

# DAVE™ version 4 – Quick Start Simple LED Blinky via a Generated PWM Signal

XMC Microcontrollers  
June 2018



# Learning Outcome

- › Learn the basic principles of DAVE™ version 4:
  - Installation
  - Required XMC kit
  - Create DAVE™ Project
  - GUI based DAVE™ APPs configuration
  - Graphical pin mapping
  - One touch code generation
  - Download and debug code
  - DAVE™ updates
  - Expert support

# DAVE™ installation and update

## › Easy installation



1. Go to [www.infineon.com/DAVE](http://www.infineon.com/DAVE) and download DAVE™ version 4
2. The downloaded zip file contains all required installation instructions, please follow the instructions described in section 1
3. Then follow the update instructions described in section 2
4. After installation, DAVE™ v4 can be started from the desktop

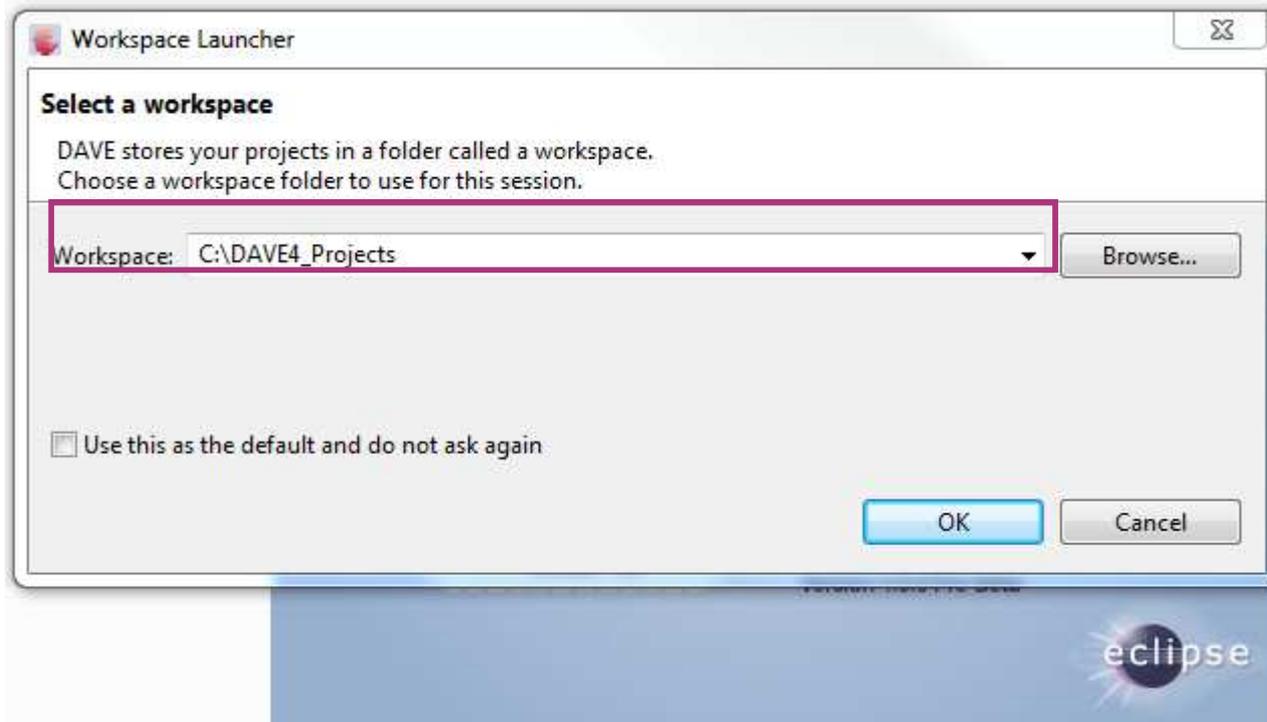
# Required XMC kit



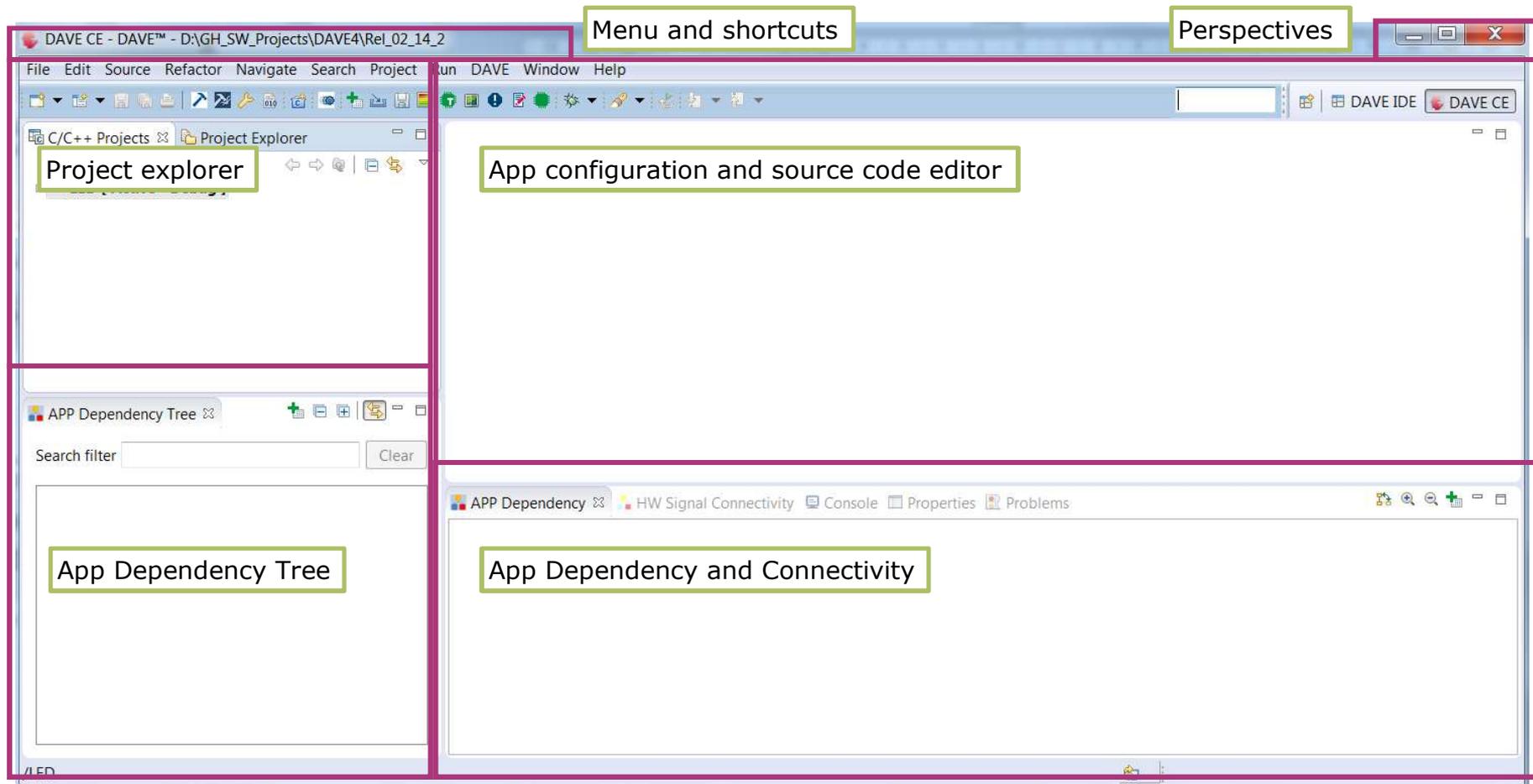
- In this tutorial we use the XMC 2Go kit equipped with an XMC1100-Q024F0064 and a Segger J-Link on-board debugger (OBD)
  
- Also any other XMC1000 or XMC4000 kit can be used. In this case the manual pin assignment described on page 17 and 18 needs to be adjusted and the pin that is connected to a LED of the respective board has to be assigned

# Starting DAVE™ for the first time

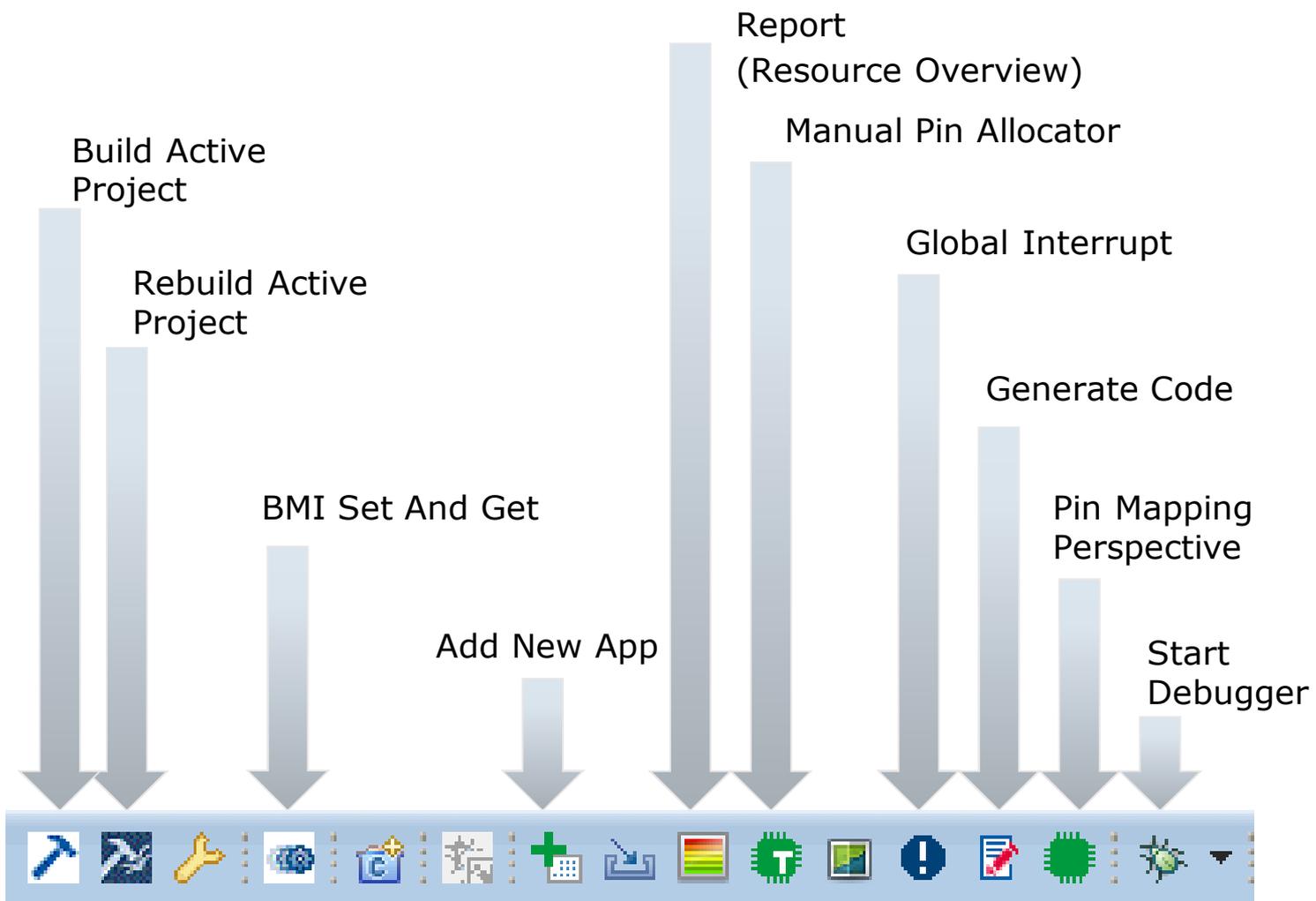
- › Start DAVE
- › Enter path to workspace folder
  - Please choose a new workspace folder, not an existing workspace folder from an earlier DAVE™ version



