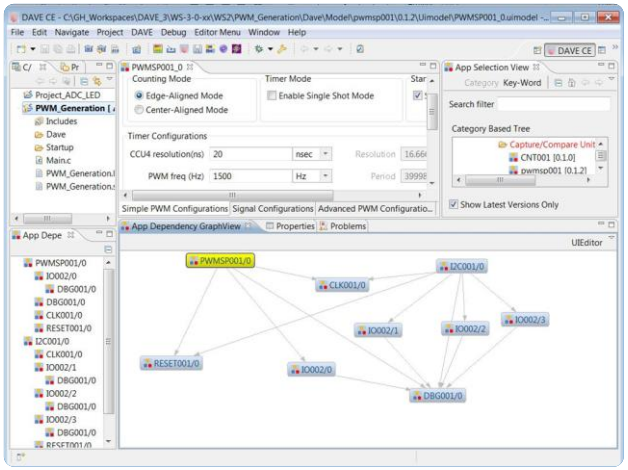
DAVE™ Overview

March, 2013

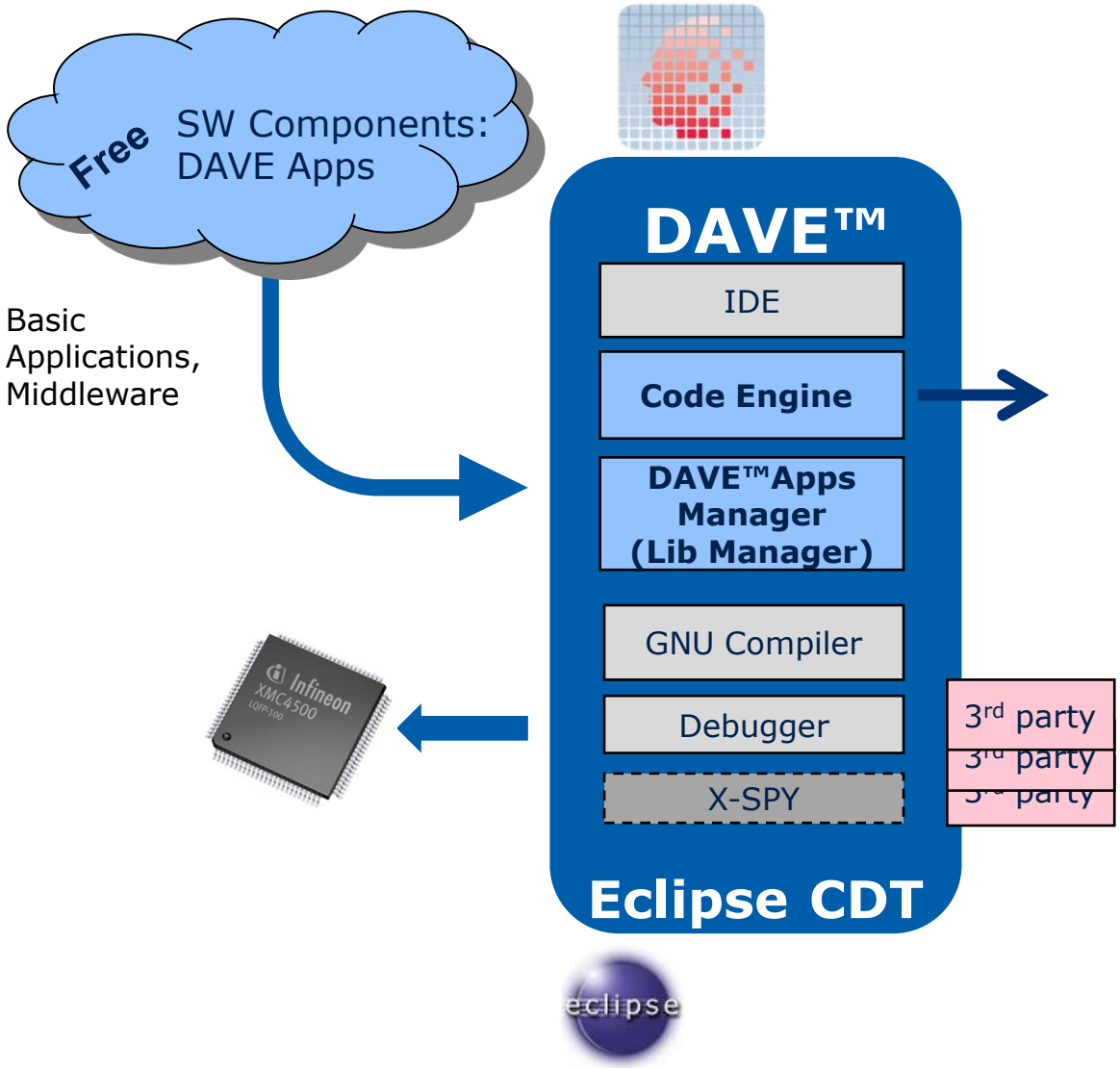
What is DAVE™ version 3?

- DAVE™ version 3 is a free and high productivity development platform for code generation based on predefined SW components: DAVE™ Apps
- With DAVE™ developers can easily generate a software library to efficiently use the innovative application-optimized peripherals of the XMC1000 and XMC4000 microcontrollers



- DAVE™ includes:
  - Eclipse CDT based IDE with improved project management
  - GNU C-Compiler tools
  - Debugger incl. Flash loader
  - Code generation plug in with graphical user interfaces
  - A resource solvers provides automatic or constrained assignment of chip resources to the DAVE™ Apps
  - Library manger to download and manage the DAVE™ Apps
  - Data visualization
  - Can be used with 3 rd party tools and SW
  - DAVE™ version 3 supports the XMC1000 and XMC4000 family

The DAVE™ Development Platform



Library code generation based on predefined and tested SW components (DAVE™ Apps)

