be more easily frozen and archived.

As the project may contain more files and functions than required by the application, the project option “remove unused section” should be used to prevent dead code from eating up memory space.
1.3 New DAVE APP Selection View

The APP selection view with improved filter and categorisation is now implemented as a modal view. This should make it easier to find the required DAVE APPs and save screen space when APP selection is not required.

The New APP selection view can be activated from any APP Dependency /Connectivity Views or the tool box bar. It should be closed after the required APPs are selected. It also includes, now, hidden categories for APPs that are normally not explicitly selected; e.g. APPs to configure Global registers.

1.4 New DAVE APP Dependency View

To improve the visualization of dependencies between DAVE APPs the APP Dependency view has been reworked in the following manner:

- Layered positioning of DAVE APPs with as many layers as required
- The top level APP of a dependency tree is positioned at the top, the next dependent DAVE APP on the next lower level and so on. The lowest level APP of a dependency tree is positioned on the lowest level of the view.
- The APPs cannot be moved outside of their assigned layer.
- The APPs can be ordered within the layer by the user, this order will be kept, also after starting the auto layout functionality.
- A new APP is always added from the left.

When a large number of DAVE APPs are instantiated, the APP Dependency TreeView can be used to select an APP - Then the Dependency will focus on the selected APP (dark blue) plus its dependency connections.
1.5 New DAVE APP HW Signal Connectivity View

The concept of the HW Connectivity view has been changed to a schematic circuit style. Each APP is now represented as an object with all input and output signals displayed. Inputs on the left side, outputs on the right side. The green lines represent connections made by the manifest and cannot be modified by the user; the blue lines are user defined lines. The purposes of this view is to visualize signal and event connections graphically, the user cannot do any changes in this view.

![HW Signal Connectivity View](image)

1.6 New graphical pin mapping functionality

In addition to the existing table view, for manual pin assignments a graphical pin mapping functionality for manual pin assignments has been implemented within its own dedicated perspective called PinMapping. This perspective can be opened by selecting:
>
- Window -> Open Perspective -> PinMapping

Or by pressing the pin mapping tool box button in the tool box bar.

The resource list on the left shows all APPs and signals than can be mapped to a pin. When selecting a signal in the resource list all possible pins are marked in green in the graphical view, assignment can be done by right mouse click on a green pin. When the pin is selected it is highlighted in light blue; it can be unassigned by right mouse click on the pin. All pins manually assigned to other signals are indicated as dark blue.

There is a specific visualization options that shows all pins that are assigned by the solver (orange) these pins have not been manually assigned by the user but have been assigned by the solver and user may check whether this fits to the board constraints.

![Virtual Pin View](image)
1.7 New Global Interrupt Editor
A new view has been introduced to show all defined interrupts at one glance. This Global Interrupt Editor allows changing the priority of all defined interrupts in one global UI.

1.8 New widgets in the Configuration UI
The new APP SDK allows modelling of advanced widgets for the configuration UI including rich text information and bitmap graphic. This leads to more appealing UIS of the DAVE APPs.

1.9 Copy and paste of APP Configurations
The existing configuration of any APP can be copied to another instance of this APP in the same project by using the new “Copy and paste” functionality which is accessible upon right mouse click on an APP. The configuration of an APP can also be exported to a file (XML) and imported from there to another instance of this APP in any project.

1.10 New instance label of APPs
DAVE v4 provides the option to change the default instance label to a user defined symbol. This symbol can then be used in the APIs of the APP to refer to the respective APP instance.

With this user code can easily ported to other projects that use a certain set of APPs. It is just required that both projects use the same user defined instance label.

DAVE APP with the default user label that is derived from the APP name plus instance suffix.
Via a right mouse click option the instance label can be redefined to a user defined label, here "MyPWM"

This user defined label can then be used in the code to reference the used API to the used instance:

\[
\text{PWM\_SetDutyCycle(&MyPWM, <integer value>);}
\]

1.11 Better management of resource conflicts
In DAVE v4 the resources solver is now executed on each action that is relevant to the required chip resources. With that it is easy to identify potential conflicts and allow to user to take appropriate actions to avoid such conflicts.

1.12 APP rating and feedback
Upon right mouse on a DAVE APP there is an APP rating implemented. It allows rating the value of DAVE APPs and providing the options to add comment, e.g. what is not good and should be improved.

We appreciate such feedback very much because it will help us to further optimize the DAVE APPs and to define new DAVE APPs driven by such inputs.

1.13 New Debugger
The TASKING debugger has been replaced by a CDT debugger from gnuarmeclipse. This debugger uses the Segger GDB server and requires a J-Link debug probe to connect to a XMC target. Almost all XMC kits include a J-Link on board debugger. The Infineon DAP miniWiggler is not be supported any more.

2. Concept and guidelines of DAVE™ APPs for DAVE™ v4

DAVE APPs are application components that abstract a certain use case. There is no change in the general philosophy of DAVE v4 APPs in comparison to DAVE v3.

However, there are two major changes in the implementation concept of DAVE v4 APPs.
2.1 DAVE v4 APPs are built on top of the newly introduced peripheral specific low level drivers (XMC Lib), see also picture of the XMC Lib section below. With this, DAVE v4 APPs generally do not directly access registers of the MCU but use the API of the XMC Lib to initialize HW. Hence DAVE v4 APPs essentially generate the configuration data structure based on the selected configuration options and call the appropriate ‘init’ functions. Depending on the use case of the DAVE v4 APPs, higher level API functions are created to implement the defined use cases, by using the APIs provided by the LLDs. With this, code generation becomes more transparent and robust and generated code can easily be qualified to be used in production.