# DAVE™ CE Workspace



# Tool Panel

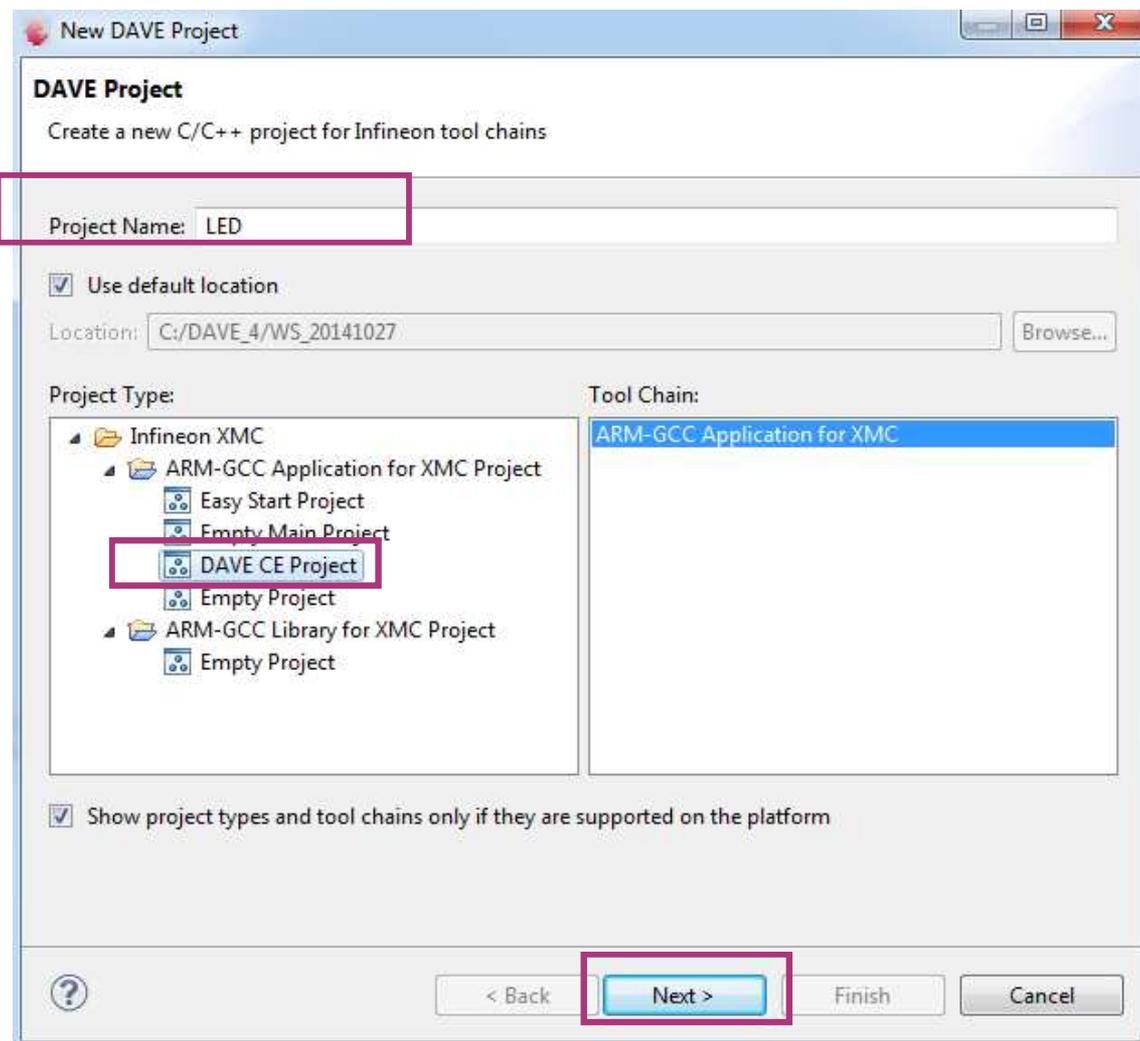


# Create a New DAVE™ CE Project (1/2)

## › Create DAVE™ Code Engine (CE) Project

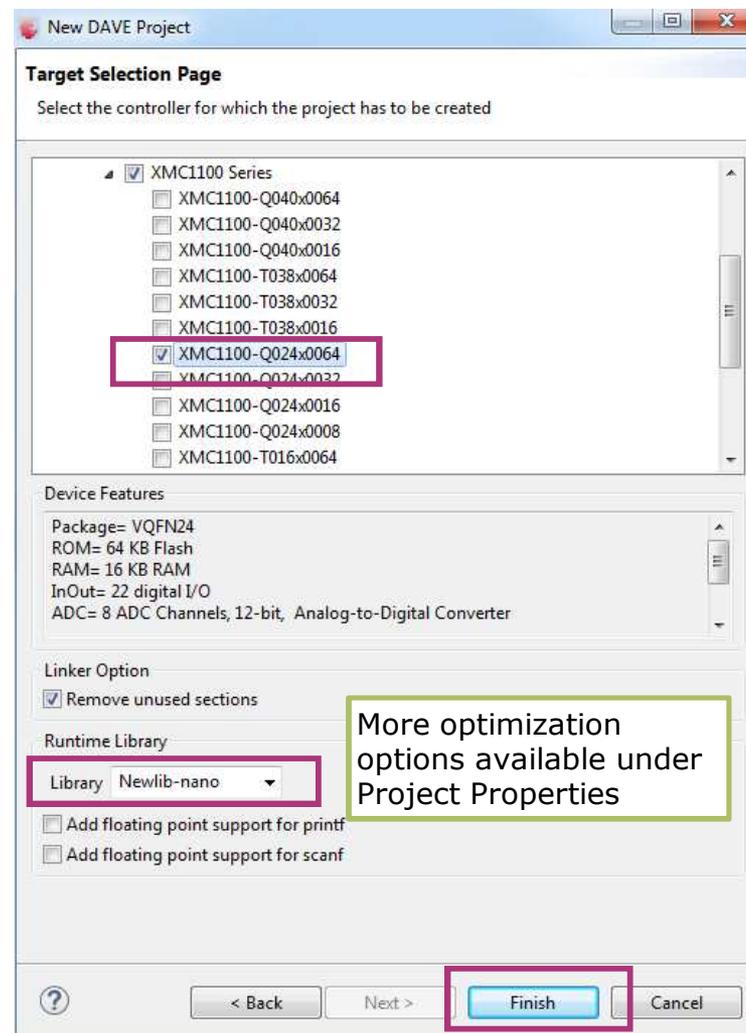
1. Go to File → New → DAVE Project
2. Select DAVE CE Project
3. Click Next

Note, a DAVE CE project is required to use DAVE APPs.

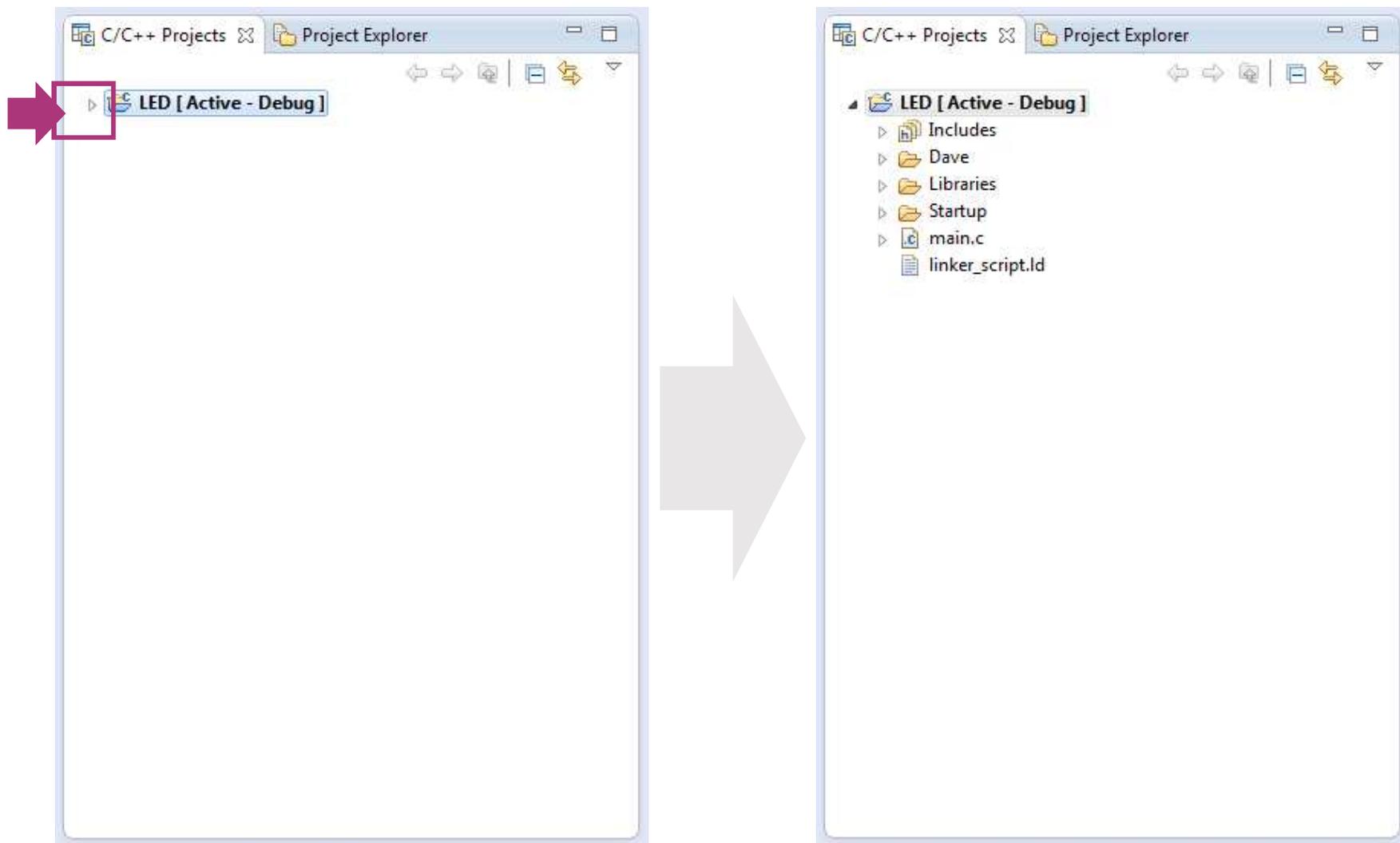


# Create a New DAVE™ CE Project (2/2)

- › Select the appropriate microcontroller
- › For XMC 2Go kit used in this tutorial
  - XMC1100-Q024F0064
- › For a different XMC kit please select the XMC microcontroller that is populated on your board