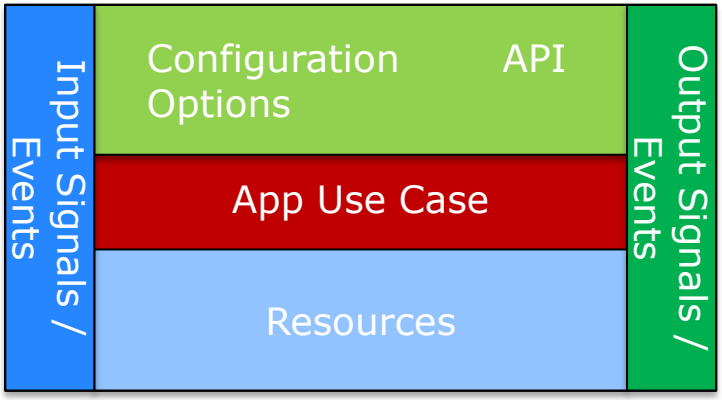
Option for third parties to plug in other commercial or free tools

DAVE™ Apps abstract Application Use Cases

- DAVE™ Apps are SW components or SW building blocks for a specific **App**lication use case
- Unlike the concept of static Libraries, DAVE™ Apps do not only provide just an API for a simple HW abstraction
- DAVE™ Apps are autonomous building blocks that can abstract any simple or complex use case

Configuration e.g. to determine the initialization status

API used in the user code (like a library)



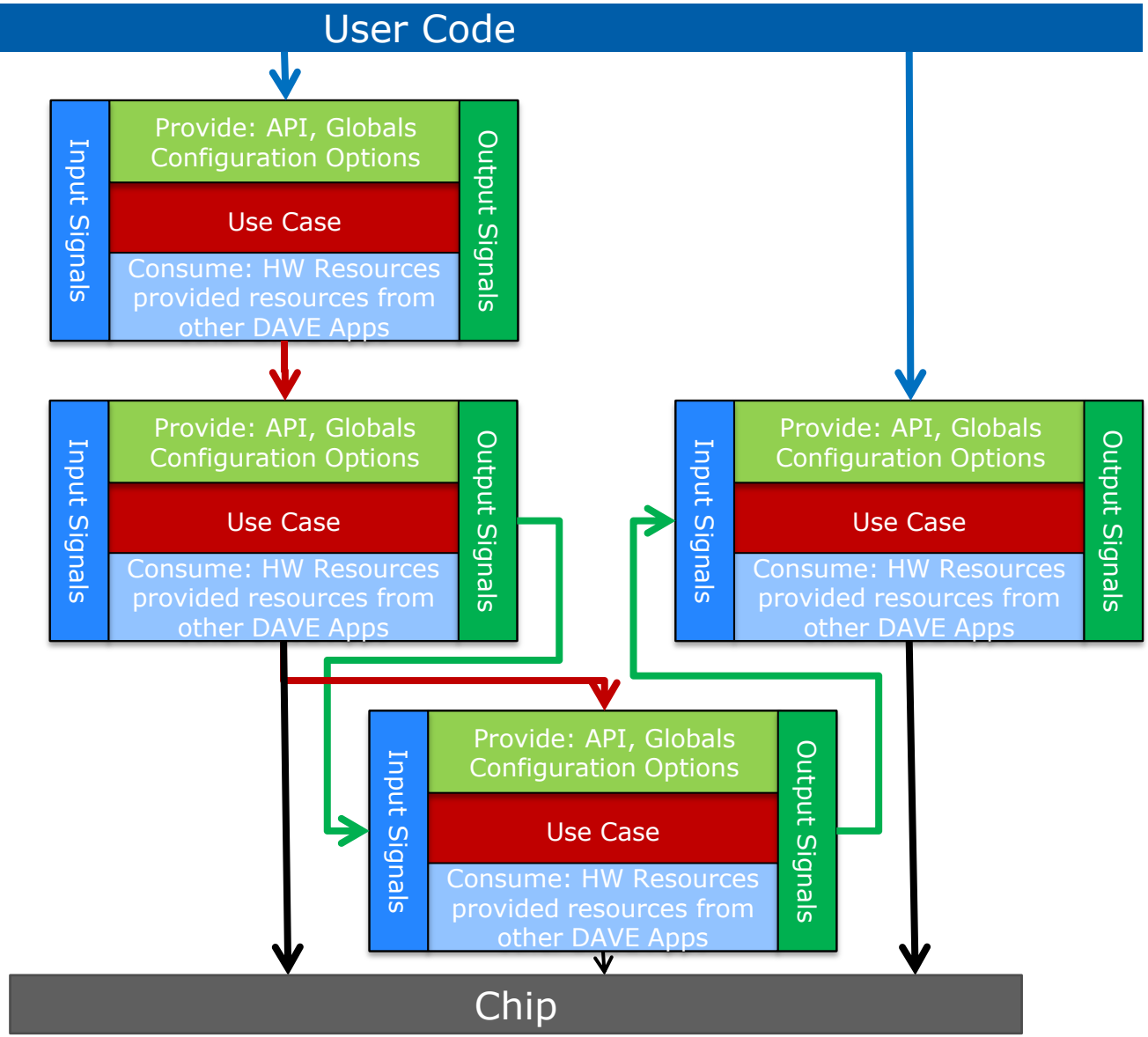
Input signal / events the triggers, enables or disables a functionality

Output Signals that can be used to trigger, or enable a functionality of another DAVE App

Resources that are required for the App Use Case, can be chip resources or an API of another DAVE App or……

DAVE™ Apps can be flexible composed to build the required Library

- User uses the composed library via APIs
- SW dependency (use of API, read/ write of configuration data)
- HW dependency (signal connection)
- Mapping to the chip resources is done automatically without conflicts