2.2. New Domain Specific Language (DSL) to define the GUI, GUI actions, instantiation actions, and code generation actions. Groovy has been chosen as the DSL to define DAVE v4 APPs, related control functions and code generation templates. With this, the GUI design to configure DAVE APPs is more flexible and the APP design is more robust.

2.3 Consumption of resources DAVE APPs for DAVEv4 should include / consume all resources and functionalities that are essentially belong to the use case of the APP. Separation to various application components should only be done if there is a good re-uses benefit. This leads to less number of APPs that are required in a project, visible by the fact that now I/O functionality and interrupt nodes included in the APP whenever I/O functionality and an interrupt node is required to fulfill the uses case. Although, in most of such cases flexibility to do an own respective composition is still possible.

2.4 Improved Concept of DAVE APPs. Most of The new DAVE APPs are also conceptually redesigned to improve ease of use but also to improve their usability in productive code.

The code generated by the DAVE APPs is fully compliant and tested with these compiler tools: GCC, ARM, IAR, TASKING.

3. **Compatibility between DAVE™ v3 and DAVE™ v4**

Because of the concept changes of DAVE APPs describe in section 7 and the changes in the data model of DAVE APPs to improve speed and robustness projects that includes DAVE APPs are not compatible between DAVE v4 and DAVE v4. However, it is possible to manually migrate projects from DAVE v3 to DAVE v4, more details are described in the DAVE user manual that can be found in help content in DAVE. On the other hand there is no real ned to migrate old projects because DAVE v3 and DAVE APPs for DAVE v3 will still be available and maintained for a sufficient period of time.

4. **Quick Start tutorial for DAVE™ v4**

Those people who have already worked with DAVEv3 may realize the working with DAVE v4 follows the same philosophy, but the described changes should improve the user experience significantly and should bring more fun in embedded software engineering. To gain some first own experience we have developed a quick start tutorial that creates the famous blinky of a LED using any XMC kit. The tutorial is a separate document and can be downloaded from here.
5. **DAVE™ SDK Overview and Quick Start**

It is now possible to offer an SDK for DAVE APPs that makes developing or changing of DAVE APPs much simpler than before.

A DAVE APP essentially consists of 2 elements:
- The Manifest that contains the properties, the GUI design, GUI functions, Instantiation actions and code generation actions, the manifest language is grove with domain specific extensions
- Code templates for script based code generation and/ or static files, the code templates are also based on groovy script.

Writing the Manifest is supported by offering:
- GUI designers with drag and drop functionality for widgets and wysiwyg preview of the designed GUI
- Professional SW editors with syntax highlighting and completion functions for efficient coding
- Comprehensive checks for types and other errors
- Debugging features

Writing the code templates is supported by offering:
- Convenient APIs to get GUI and device data
- Debugging features

The DAVE APP SDK is open to be used for any use case. One valuable use case would be to develop DAVE APPs to create the configuration code for any static library based on an easy to use graphical user interface, or to create configuration files to customize software.

Tutorials are included in the DAVE SDK user manual the can be found in the DAVE SDK help content. Or [download a tutorial](#) as separate pdf file.

The DAVE APPs projects will be provided from a Git repository in a time frame mid-2015. They can be also obtained earlier by sending a specific request to [dave@infineon.com](mailto:dave@infineon.com).
6. **Overview of XMC Peripheral Drivers, XMC Lib**

In addition to the application components DAVE APPs we will also offer a set of XMC peripheral drivers. The libraries of the XMC Lib are peripheral IP oriented not specific to individual devices. Device specific deviation of a IP is handled by conditional compilation.

Three possible usage models exist along with DAVE:
- Used by DAVE APPs to improve robustness and transparency of code generation
- Standalone (independent of DAVE APPs) to simplify the usage of the XMC microcontrollers in case DAVE APPs are not adequate
- Mixed use of DAVE APPs and LLD, e.g. if there is no APP for the required Application use case the LLD can be used. With the APIs of the LLDs the full flexibility of peripherals and connectivity can be used instead of accessing the registers directly

The XMC Lib can also be used independent of DAVE, along with all compiler tools supporting the XMC family which is Keil MDK, IAR, Atollic, TASKING, Rowley or other GCC based tool chains. The LLDs are integrated in the latest CMSIS PACK for Keil and in the getting started of IAR.

[Download LLDs as separate package]

---

**Legal Disclaimer** The information given in this document shall in no event be regarded as a guarantee of conditions or characteristics. With respect to any examples or hints given herein, any typical values stated herein and/or any information regarding the application of the device, Infineon Technologies hereby disclaims any and all warranties and liabilities of any kind, including without limitation, warranties of non-infringement of intellectual property rights of any third party.

**Information** For further information on technology, delivery terms and conditions and prices, please contact the nearest Infineon Technologies Office (www.infineon.com).

**Warnings** Due to technical requirements, components may contain dangerous substances. For information on the types in question, please contact the nearest Infineon Technologies Office. Infineon Technologies components may be used in life-support devices or systems only with the express written approval of Infineon Technologies, if a failure of such components can reasonably be expected to cause the failure of that life-support device or system or to affect the safety or effectiveness of that device or system. Life support devices or systems are intended to be implanted in the human body or to support and/or maintain and sustain and/or protect human life. If they fail, it is reasonable to assume that the health of the user or other persons may be endangered.