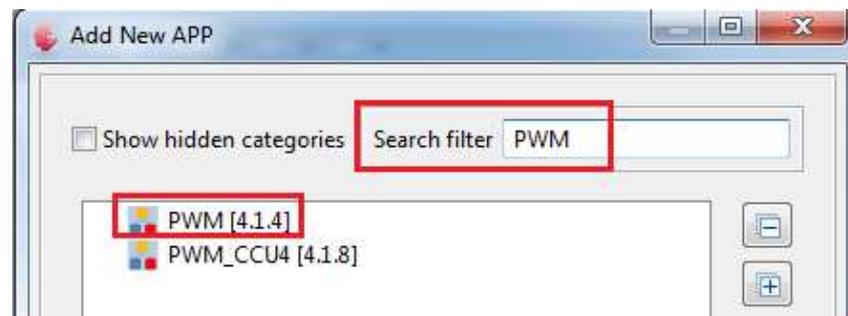
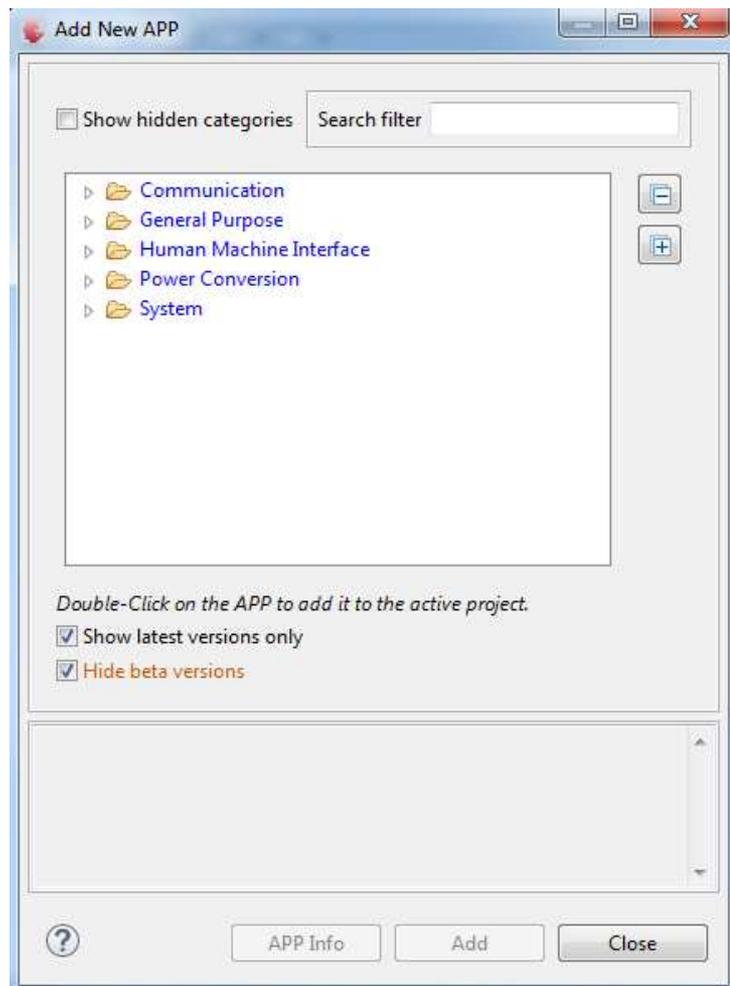
# Project View



# Add DAVE™ APP from the Local Library Store

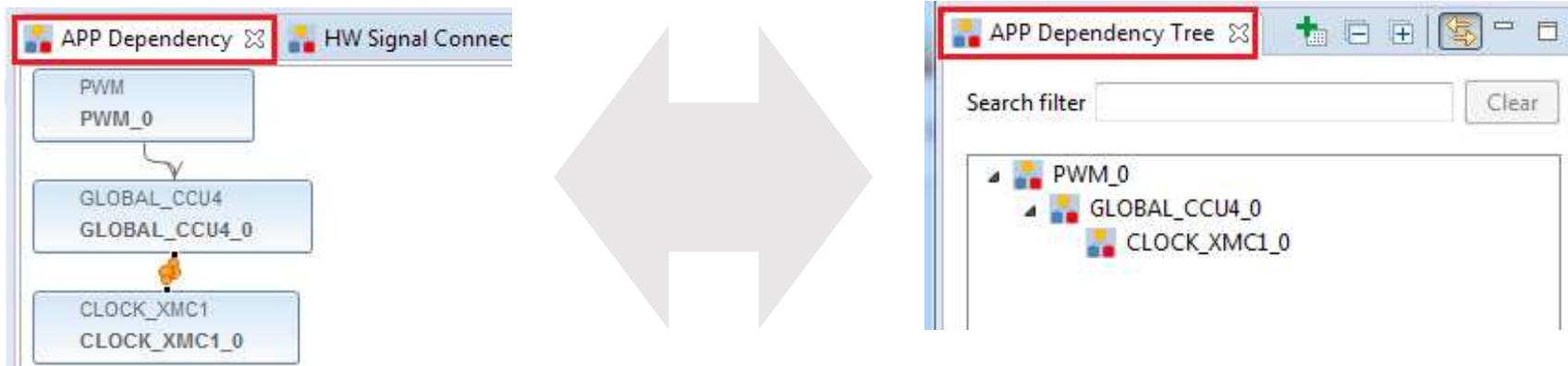
› Add DAVE™ APP to project

1. Click  in Tool Panel, or
2. DAVE → APP New APP
3. Type "PWM" in the search filter field, and double-click PWM APP

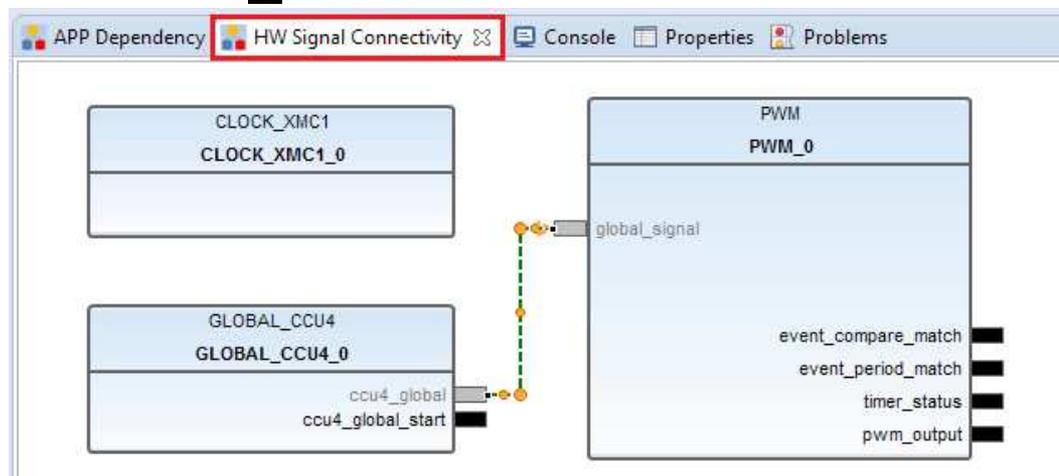


# More Project Views

All APPs included in the Project are displayed in different views:



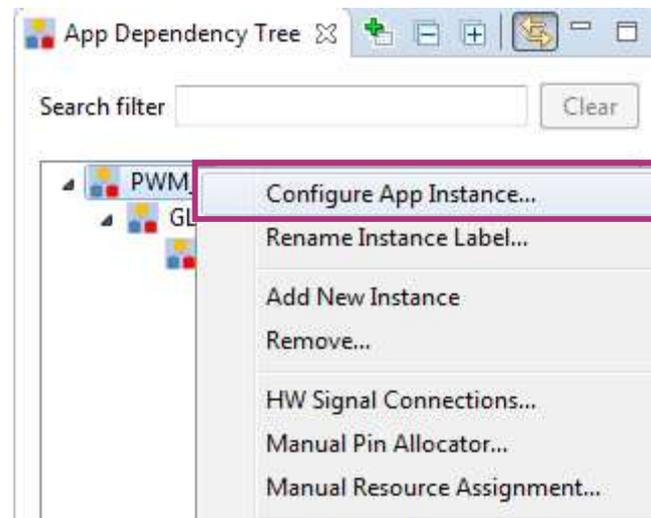
The number behind “\_” identifies the instance of an APP