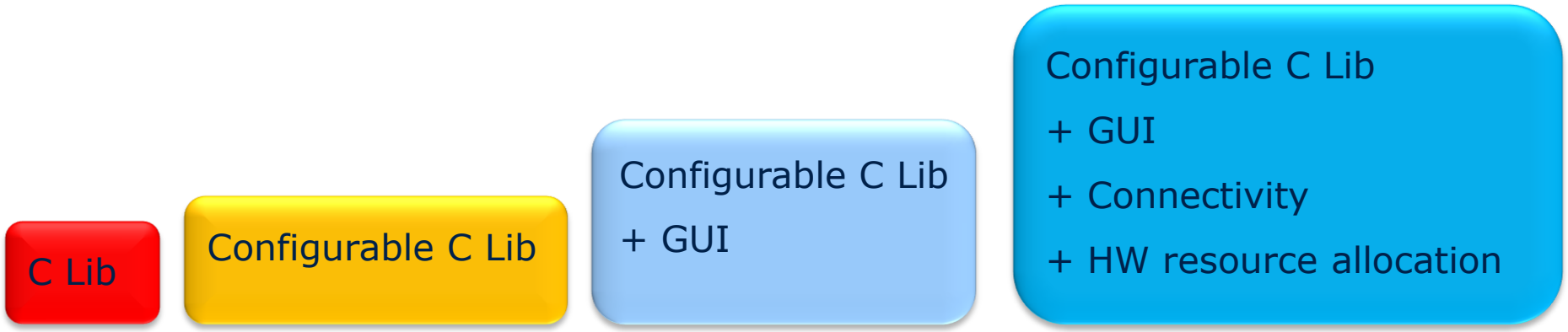


DAVE™ Apps reduces SW Development Time

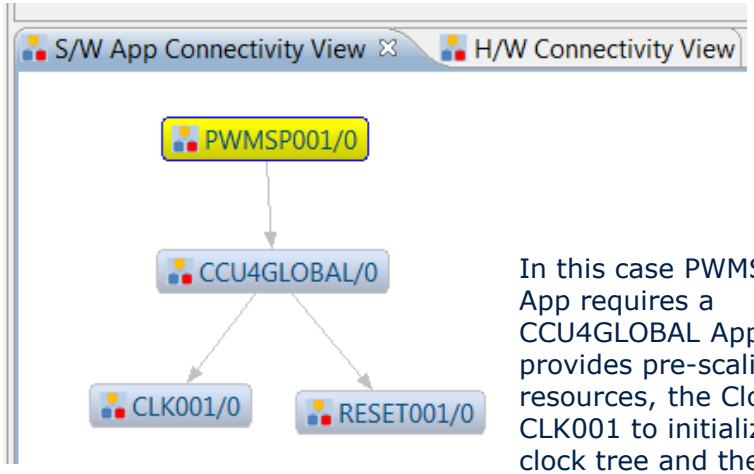
- DAVE™ Apps make flexible peripherals easy to use without compromise in the functionality
- DAVE™ Apps represent the flexible Chip HW as specific application use cases
- DAVE™ Apps abstract any kind of application use case
- DAVE™ Apps can be used as SW building block to compose the required library for any solution
- The result is a tailored library that provides all required programming interfaces (APIs)
- DAVE™ and DAVE™ Apps provide a very innovative object oriented SW development methodology

DAVE™ and DAVE™ Apps adds a big Innovation to the State of the Art Library concept

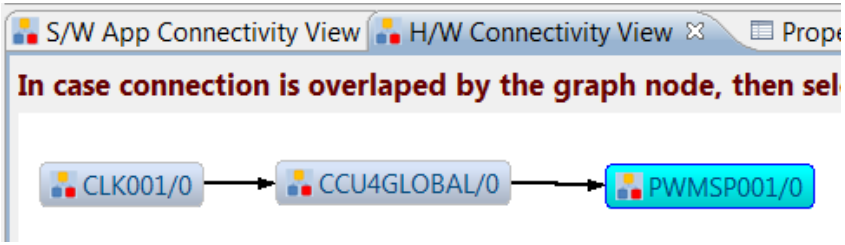
- State of the Art:
  - C Library configurable by #DEFINEs in *.h files or C structures in *.c files.
  - Configuration Header files generated from a template using parameters taken from a Graphic User Interface (GUI)
  - → Resources assignment and connectivity is not supported
- DAVE™ Innovation: DAVE™ allows composition and connectivity of Library components (DAVE Apps) and supports conflict free HW resource allocation



Example: PWMSP001 App



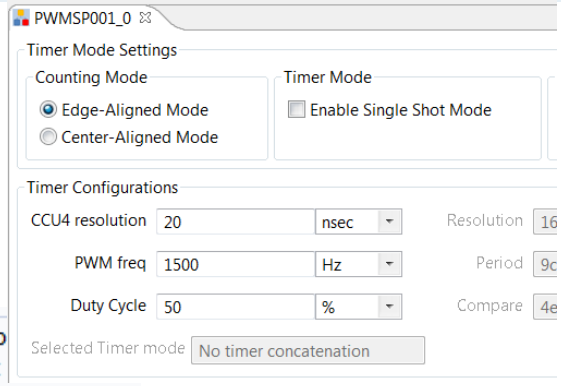
In this case PWMSP001 App requires a CCU4GLOBAL App that provides pre-scaling resources, the Clock App CLK001 to initialize the clock tree and the Reset App.



In this case it shows the connection of the clock signal from the Clock App via the CCU4GLOBAL App to the PWMSP001 App.

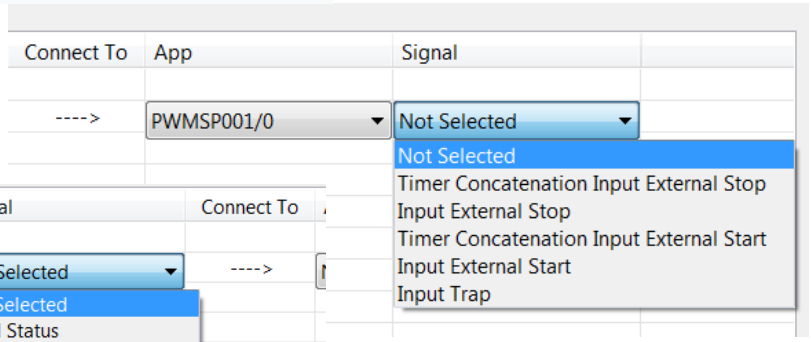
- The application use case of the PWMSP001 App is to generate a single phase PWM signal
- When adding the PWMSP001 App to the Project, all other DAVE™ Apps that are required by this PWM App are added
- The SW connectivity shows the hierarchy of required DAVE™ App
- The HW connectivity view shows the connection of signals between DAVE™ Apps

Major User Interfaces to use the PWMSP001 App

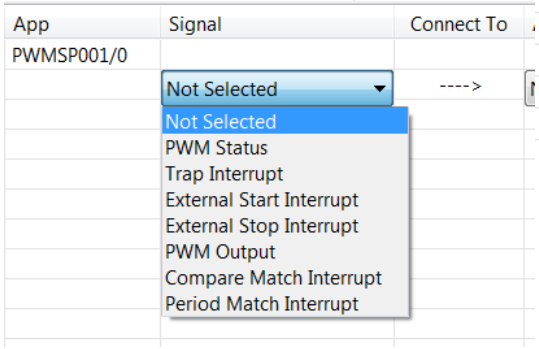


Example of APIs

```
status_t  PWMSP001_Start (co
          This function will start
          This function needs to be called
status_t  PWMSP001_Stop (const PWMSP
          This function will stop the Single P
status_t  PWMSP001_SetCompare (const
          This function will update the duty
          Duty cycle is given in terms of t
```



Input list



Output list

- In a graphical UI the DAVE™ App can be configured to determine the detailed use case and the initialization
- API of the generated code to call various functions in the user code to control the PWM signal
- The PWMSP001 App provides (logical) HW signal or events that can be connected to other DAVE™ Apps to extend the use cases in combination with other DAVE™ Apps

Mapping of the Chip Resources required by the PWMSP001 App to the available Chip Resources

- This is the task of the resource solver
- The DAVE™ App requires the "type" of resources, but not the very specific resources group (group of SFRs / SFR bit-fields)

Name	URI	Description	Conditional consumption
slice	peripheral/ccu4/*/cc4/*	CCU4 slice	Always
slice1	peripheral/ccu4/*/cc4/*	CCU4 slice	If Timer Concatenation is required
pin_directoutput	port/*/pad/*	The IO Pad resource for direct output	Conditionally depending on UI
ccu4globalapp	App/ccu4global/*	CCU4Global App -- to enable CCU clock	Always

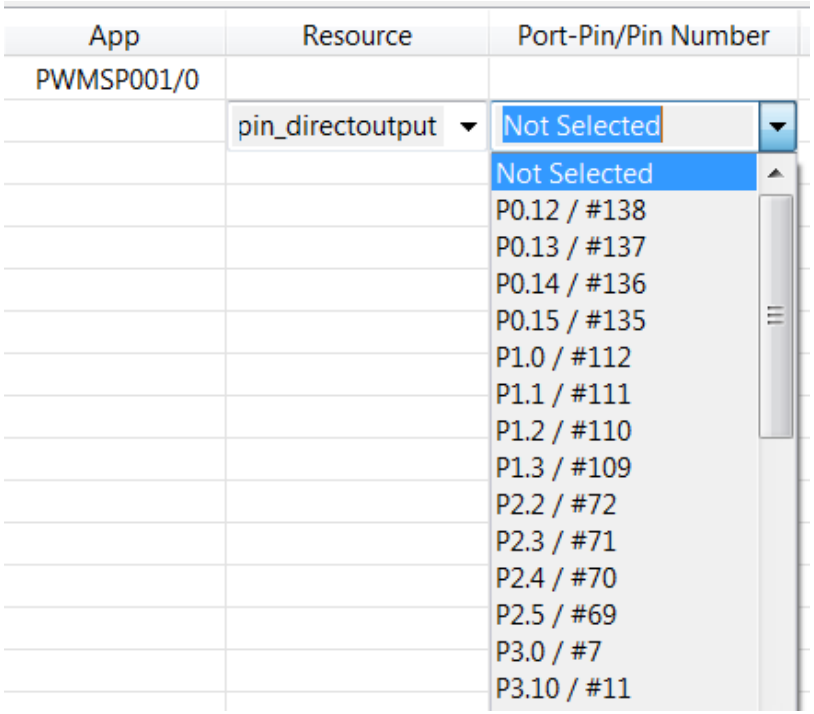
Screen shot of the documentation of the PWMSP001 App. The table shows all required resources in URI format The * represent the variable that will be resolved by the solver.

- The resource solvers takes the request of all the DAVE Apps and finds a solution without any HW conflict.

App	Resource	Mapped Resource
◢ PWMSP001/0		
	pin_directoutput	p/4/pad/3
	slice	ccu4/3/cc4/3
	slice1	

Screen shot of the report that shows the assigned resources after a solver run:
The slice to generate the PWM signal is assigned to ccu4/instance 3/cc4/slice 3.
The PWM output signal is assigned to port 4.3.
Slice1 is not required because timer concatenation is not required.

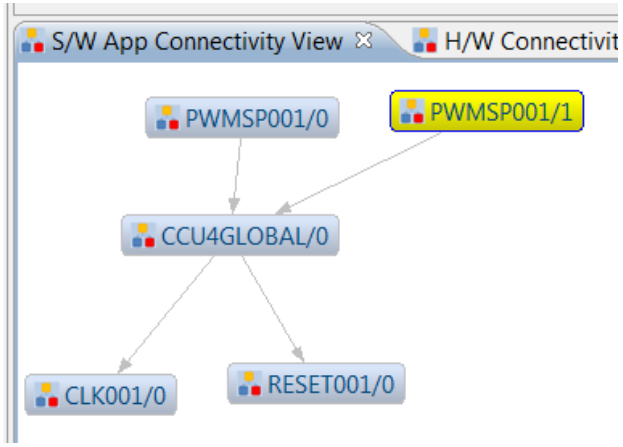
Manual Pin Assignment



Manual pin assignment view.
The output signal of PWMSP001 App (pin_directoutput) can be assigned to list of possible pin options.
All show options are viable.