# DAVE™ APPs Configuration View

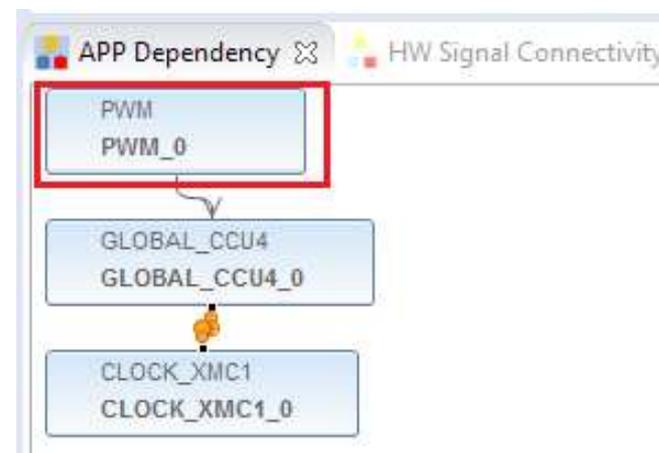
› Either

1. Right-click APP name in the App Dependency Tree view
2. Select Configure App Instance



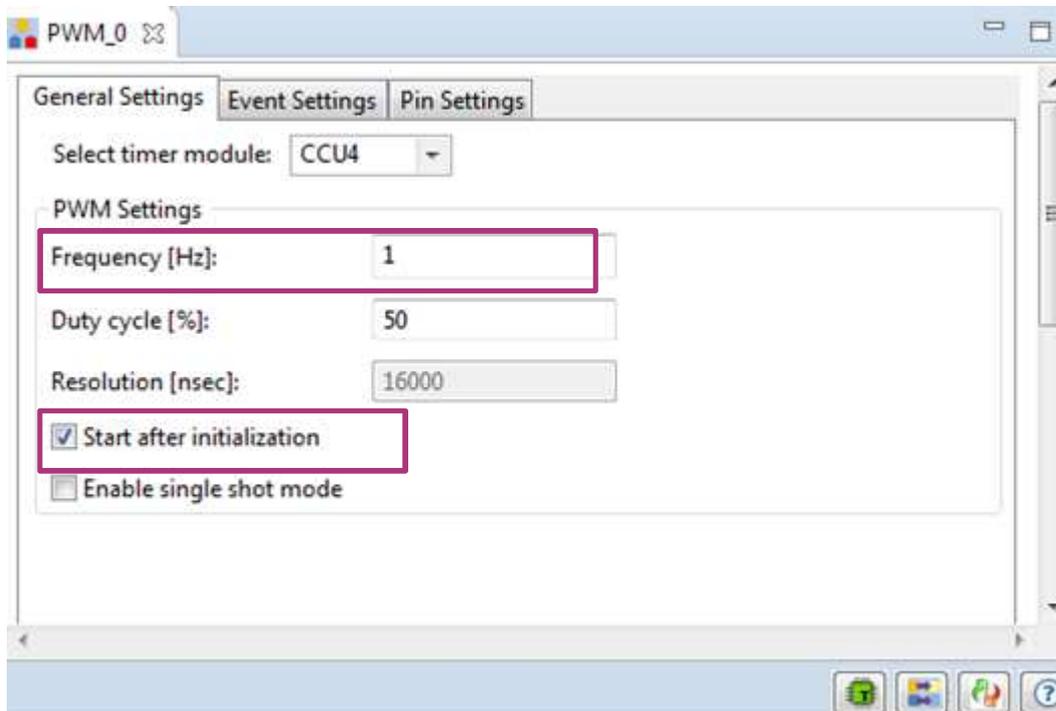
› Or

- Double-click APP name in the App Dependency view



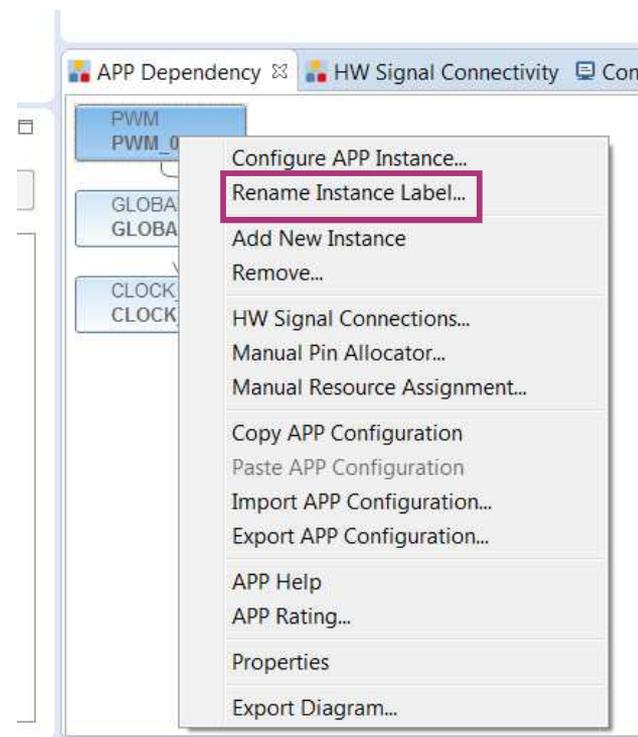
# PWM APP Configuration

- › Configure PWM APP via graphical user interface editor
  1. Set PWM Frequency to 1 Hz
  2. LED blinks every 0.5 second
  3. Check “Start Timer After Initialization”



# Rename the Instance Label of the PWM APP

- › Right click on the PWM APP
- › Select  
Rename Instance Label...
- › Type in: MyLED
- › Now “MyLED” can be used as handler in the APIs of the PWM APP to reference this instance

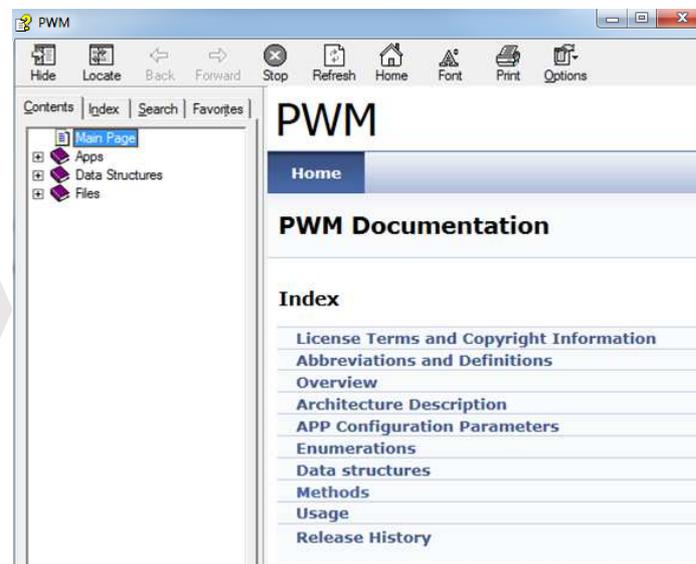
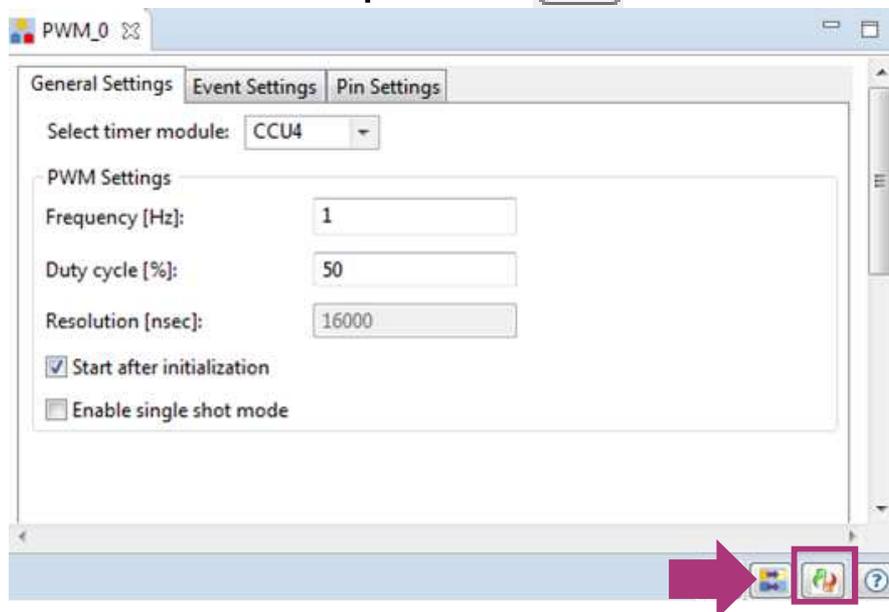


# Hint: Additional Information about APPs

› Reference to DAVE™ APP information

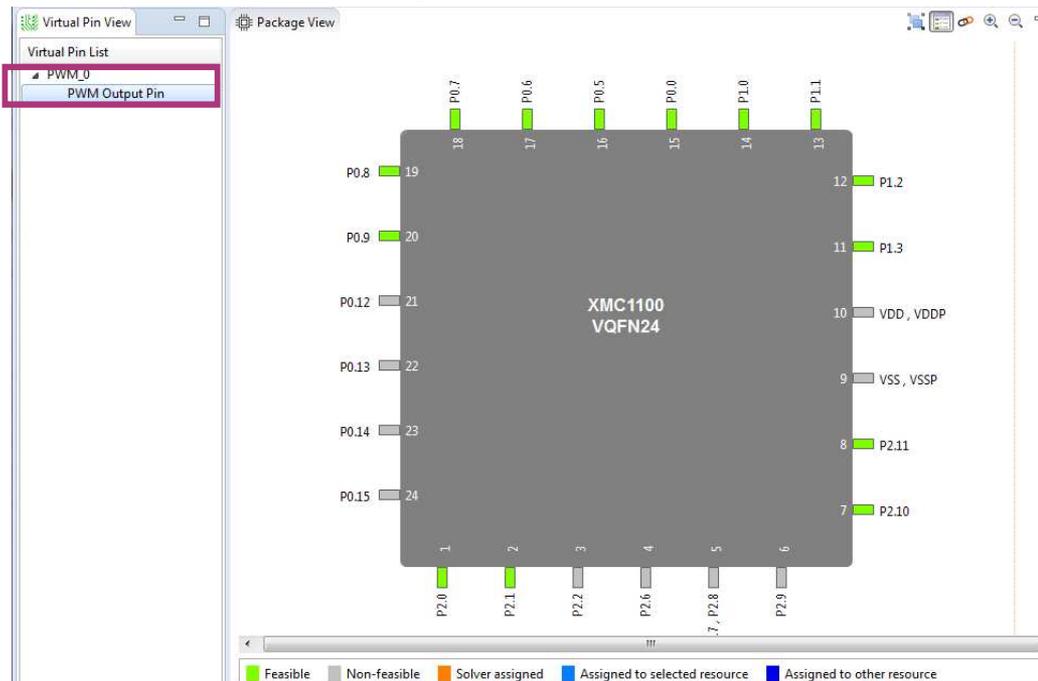
1. Double-click DAVE™ APP (e.g. PWM\_0) in App Dependency View