- Manual pin assignment means that a specific output or input signal should be mapped to a specific pin (pad)
  - E.g. to meet the constraints of an existing PCB design
- The solver is now doing the respective mapping considering the requested pin assignment (constraint)

Instances of DAVE™ Apps



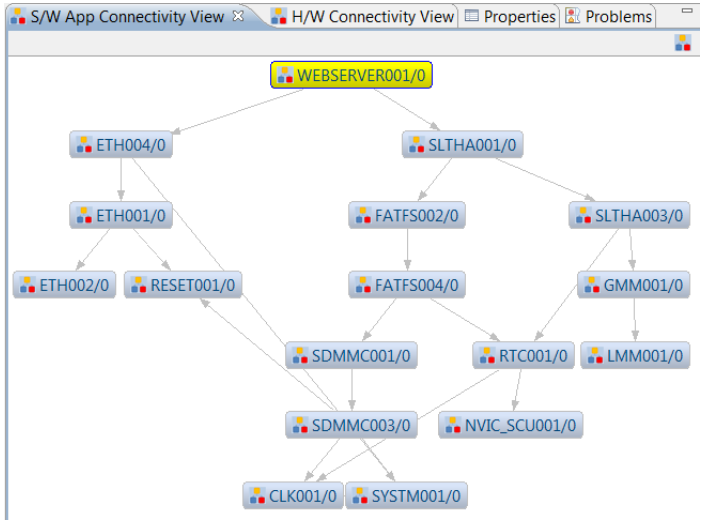
- Most of the DAVE™ Apps (like the PWMSP001 App) can be instantiated as often as it is required (limited only by the available chip resources)
- The generated library code is independent of the number of instances
- Each instance has its own data structure of configuration data

```
PWMSP001_SetDutyCycle(&PWMSP001_Handle0, 25);
PWMSP001_SetDutyCycle(&PWMSP001_Handle1, 75);
```

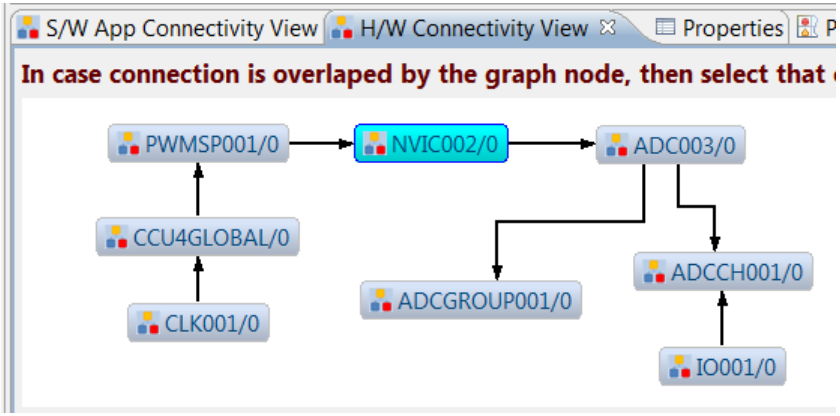
- Each instance can be selected with a pointer to the data structure (handle) in the API
- Important for migration: API is chip resource independent

- Some DAVE™ Apps can only be instantiated once like CLK001 App: Singleton

Composition of DAVE™ Apps



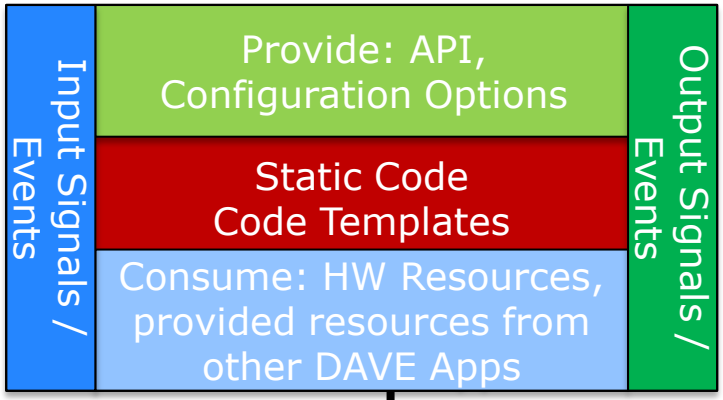
- Composition by the DAVE™ App designer
  - A DAVE™ App can require instances of other DAVE™ Apps to compose the required functionality



- Composition by the user
  - User can connect (logical) HW signals/ events between DAVE™ Apps

In this case the PWM status event is connected via an NVIC App (connection requires an interrupt node but no interrupt handler) to trigger the ADC003 App.

Architecture of DAVE™ to Generate the Library Code from DAVE™ Apps



Consume one USIC channel

Code Engine (eclipse plug in)

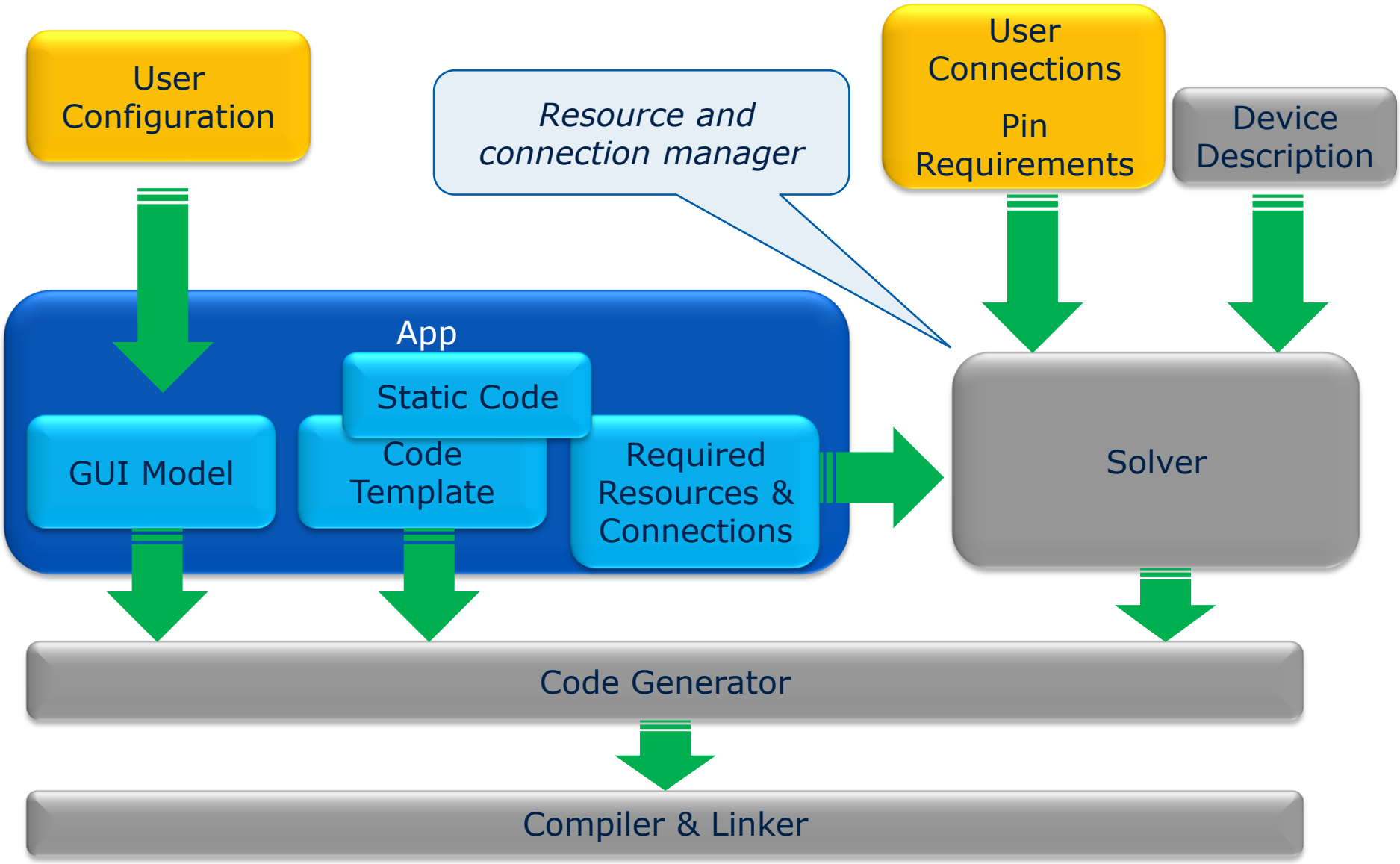
- DAVE App management
- Graphical UI
- Resource management
  - Assign USIC channel 2
- Code generation
  - Based on code templates for USIC channel 2
  - Copy of static code

Provide six USIC channels

DAVE™ Device Description
Resource and connection model of the target MCU

Generated Library

Implementation Concept to generate the Library Code



How Code Generation works in DAVE™

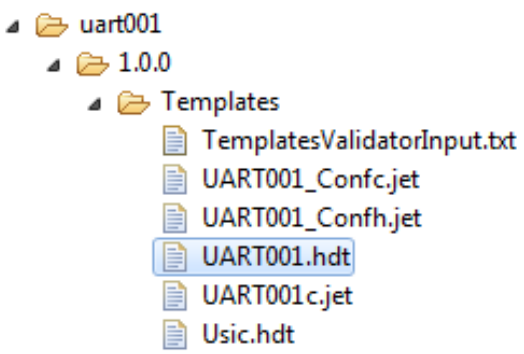
User Interface

Desired Baud Rate: 19200 baud

Solver

UART001/0		
	channel	usic/1/channel/0
	UART Receive	p/2/pad/15
	UART Transmit	p/2/pad/14

Code Template




```
 .BaudRate   = <%=
app.getIntegerValue(AppBaseuri +
appInst
+"/uart001_irwbaudrate") %>,/*Baud
Rate */
```

Code Generation

```
const UART001_HandleType UART001_Handle0  =
{
  // Temp Code for testing Eval functions
   .UartRegs = USIC1_CH0, /* Usic Channel offset value */
   .Mode = 0, /* Mode */
   .StopBit = 0,/* StopBit */
   .Parity  = 0,/* Parity */
   .DataBits = 7,/* Word Length */
   .BaudRate  = 19200,/*Baud Rate */
   .TxLimit = 1,/* FIFO Tigger Level */
   .RxLimit = 1,/* FIFO Tigger Level */
   .TxFifoSize = 1,/* Tx FIFO Size */
   .RxFifoSize = 1,/* Rx FIFO Size */
   .RecvNoiseEn = 0, /* Protocol specific interrupt enable */
   .FormatErrEn = 0, /* Protocol specific interrupt enable */
   .FrameFinEn = 0 /* Protocol specific interrupt enable */
};
```

Baudrate = 19200

How Code Generation works in DAVE™

- The Code template contains static code plus variables that represent user configurations or the mapped chip resources

Code Snippet from the Template where the code is generated from

```
/* Configuration of SDA Pin <%=portNo2%>.<%=pinNo2%> based on User configuration */

 <% if(Pin2 < 8) {%>
 PORT<%=portNo2%>->PDR0  &= (~(PORT<%=pinNo2%>_PDR0_PD<%=Pin2%>_Msk));
 PORT<%=portNo2%>->PDR0  |= ((<%=PDR_PD2%> << PORT<%=portNo2%>_PDR0_PD<%=Pin2%>_Pos) & \
                                    PORT<%=portNo2%>_PDR0_PD<%=Pin2%>_Msk);

 <% } else {%>
 PORT<%=portNo2%>->PDR1 &= (~(PORT<%=portNo2%>_PDR1_PD<%=Pin2%>_Msk));
 PORT<%=portNo2%>->PDR1 |= ((<%=PDR_PD2%> << PORT<%=portNo2%>_PDR1_PD<%=Pin2%>_Pos) & \
                                PORT<%=portNo2%>_PDR1_PD<%=Pin2%>_Msk);
```

Respective source code that is generated from the Template

```
   /* Configuration of SDA Pin 2.14 based on User configuration */
 PORT2->PDR1 &= (~(PORT2_PDR1_PD14_Msk));
 PORT2->PDR1 |= ((0 << PORT2_PDR1_PD14_Pos) & \
                                PORT2_PDR1_PD14_Msk);
```

Code snippet example to create the initialization code for the Port PDR register .