2. Click Help icon 



# Pin Mapping for PWM App (1/2)

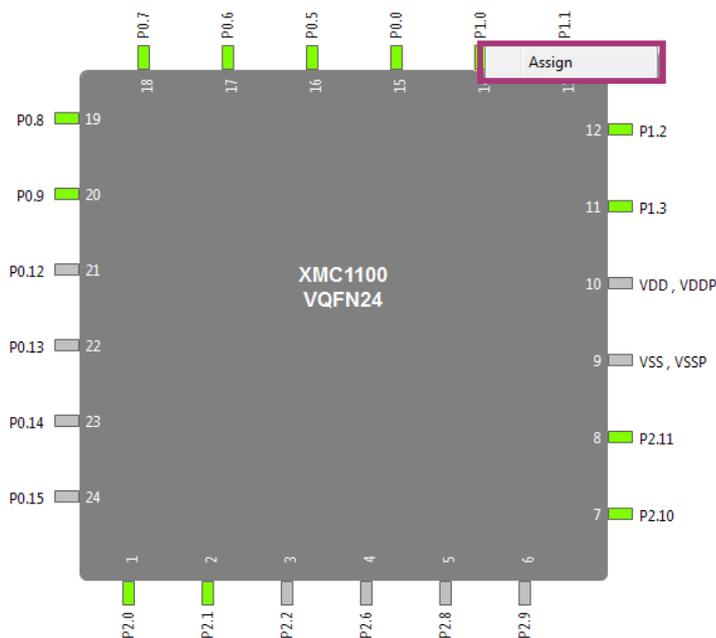
- › Assign signal to pin with graphical pin mapping view
  1. Click  to open Pin Mapping Perspective
  2. Under Virtual Pin List, select PWM Output Pin
    - Green pin: All possible pins for selected signal
    - Blue pin: User assigned pin



# Pin Mapping for PWM App (2/2)

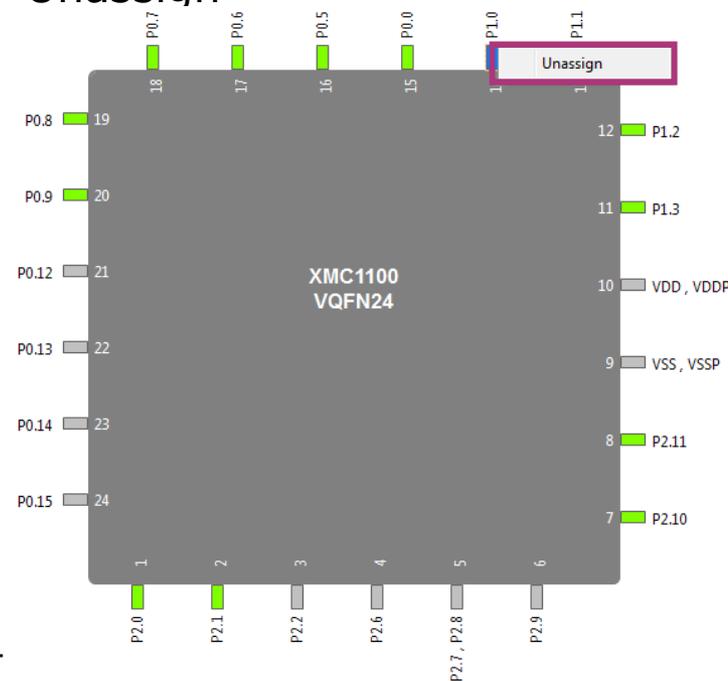
› To assign pin:

- Right-click on a **green** pin → Assign



› To unassign pin:

- Right-click on a **blue** pin → Unassign

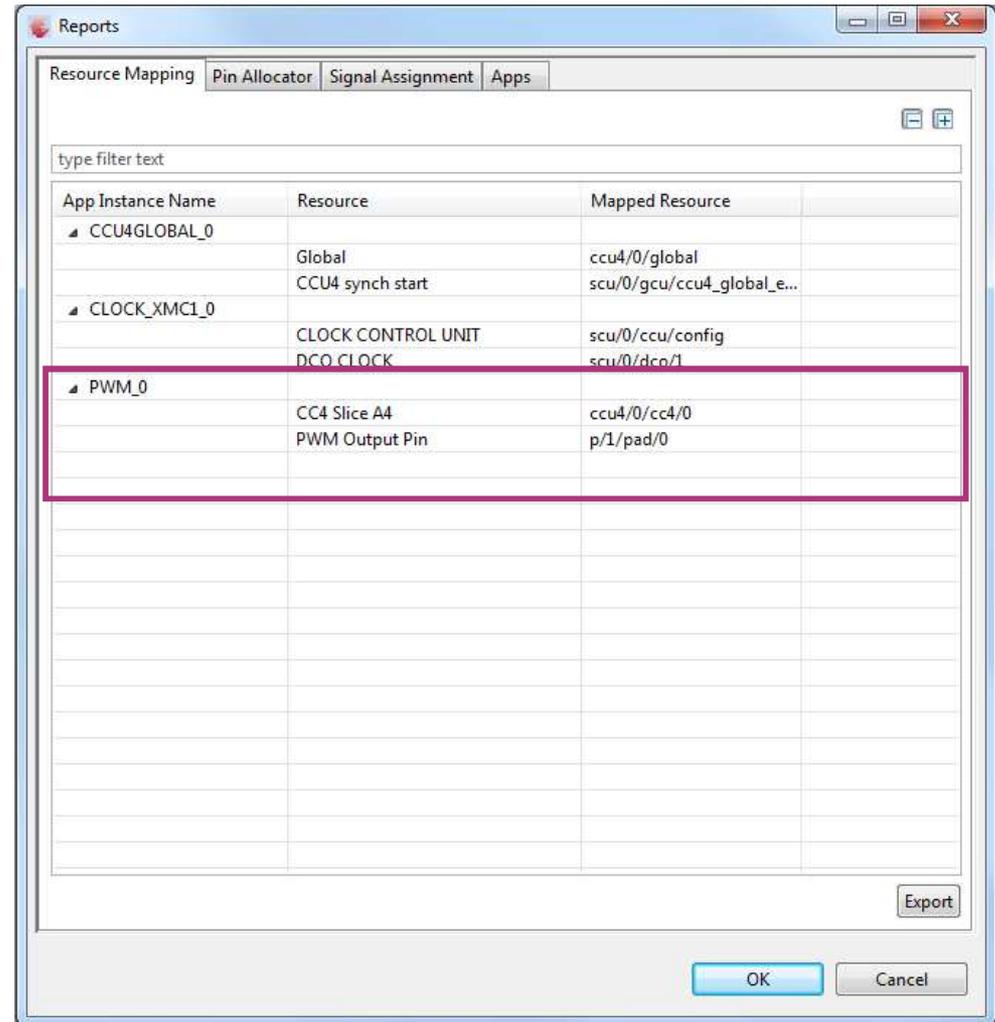


- Assign PWM Output Pin to User LED1 at P1.0/#14

- Right-click on pin 14 → Assign
- In case you use a different board / device please select a pin that is connected to a LED

# Hint: Check Correct Resource Mapping

- › Check resource mapping
  - Click  to open Reports in DAVE CE perspective



# Generate Code and add change the Duty Cycle of the PWM and compile code

## › One touch code generation

1. Click  in the tool panel

2. Generated code can be found under C/C++ Projects window, DAVE → Generated

## › Open main.c and Add the following lines

```
PWM_SetDutyCycle(&MyLED, 1000); // set duty cycle to 10%
```

```
PWM_SetDutyCycle(&MyLED, 9000); // set duty cycle to 90%
```

**Note:** you may use the eclipse code completion features (SRTG SPACE) to support correct coding.

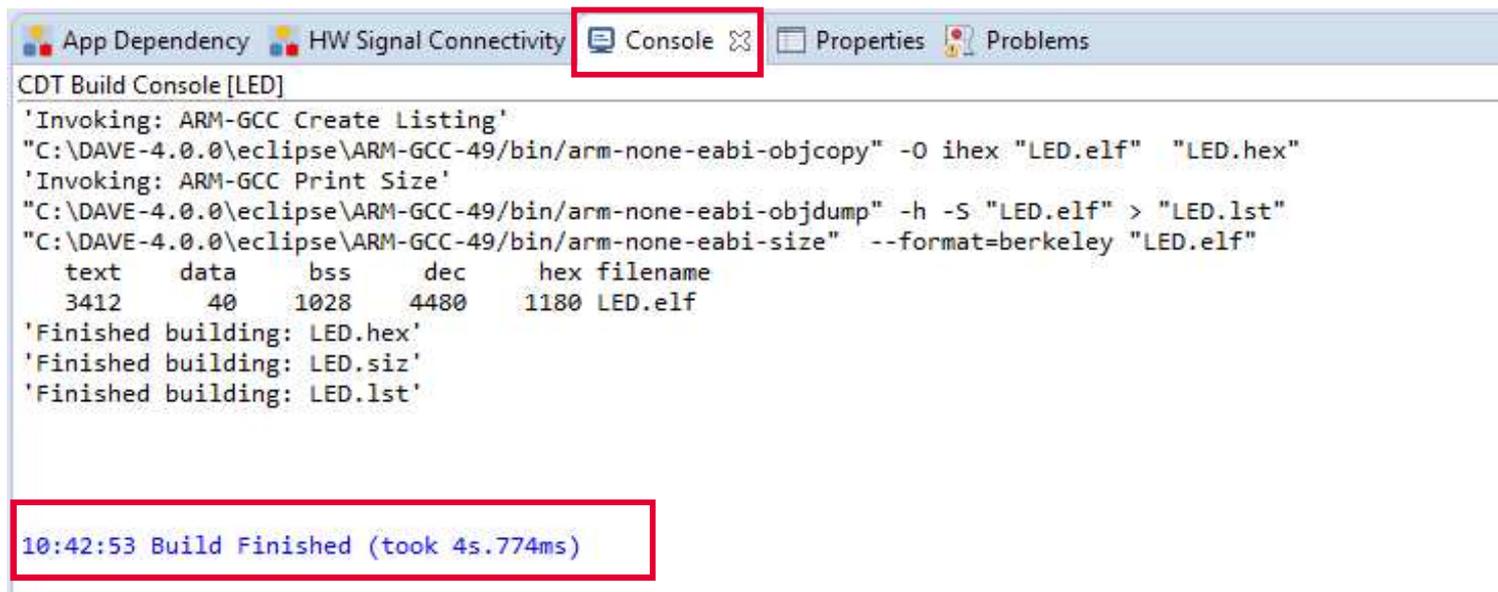
## › Start Compiler tools to build the project

– Click  in the tool panel

```
28 if(status == DAVE_STATUS_FAILURE)
29 {
30     /* Placeholder for error handler code. The while loop below
31     XMC_DEBUG(("DAVE Apps initialization failed with status %d\n
32     while(1U)
33     {
34     }
35 }
36
37 /* Placeholder for user application code. The while loop below
38
39 PWM_SetDutyCycle(&MyLED, 1000); // set duty cycle to 10%
40 PWM_SetDutyCycle(&MyLED, 9000); // set duty cycle to 90%
41
42 while(1U)
43 {
44 }
45 }
```