Here:
portNo = 2
pinNo= 14
Set in the data model by solver or user via UI

- The generated code is easy readable and full deterministic

DAVE™ App Summary

- A DAVE™ App is an application use case oriented SW component
- DAVE™ Apps make complex peripherals easy to use without compromise in the functionality
  - DAVE™ Apps represent the flexible chip HW as specific application use cases
  - DAVE™ Apps can be composed to create a library for complex requirements
- A resource solver ensure conflict free assignment of resources
- Major User actions
  - Configuration via the graphical UI
  - Connect HW signals between DAVE Apps
  - Manual Pin assignment as required
  - Add API to the source code

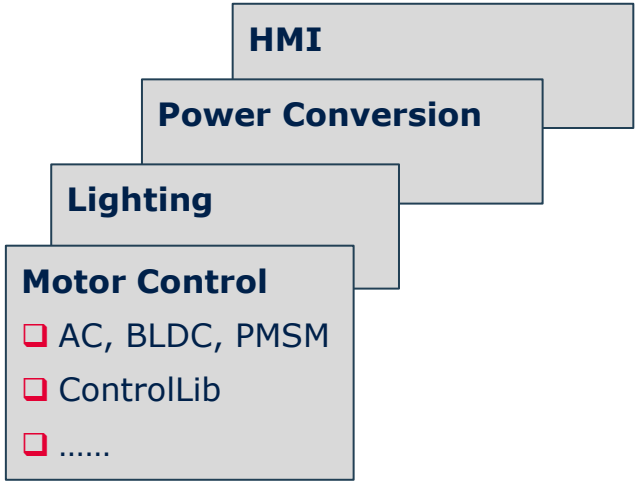
Overview of available and planned DAVE™ Apps for the XMC Families

Service Apps
- Clock
- Reset
- SystemControl
- Power Mgmt.
- Watchdog
- DMA
- NVIC /Exception
- Request Unit
- I/O
- EBU
- Flash
- CRC
- AES
- SW/SysTimer
- Debug Log
- CMSIS RTOS
- Libraries

Middleware
- USB stack, class drivers, host and device
- TCP/IP stack plus HTTP, FTP, SNMP
- SD/MMC, file system,…
- GUI lib plus driver for intel. LCD
- …..

Basic Applications
- PWM, Capture, Timer, Counter
- Communication
- Analog
- ……..

Application Specific Middleware

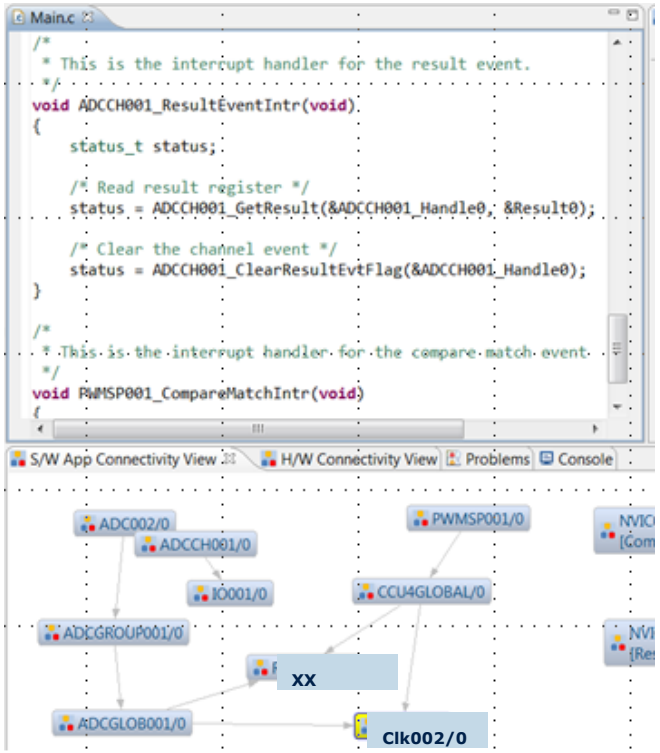


Ref. Projects
- elua
- HMI system
- ……

In total Infineon is working on > 200 DAVE Apps
Details about the latest released DAVE Apps can be found here:
http://www.infineon.com/cms/en/product/promopages/aim-mc/DAVE_3_Support_Portal/Release_Note_update.html

Most of the DAVE™ Apps will support the entire XMC family: XMC1000 and XMC4000

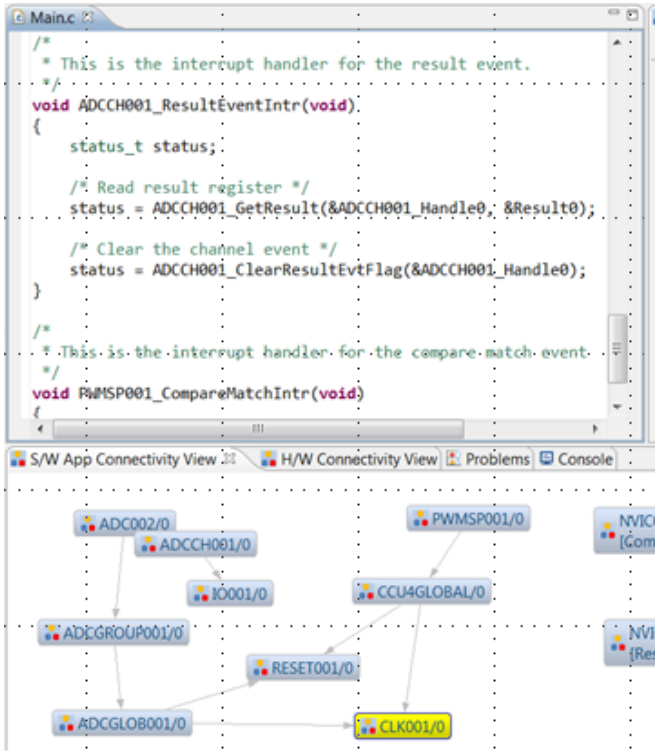
- Around 75% of the DAVE Apps can be used on both families
  - Respective user SW can be reused for both families