# Check Compiler Results

- › Ensure that Compiler finished building in Console window



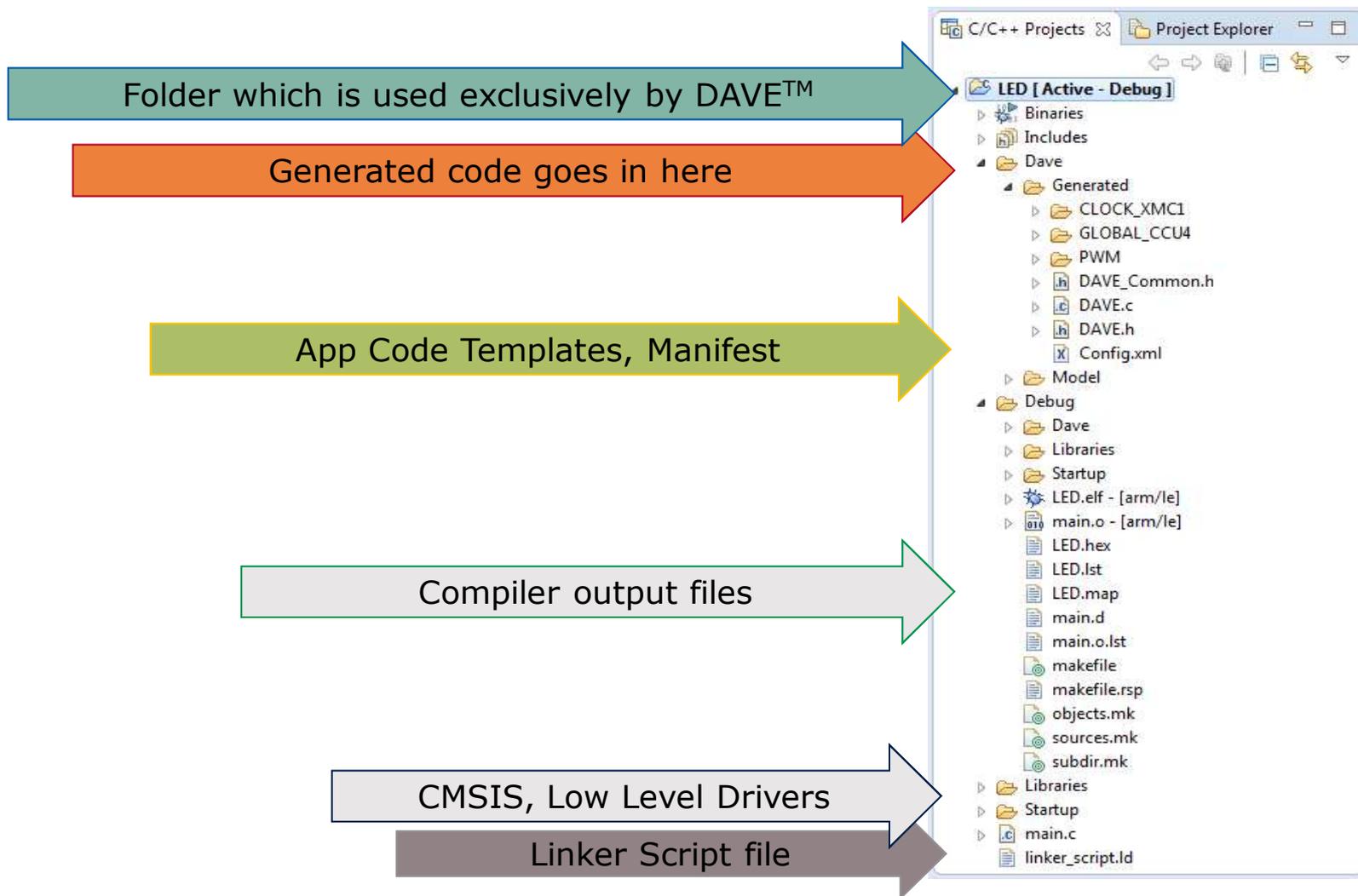
```

CDT Build Console [LED]
'Invoking: ARM-GCC Create Listing'
"C:\DAVE-4.0.0\eclipse\ARM-GCC-49/bin/arm-none-eabi-objcopy" -O ihex "LED.elf" "LED.hex"
'Invoking: ARM-GCC Print Size'
"C:\DAVE-4.0.0\eclipse\ARM-GCC-49/bin/arm-none-eabi-objdump" -h -S "LED.elf" > "LED.lst"
"C:\DAVE-4.0.0\eclipse\ARM-GCC-49/bin/arm-none-eabi-size" --format=berkeley "LED.elf"
  text  data  bss   dec   hex filename
 3412   40  1028  4480  1180 LED.elf
'Finished building: LED.hex'
'Finished building: LED.siz'
'Finished building: LED.lst'

10:42:53 Build Finished (took 4s.774ms)

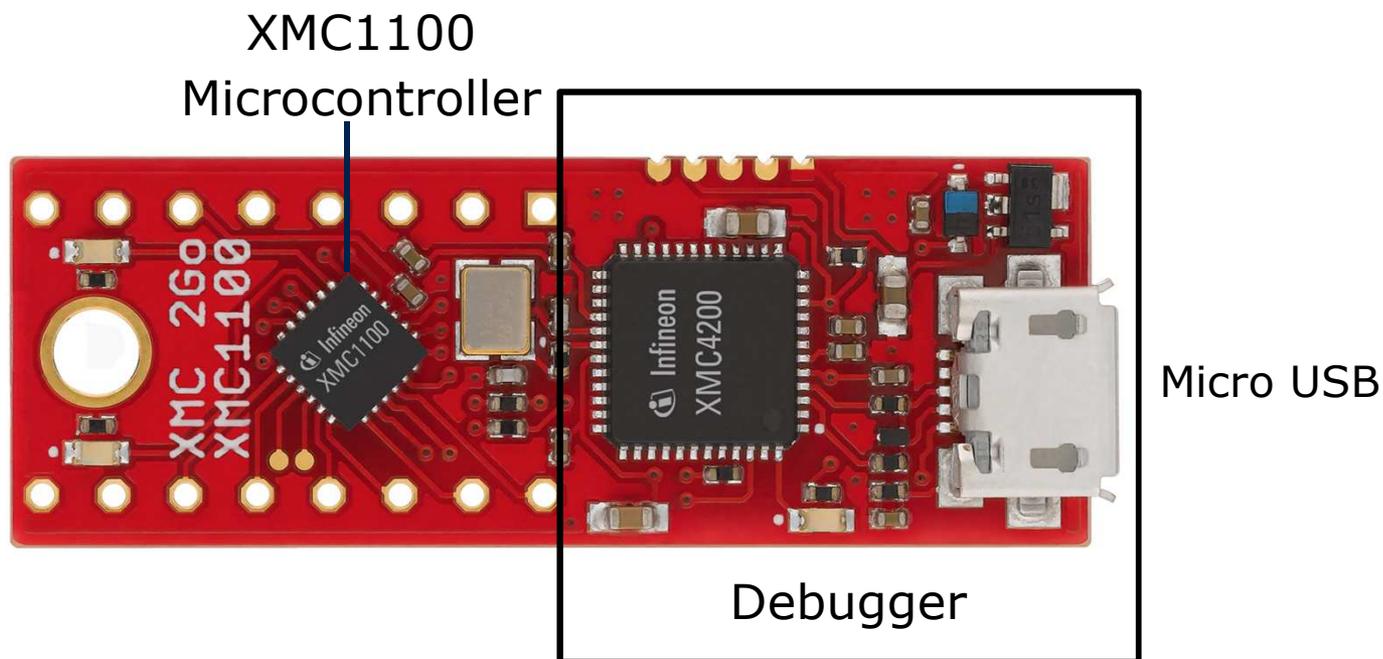
```

# The Project Folder



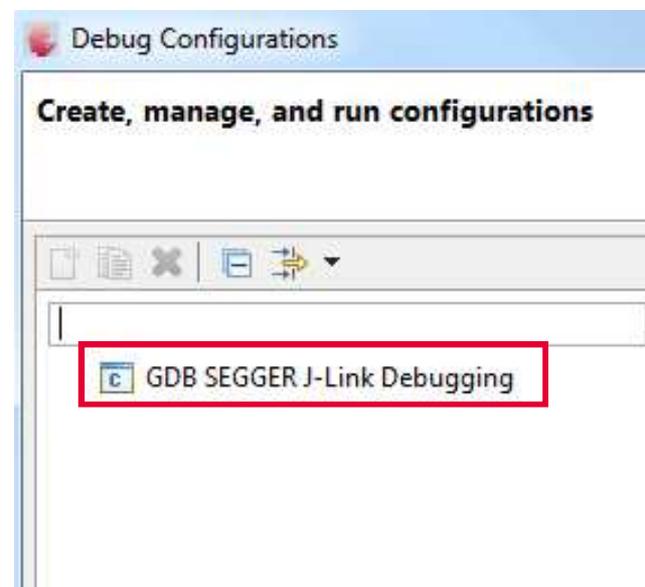
# Flash and Debug (1/3)

- › Ensure the Debugger of the XMC 2Go Kit is connected to your PC via USB



# Flash and Debug (2/3)

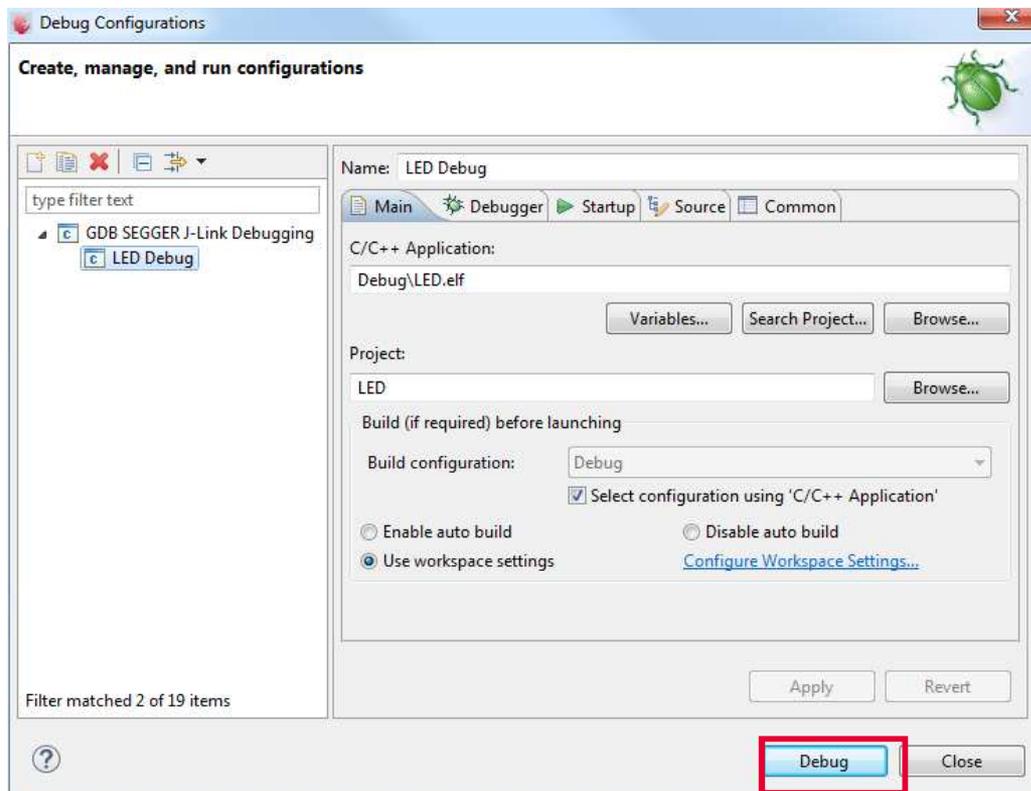
- › Start Debug Session
  - Click  in the tool panel
  
- › Create a new Debug Configuration
  - Double-click "GDB SEGGER J-Link Debugging"



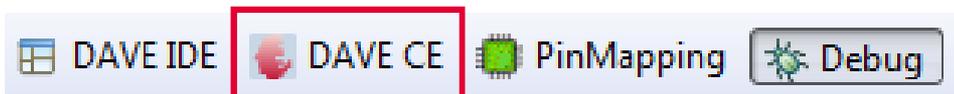
Segger J-link Driver software 4.96h or above needs to be installed

# Flash and Debug (3/3)

- › Click “Debug”
- › The flashing process is started and DAVE automatically switches to Debug Perspective



- › Hint: To switch to Project Workspace Perspective, click DAVE CE at upper right corner of window



# The Debug Perspective (1/6)

## Debug Workspace

The screenshot displays the IDE's Debug Perspective with several key components and annotations:

- Debugger actions:** A red box highlights the toolbar icons for running, stepping, and other debugging actions.
- Inspection controls:** A green box highlights the 'Variables' tab in the 'Inspect Data' window, which shows a table of variables.
- Breakpoint automatically set in main():** A green box highlights the source code editor where a breakpoint is set on the line `status = DAVE_Init();`.
- Device memory:** A red box highlights the 'Memory' tab in the 'Inspect Data' window, which shows the disassembly of the code at the current instruction address.

Name	Type	Value
(0)- status	DAVE_STATUS_t	DAVE_STATUS_SUCCESS
▶ MyLED	PWM_t*	0x0

```
22 int main(void)
23 {
24     DAVE_STATUS_t status;
25     PWM_t* MyLED;
26
27     status = DAVE_Init(); /* Initialization of DAVE Apps */
28
29     if(status == DAVE_STATUS_SUCCESS)
30     {
31         XMC_DEBUG("DAVE Apps initialization success\n");
32     }
33     else
34     {
35         /* Placeholder for error handler code. The while loop below can be replaced with a user error handler */
36         XMC_DEBUG(("DAVE Apps initialization failed with status %d\n", status));
37         while(1)
38         {
39             // ...
40         }
41     }
42 }
```

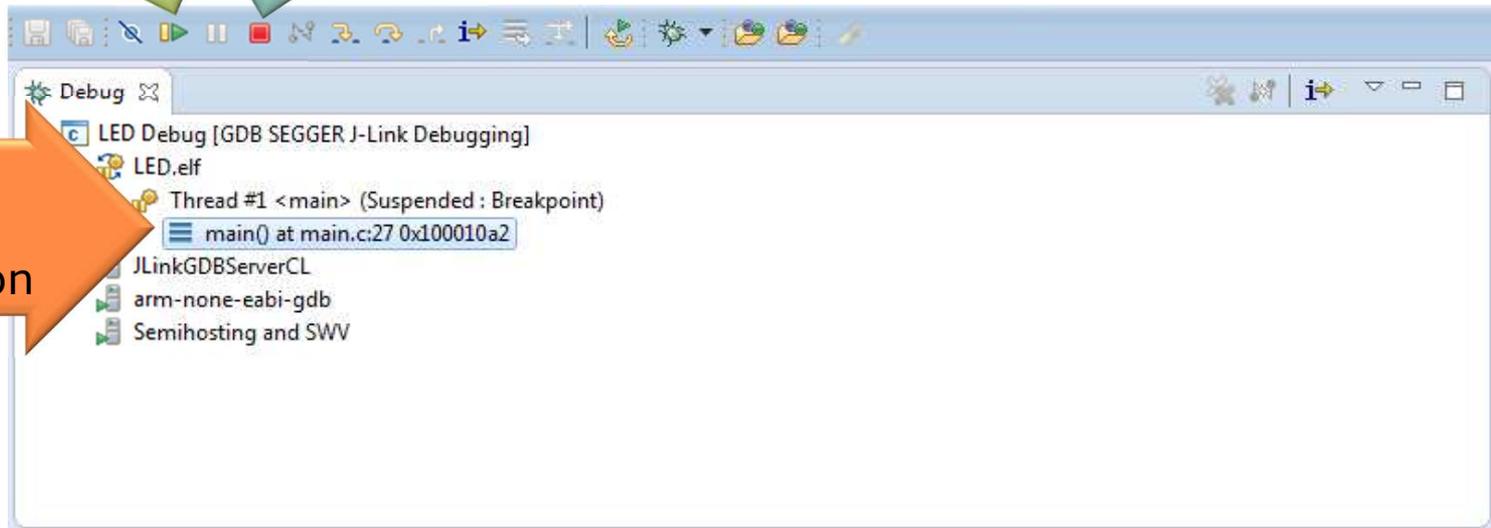
```
100010a2:  adds r4, r7, #7
100010a4:  bl 0x10001a1c <DAVE_Init>
100010a8:  adds r3, r0, #0
100010aa:  strb r3, [r4, #0]
29      if(status == DAVE_STATUS_SUCCESS)
100010ac:  adds r3, r7, #7
100010ae:  ldrb r3, [r3, #0]
100010b0:  cmp r3, #0
100010b2:  beq.n 0x100010b6 <main+26>
39
100010b4:  b.n 0x100010b4 <main+24>
44      PWM_SetDutyCycle(&MyLED, 1000); // set
100010b6:  adds r2, r7, #0
100010b8:  movs r3, #250 ; 0xfa
100010ba:  lsls r3, r3, #2
100010bc:  adds r0, r7, #0
```

Console output:

```
LED Debug [GDB SEGGER J-Link Debugging] JLinkGDBServerCL
Removing breakpoint @ address 0x100010A2, Size = 2
Reading 64 bytes @ address 0x10001080
Reading 64 bytes @ address 0x100010C0
Read 1 bytes @ address 0x2000050F (Data = 0x00)
Read 4 bytes @ address 0x20000508 (Data = 0x00000000)
Read 1 bytes @ address 0x2000050F (Data = 0x00)
Read 4 bytes @ address 0x20000508 (Data = 0x00000000)
```

# The Debug Perspective (2/6)

## Debug Window

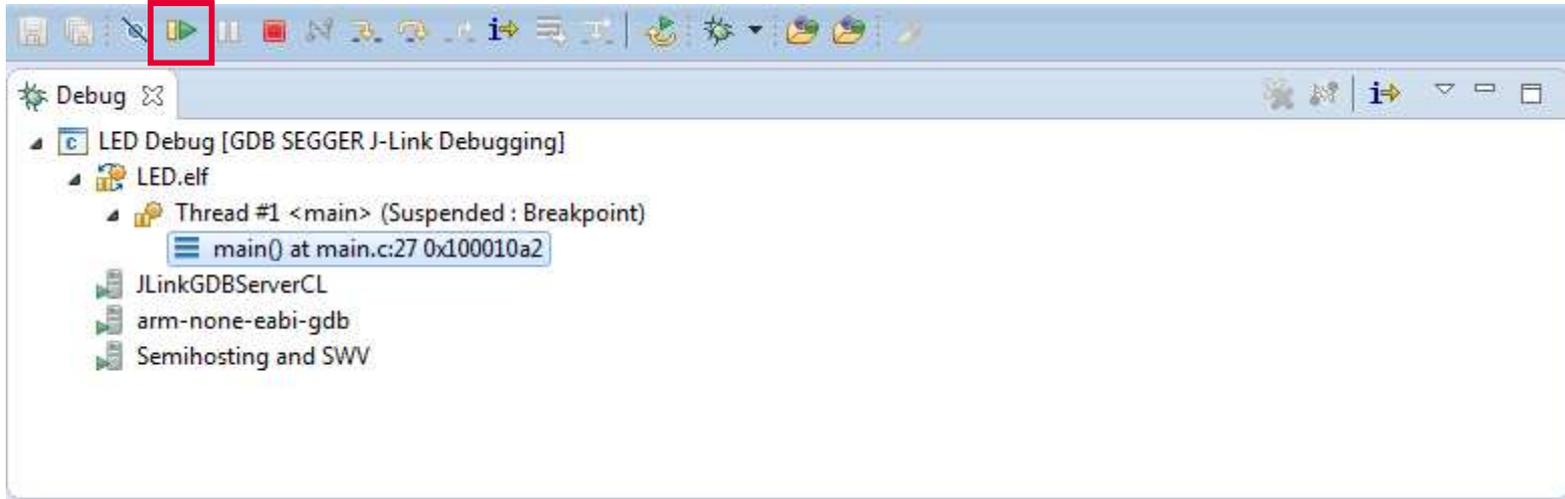


Debug  
Session  
information

# The Debug Perspective (3/6)

## Start Program

- › Click on the Resume button to start code execution



- › User LED1 (P1.0) on XMC2Go board should be blinking



# The Debug Perspective (4/6)

## Breakpoints



- › To place a breakpoint, double-click on the blue bar at the line of code

```
main.c | pwm.c
36     XMC_DEBUG(("DAVE Apps initialization failed with status %d\n", status));
37     while(1U)
38     {
39     }
40 }
41
42 /* Placeholder for user application code. The while loop below can be replaced with user application code. */
43
44 PWM_SetDutyCycle(&MyLED, 1000); // set duty cycle to 10%
45 PWM_SetDutyCycle(&MyLED, 9000); // set duty cycle to 90%
46
47 while(1U)
48 {
49 }
50
51 return 1;
52 }
```

# The Debug Perspective (5/6)

## Single Step

- › After placing breakpoint, click on Step Into or Step Over button to do single stepping
- › Step into (F5)
- › Step over (F6)



The left screenshot shows the Debug Perspective with the following structure:

- LED Debug [GDB SEGGER J-Link Debugging]
  - LED.elf
    - Thread #1 <main> (Suspended : Step)
      - PWM\_SetDutyCycle() at pwm.c:941 0x1000198e
      - main() at main.c:44 0x100010c4
    - JLinkGDBServerCL
    - arm-none-eabi-gdb