XMC1000 Project

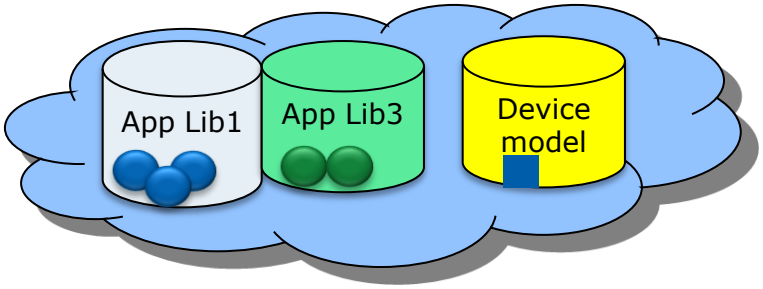
Migration

Automated migration support is planned

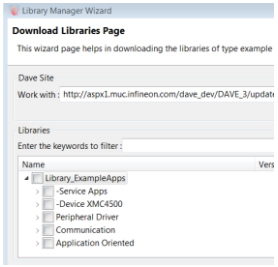


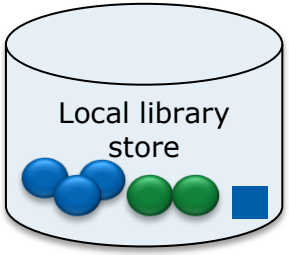
XMC4000 Project

Installation and Selection of DAVE™ Apps

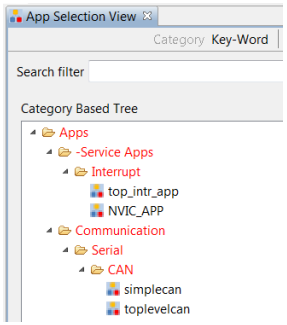


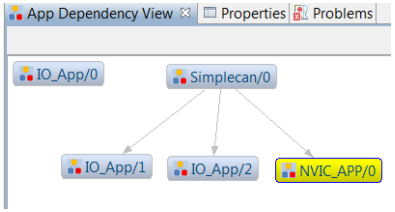
http://dave.infineon.com/Libraries/DAVEApps/XMC4500/v3.1/





C:\Users\yourname\Infineon\LibraryStore



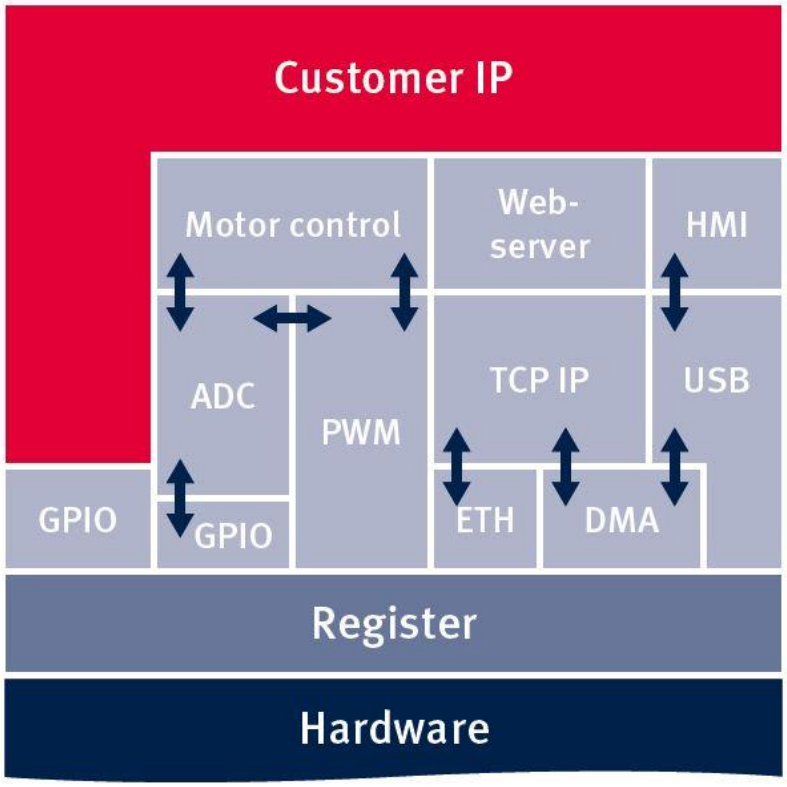


- DAVE Apps are provided as libraries of DAVE™ Apps (UpdateSite) from the web or any other media
- The DAVE™ Apps can be downloaded from the web (or UpdateSite zip file) with DAVE™ to a local library store
- The user can select the DAVE™ Apps from his local library store and add them to his project
- Searching and selection is support by categories and key word filters
- For each DAVE™ App there is a dedicated documentation and example projects

Interfaces to 3rd Party Tools / SW or existing Legacy SW

- Using DAVE™ generated code with commercial tool chains
  - Keil, IAR and Altium provide a dedicated import functionality
  - For Atollic and Rowley the generated code can manually imported
  - In future some of them may include the DAVE CE plug in into their eclipse based tool chain
- Using DAVE™ generated code with 3rd party SW and middle ware
  - To avoid HW resource conflicts a resource reservation functionality will be provided in future
  - We consider to provide a DAVE™ App frame for 3rd party components
- Existing legacy code
  - If this code requires HW resources, these recourses should be excluded form solving with resource reservation functionality that will be provided in future

The SW Developers can focus to the Important IP and leave the Rest to DAVE™



Composing the HW related drivers and middleware with DAVE™ Apps

DAVE™ ensures that the DAVE™ Apps are mapped to the available Chip resources without conflicts:

Considering:

- **Signal Interconnections**
- **I/O PIN constraints**

- **DAVE™ makes SW development faster**
- **Free download: www.infineon.com/dave**

ENERGY EFFICIENCY
MOBILITY
SECURITY

Innovative semiconductor solutions for energy efficiency, mobility and security.