The right screenshot shows the Debug Perspective with the following structure:

- LED Debug [GDB SEGGER J-Link Debugging]
  - LED.elf
    - Thread #1 <main> (Running : Step)
    - JLinkGDBServerCL
    - arm-none-eabi-gdb
    - Semihosting and SWV

The code editor shows the following code in pwm.c:

```
931 PWM_ICCU8_Stop(handle_ptr);
932 }
933 #endif
934 }
935
936 /*~~~~~*/
937 /*This function is used to set the duty cycle (uint32_t) of the PWM waveform */
938 PWM_STATUS_t PWM_SetDutyCycle(PWM_t* handle_ptr, uint32_t duty_cycle)
939 {
940     PWM_STATUS_t status;
941     status = PWM_STATUS_FAILURE;
942
943     #ifndef PWM_SLICE_USED_CCU4
944     if (PWM_TIMER_SLICE_CCU4 == handle_ptr->timer_type)
945     {
```

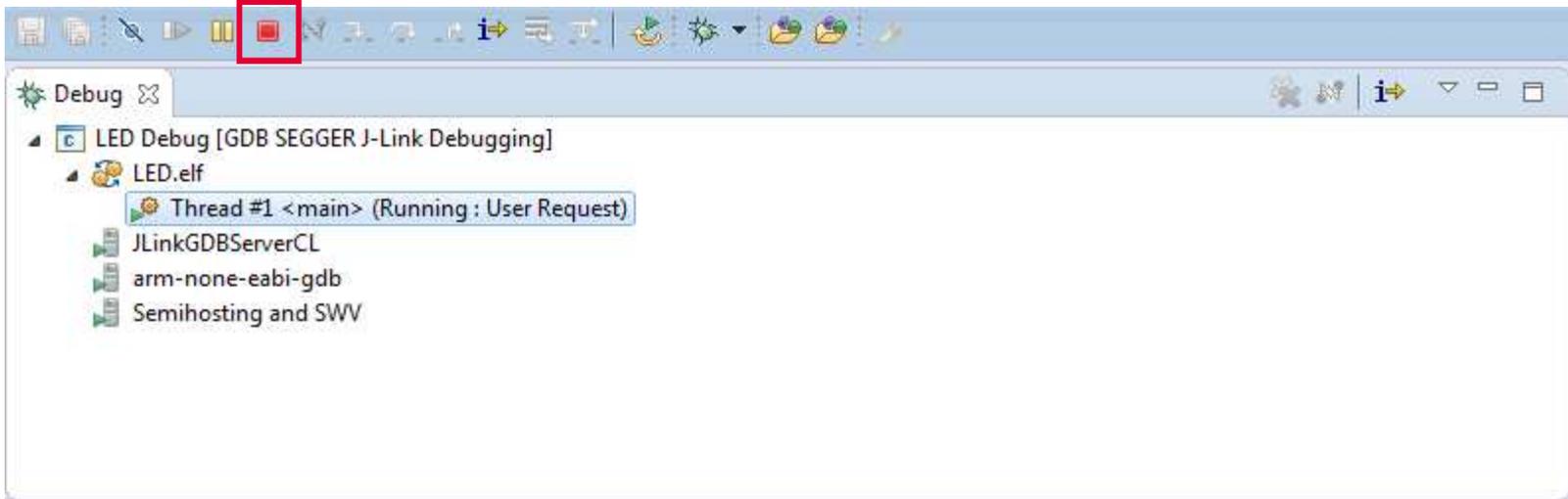
The code editor also shows the following code in main.c:

```
40 }
41
42 /* Placeholder for user application code. The while loop below can be replaced with
43
44 PWM_SetDutyCycle(&MyLED, 1000); // set duty cycle to 10%
45 PWM_SetDutyCycle(&MyLED, 9000); // set duty cycle to 90%
46
47 while(1U)
48 {
49 }
50
51 return 1;
52 }
53
```

# The Debug Perspective (6/6)

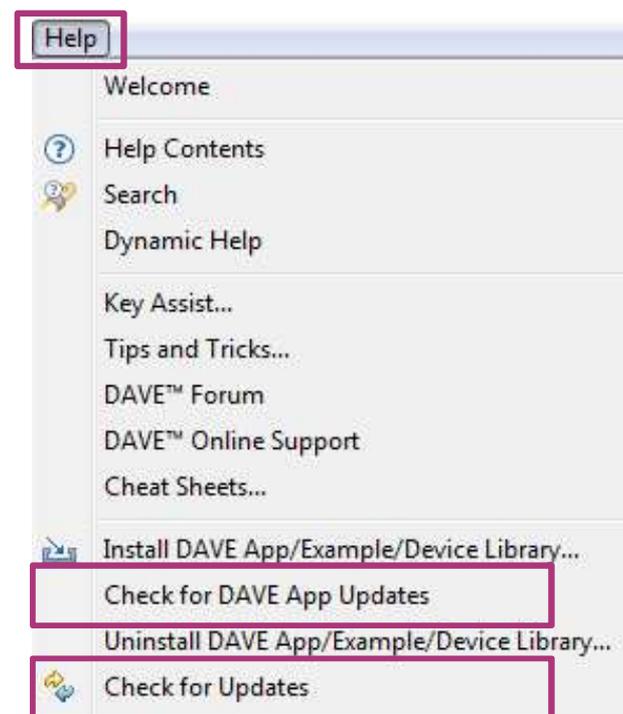
## End Debug Session

- › Always end a debug session by clicking the Terminate Button



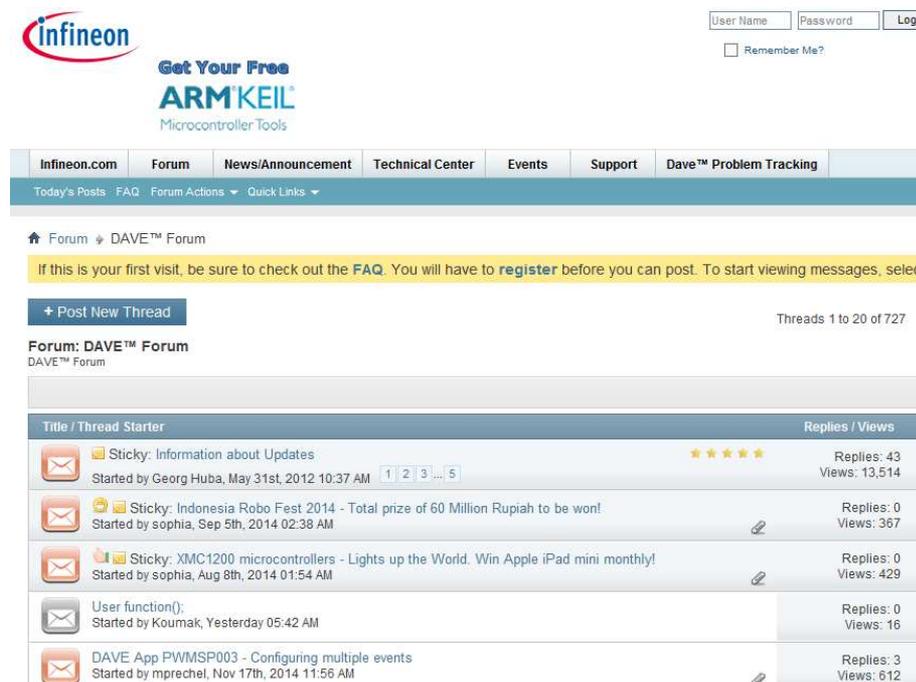
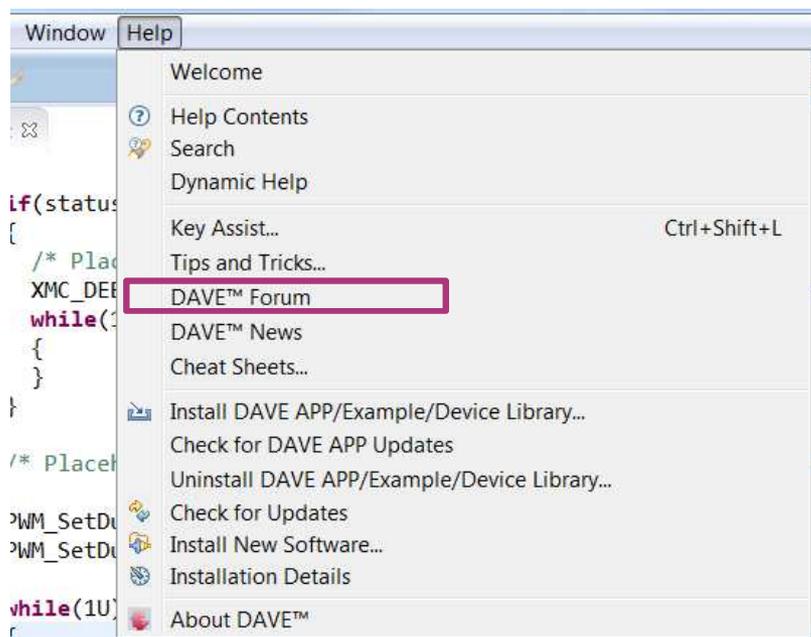
# One-click DAVE™ Update

- › DAVE™ APPs and device support can be updated locally
- › Re-installation not required
  
- › Update DAVE™ system
  - Help → Check for Updates
  
- › Update DAVE™ APPs and device support
  - Help → Check for DAVE App Updates



# Expert Support

- › Easy access to DAVE™ technical support, downloads and information updates



DAVE™ Forum

# Support material:

## Collaterals and Brochures



- › Product Briefs
- › Selection Guides
- › Application Brochures
- › Presentations
- › Press Releases, Ads

› [www.infineon.com/XMC](http://www.infineon.com/XMC)

## Technical Material



- › Application Notes
- › Technical Articles
- › Simulation Models
- › Datasheets, MCDS Files
- › PCB Design Data

› [www.infineon.com/XMC](http://www.infineon.com/XMC)

› [Kits and Boards](#)

› [DAVE™](#)

› [Software and Tool Ecosystem](#)

## Videos



- › Technical Videos
- › Product Information Videos

› [Infineon Media Center](#)

› [XMC Mediathek](#)

## Contact



- › Forums
- › Product Support

› [Infineon Forums](#)

› [Technical Assistance Center \(TAC\)](#)

# Glossary abbreviations

- › DAVE™ Free development IDE for XMC

# Disclaimer

The information given in this training materials is given as a hint for the implementation of the Infineon Technologies component only and shall not be regarded as any description or warranty of a certain functionality, condition or quality of the Infineon Technologies component.

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) with respect to any and all information given in this training material.



Part of your life. Part of tomorrow.

